aws

Guide du développeur

AWS Elastic Beanstalk

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Elastic Beanstalk: Guide du développeur

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Les marques et la présentation commerciale d’Amazon ne peuvent être utilisées en relation avec un produit ou un service qui n’est pas d’Amazon, d’une manière susceptible de créer une confusion parmi les clients, ou d’une manière qui dénigre ou discrédite Amazon. Toutes les autres marques commerciales qui ne sont pas la propriété d’Amazon appartiennent à leurs propriétaires respectifs, qui peuvent ou non être affiliés ou connectés à Amazon, ou sponsorisés par Amazon.

Table of Contents

Qu'est-ce que c'est AWS Elastic Beanstalk ? 1
Plateformes prises en charge 2
Flux de travail de déploiement d'applications 3
Tarification 4
Étapes suivantes 4
Didacticiel de démarrage 5
Ce que vous allez construire 7
Étape 1 - Création d'une application 7
Étape 2 - Déployez votre application 10
Étape 3 - Explorez l'environnement 13
Résolution des problèmes liés aux journaux 16
Étape 4 - Mettez à jour votre application 18
Étape 5 - Faites évoluer votre application 19
Augmenter les paramètres de capacité 20
Vérifiez l'augmentation de la capacité 21
Nettoyage 22
Étapes suivantes 23
Configuration de l'interface de ligne de commande EB 24
Installation de la CLI EB 25
Installation manuelle 26
Installer dans virtualenv 28
Installation de l'interface de ligne de commande EB avec Homebrew 29
Configuration de l'interface de ligne de commande EB 30
Utilisation d'un fichier .ebignore pour ignorer des fichiers 33
Utilisation de profils nommés 33
Déploiement d'un artefact à la place du dossier de projet 34
Configuration des paramètres et des priorités 34
Métadonnées de l’instance 35
Utilisation de l'interface de ligne de commande EB avec Git 35
Associer des environnements Elastic Beanstalk à des branches Git 36
Déploiement des modifications 36
Utilisation des sous-modules Git 37
Affectation de balises Git à votre version de l'application 38
Commandes de l'interface de ligne de commande (CLI) EB 39

Options courantes 40
eb abort 41
Description 41
Syntaxe 41
Options 41
Output 41
Exemple 42
eb appversion 42
Description 42
Syntaxe 42
Options 43
Utilisation interactive de la commande 45
Output 45
Exemples 45
eb clone 46
Description 46
Syntaxe 47
Options 47
Output 49
Exemple 49
eb codesource 50
Description 50
Syntaxe 50
Options 50
Output 50
Exemples 51
eb config 51
Description 51
Syntaxe 52
Options 53
Output 56
Exemples 57
eb console 61
Description 61
Syntaxe 61
Options 61

eb create 61
Description 61
Syntaxe 62
Options 63
Output 77
Exemples 78
eb deploy 80
Description 80
Syntaxe 80
Options 81
Output 82
Exemple 82
eb events 82
Description 82
Syntaxe 83
Options 83
Output 83
Exemple 83
eb health 84
Description 84
Syntaxe 84
Options 84
Output 85
Exemple 85
eb init 86
Description 86
Syntaxe 87
Options 87
CodeBuild soutien 90
Output 90
Exemple 90
eb labs 92
Description 92
eb list 92
Description 92
Syntaxe 92

Options 92
Output 93
Exemple 1 93
Exemple 2 93
eb local 93
Description 93
Syntaxe 94
Options 94
Output 95
Exemples 96
eb logs 97
Description 97
Syntaxe 98
Options 98
Output 102
Exemples 102
eb migrate 103
Description 103
Syntaxe 104
Sous-commandes 105
Options 107
Output 115
Exemples 115
eb open 120
Description 120
Syntaxe 120
Options 120
Output 120
eb platform 120
Description 120
Utilisation de la plateforme eb pour les plateformes personnalisées 121
Utilisation de la plateforme eb pour les environnements 128
eb printenv 131
Description 131
Syntaxe 131
Options 131

Output 131
Exemple 131
eb restore 132
Description 132
Syntaxe 132
Options 132
Output 132
Exemple 132
eb scale 133
Description 133
Syntaxe 133
Options 133
Output 134
Exemple 134
eb setenv 134
Description 134
Syntaxe 134
Options 135
Output 135
Exemple 135
eb ssh 135
Description 135
Syntaxe 136
Options 137
Output 138
Exemple 138
eb status 139
Description 139
Syntaxe 139
Options 139
Output 139
Exemple 140
eb swap 140
Description 140
Syntaxe 141
Options 141

Output 141
Exemples 141
eb tags 142
Description 142
Syntaxe 142
Options 143
Output 144
Exemples 145
eb terminate 146
Description 146
Syntaxe 147
Options 147
Output 147
Exemple 148
eb upgrade 148
Description 148
Syntaxe 149
Options 149
Output 149
Exemple 149
eb use 150
Description 150
Syntaxe 150
Options 150
Concepts 151
Application 152
Version de l'application 152
Environnement 152
Niveau environnemental 152
Configuration de l'environnement 153
Configuration enregistrée 153
Plateforme 153
Environnements de serveur web 153
Environnements Worker 155
Considérations relatives à la conception 156
Capacité de mise à l’échelle 157

Sécurité ... 158
Stockage permanent ... 158
Tolérance aux pannes ... 160
Diffusion de contenu ... 160
Correctifs et mises à jour de logiciels ... 160
Connectivité ... 161
Gestion des applications ... 162
Console de gestion d'application ... 163
Gestion des versions d'application ... 164
Création de versions d'applications ... 165
Supprimer des versions de l'application ... 166
Cycle de vie des versions ... 167
Étiquette des versions d'application ... 169
Création d'une solution groupée source ... 172
Création d'une solution groupée source à partir de la ligne de commande ... 173
Création d'une solution groupée source avec Git ... 173
Compression de fichiers dans le Finder de Mac OS X ou l'Explorateur Windows ... 174
Création d'une solution groupée source pour une application .NET ... 175
Test de votre solution groupée source ... 176
Construire avec CodeBuild ... 176
Création d’une application ... 177
Génération et déploiement du code de votre application ... 177
L'étiquette des applications ... 179
Ajout d'identifications lors de la création d'application ... 180
Gestion des identifications d'une application existante ... 180
Balisage de ressources ... 182
Ressources que vous pouvez étiqueter ... 183
Propagation des balises vers les modèles de lancement ... 183
Gestion des environnements ... 186
Console de gestion de l'environnement ... 188
Accès à la console ... 188
Volet de présentation de l'environnement ... 189
Détail de l'environnement ... 191
Actions dans l'environnement ... 194
Création d'environnements ... 197
Assistant de création d'un environnement ... 202

Clonage d'un environnement ... 225
Résiliation d'un environnement ... 228
Avec le AWS CLI ... 230
Via l'API ... 232
URL Launch Now ... 236
API Compose Environments ... 240
Composition des environnements ... 243
Déploiements ... 246
Choix d'une stratégie de déploiement ... 247
Déploiement d'une nouvelle version de l'application ... 250
Redéploiement d'une version précédente ... 250
Autres méthodes de déploiement de votre application ... 251
Options de déploiement ... 251
Déploiements bleu/vert ... 260
Intégration CI/CD ... 263
Configuration changes ... 266
Mises à jour continues ... 268
Mises à jour immuables ... 273
Mises à jour de plateforme ... 277
Méthode 1 – Mettre à jour la version de la plateforme de votre environnement ... 281
Méthode 2 — Effectuer un Blue/Green déploiement ... 283
Mises à jour gérées ... 285
Mise à niveau d'un environnement hérité ... 292
Migrer vers AL2 023/ AL2 ... 294
FAQ sur le retrait de la plateforme ... 314
Annulation d'une mise à jour ... 319
Reconstruction d'un environnement ... 320
Reconstruction d'un environnement en cours d'exécution ... 321
Reconstruction d'un environnement suspendu ... 321
Types d'environnement ... 323
Environnement évolutif et équilibré en charge ... 324
Environnement à instance unique ... 324
Changement de type d'environnement ... 325
Environnements Worker ... 326
Démon SQS d'environnement de travail ... 328
Files d’attente de lettres mortes ... 330

Tâches périodiques ... 331
Utilisez Amazon CloudWatch pour le dimensionnement automatique dans les différents niveaux de l'environnement de travail ... 332
Configuration des environnements de travail ... 333
Liens entre environnements ... 337
Restauration après un état de pile non valide ... 339
Corriger l'erreur ... 339
Pourquoi l'erreur se produit ... 340
Configuration des environnements ... 342
Ressources provisionnées ... 342
Configuration via la console ... 344
Page de configuration ... 344
Page de vérification des modifications ... 347
EC2 Instances Amazon ... 347
Types d' EC2 instances Amazon ... 349
Configuration à l'aide de la console ... 350
Gestion des groupes de sécurité EC2 ... 359
Configuration à l'aide du AWS CLI ... 361
Configuration avec un espace de noms ... 372
IMDS ... 373
Groupe Auto Scaling ... 376
Modèles de lancement ... 378
Support pour les instances Spot ... 381
Triggers ... 399
Actions planifiées ... 401
Paramètres de vérification de l'état ... 405
Équilibreur de charge ... 407
Classic Load Balancer ... 409
Application Load Balancer ... 421
Application Load Balancer partagé ... 442
Network Load Balancer ... 462
Configuration de la double pile ... 475
Configuration des journaux d'accès ... 486
Base de données ... 487
Cycle de vie de base de données ... 488

Ajout d'une instance de base de données Amazon RDS à votre environnement à l'aide de la console ... 489
Connexion à la base de données ... 491
Configuration d'une instance de base de données RDS intégrée à l'aide de la console ... 492
Configuration d'une instance de base de données RDS intégrée à l'aide des fichiers de configuration ... 493
Découplage d'une instance de base de données RDS à l'aide de la console ... 493
Découplage d'une instance de base de données RDS à l'aide de fichiers de configuration ... 497
Sécurité ... 499
Configuration de la sécurité de votre environnement ... 499
Espaces de noms de configuration de la sécurité de l'environnement ... 502
Environnements de balisage ... 503
Ajout de balises lors de la création de l'environnement ... 503
Gestion des balises d'un environnement existant ... 504
Variables d'environnement et paramètres logiciels ... 507
Configurer les paramètres spécifiques à la plateforme ... 507
Configuration des propriétés de l'environnement (variables d'environnement) ... 509
Espaces de noms des paramètres de logiciel ... 510
Accès aux propriétés de l'environnement ... 512
Débogage ... 514
Affichage des journaux ... 517
Notifications ... 520
Configuration des notifications à l'aide de la console Elastic Beanstalk ... 521
Configuration des notifications à l'aide des options de configuration ... 522
Configuration des autorisations d'envoi de notifications ... 524
Amazon VPC ... 527
Configuration des paramètres VPC dans la console Elastic Beanstalk ... 527
Espace de noms aws:ec2:vpc ... 530
Migration de EC2 -Classic vers un VPC ... 531
Nom de domaine ... 536
Configuration d'environnements (niveau avancé) ... 539
Options de configuration ... 540
Priorité ... 540
Valeurs recommandées ... 542
Avant la création de l'environnement ... 544
Lors de la création ... 550

Après la création ... 557
Options générales ... 568
Options spécifiques à une plateforme ... 668
Options personnalisées ... 681
.Ebextensions ... 683
Paramètres d'option ... 685
Serveur Linux ... 687
Serveur Windows ... 706
Ressources personnalisées ... 716
Configurations enregistrées ... 745
Balisage de configurations enregistrées ... 751
`env.yaml` ... 753
Image personnalisée ... 756
Création d'une AMI personnalisée ... 757
Gestion d'une AMI personnalisée ... 762
Nettoyage d'une AMI personnalisée ... 762
AMI basée sur une plateforme retirée ... 763
Fichiers statiques ... 770
Configurer les fichiers statiques à l'aide de la console ... 770
Configurer des fichiers statiques à l'aide des options de configuration ... 771
HTTPS ... 772
Certificats de serveur ... 774
Mettre fin au protocole HTTPS au niveau de l'équilibreur de charge ... 781
Mettre fin au protocole HTTPS sur l'instance ... 785
End-to-end chiffrement ... 822
TCP Passthrough ... 826
Redirection HTTP vers HTTPS ... 827
Plateformes ... 830
Glossaire des plateformes ... 830
Modèle de responsabilité partagée ... 834
Stratégie de prise en charge de la plateforme ... 836
Branches de plate-forme mises hors service ... 836
Au-delà de la période de grâce de 90 jours ... 837
Calendrier de la plateforme ... 837
Ressources de planification ... 838
Prochaines versions des branches de la plateforme ... 839

Calendrier de mise hors service des branches de plateforme 839
Historique des branches de plateforme retirées 840
Historique du serveur et du système d'exploitation 846
Plateformes prises en charge 847
Plateformes prises en charge et historique des composants 848
Plateformes Linux 849
Versions d'Amazon Linux prises en charge 849
Liste des plateformes Linux Elastic Beanstalk 852
Flux de travail (workflow) de déploiement d'instance 853
Flux de travail de déploiement d'instances pour ECS AL2 à partir de 855
Outils de script de plateforme 858
Extension des plateformes Linux 868
Buildfile et Procfile 869
Hooks de plateforme 871
Fichiers de configuration 873
Configuration du proxy inverse 874
Exemple d'application avec extensions 878
Déploiement de .NET (Windows) 880
QuickStart pour .NET Core sous Windows 881
Votre AWS compte 882
Conditions préalables 884
Étape 1 : Création d'une application .NET Core sous Windows 885
Étape 2 : Exécutez votre application localement 885
Étape 3 : Déployez votre application .NET Core sous Windows avec l'EB CLI 886
Étape 4 : Exécutez votre application sur Elastic Beanstalk 887
Étape 5 : nettoyer 887
AWS ressources pour votre application 887
Étapes suivantes 888
Déployez avec la console 889
QuickStart pour ASP.NET 889
Votre AWS compte 890
Conditions préalables 892
Étape 1 : créer une application ASP.NET 892
Étape 2 : Exécutez votre application localement 894
Étape 3 : Déployez votre application ASP.NET avec AWS Toolkit for Visual Studio 894
Étape 4 : Exécutez votre application sur Elastic Beanstalk 894

Étape 5 : nettoyer 895
AWS ressources pour votre application 895
Étapes suivantes 896
Déployez avec la console 896
Environnement de développement 897
Installation d'un IDE 897
Installation du AWS Toolkit for Visual Studio 897
Plateforme .NET 898
Configuration de votre environnement .NET dans la console Elastic Beanstalk 898
Espace de noms aws:elasticbeanstalk:container:dotnet:apppool 900
Migration entre les versions majeures 900
Manifeste de déploiement 904
EC2 Lancement rapide 930
Ajout d'une base de données 931
Ajout d'une instance de base de données à votre environnement 931
Téléchargement d'un pilote 933
Connexion à une base de données 933
Le AWS Toolkit for Visual Studio 934
Test local 935
Créer un environnement Elastic Beanstalk 935
Résiliation d'un environnement 945
Déploiement 946
Gestion des environnements 949
Gestion des comptes 963
Débogage 964
Contrôle 966
Outils de déploiement 968
Migration de l'application sur site 968
Recommandations relatives aux composants retirés 968
Succursales Windows 2012 R2 retirées 969
Compatibilité avec TLS 1.2 969
Déploiement du .NET Core (Linux) 971
QuickStart pour .NET Core sous Linux 972
Votre AWS compte 972
Conditions préalables 975
Étape 1 : Création d'une application .NET Core sous Linux 975

Étape 2 : Exécutez votre application localement 976
Étape 3 : Déployez votre application .NET Core sur Linux avec l'interface de ligne de commande EB 976
Étape 4 : Exécutez votre application sur Elastic Beanstalk 978
Étape 5 : nettoyer 887
AWS ressources pour votre application 979
Étapes suivantes 980
Déployez à l'aide de la console 980
Environnement de développement 980
Installation du kit SDK .NET Core 980
Installation d'un IDE 897
Installation du AWS Toolkit for Visual Studio 897
Le noyau .NET sur la plateforme Linux 981
Considérations relatives à .NET Core sur la plate-forme Linux 981
Configuration de votre environnement .NET Core sous Linux 982
Espace de noms de la configuration .NET Core sous Linux 984
Regroupement d'applications 984
Procfile 986
Serveur proxy 987
Le AWS Toolkit for Visual Studio 988
Conditions préalables 988
Créer un nouveau projet d'application 989
Création d'un environnement Elastic Beanstalk et déploiement de votre application 990
Résiliation d'un environnement 992
Gestion des environnements 993
Contrôle 1007
Migration de Windows vers Linux 1008
Considérations relatives à la migration vers la plateforme .NET Core sous Linux 1009
Déploiement de Go 1010
QuickStart pour Go 1010
Votre AWS compte 1011
Conditions préalables 1013
Étape 1 : créer une application Go 1013
Étape 2 : Déployez votre application Go avec l'interface de ligne de commande EB 1014
Étape 3 : Exécutez votre application sur Elastic Beanstalk 1015
Étape 4 : nettoyer 887

AWS ressources pour votre application 1016
Étapes suivantes 1017
Déployez avec la console 1017
Environnement de développement 1017
Installation de Go 1017
Installation du AWS SDK pour Go 1018
Plateforme Go 1018
Configuration de votre environnement Go 1019
Espaces de noms de la configuration Go 1020
Procfile 1022
BuildFile 1024
Configuration du proxy 1025
Déploiement de Java 1027
QuickStart pour Java 1028
Votre AWS compte 1029
Conditions préalables 1031
Étape 1 : Création d'une application Java 1032
Étape 2 : Exécutez votre application localement 1034
Étape 3 : Déployez votre application Java avec l'interface de ligne de commande EB 1035
Étape 4 : Exécutez votre application sur Elastic Beanstalk 1036
Étape 5 : nettoyer 1037
AWS ressources pour votre application 1037
Étapes suivantes 1038
Déployez avec la console 1038
QuickStart pour Java sur Tomcat 1039
Votre AWS compte 1039
Conditions préalables 1042
Étape 1 : Création d'une application Java JSP 1042
Étape 2 : Déployez votre application Java JSP avec l'interface de ligne de commande EB 1043
Étape 3 : Exécutez votre application sur Elastic Beanstalk 1044
Étape 4 : nettoyer 887
AWS ressources pour votre application 1044
Étapes suivantes 1045
Déployez avec la console 1046
Environnement de développement 1046
Installation du kit de développement Java 1046

Installation d'un conteneur web 1046
Téléchargement de bibliothèques 1047
Installation du AWS SDK pour Java 1047
Installation d'un IDE ou d'un éditeur de texte 1047
Exemples d'applications et de didacticiels 1048
Lancement d'un environnement avec un exemple d'application Java 1048
Étapes suivantes 1057
Plateforme Tomcat 1057
Configuration de votre environnement Tomcat 1059
Espaces de noms de la configuration Tomcat 1063
Création d'une offre groupée de fichiers WAR 1065
Structuration de votre dossier de projet 1067
Configuration du proxy 1070
Plateforme Java SE 1076
Configuration de votre environnement Java SE 1078
Espaces de noms de la configuration Java SE 1079
BuildFile 1081
Procfile 1081
Configuration du proxy 1083
Ajout d'une base de données 1086
Téléchargement du pilote JDBC 1089
Connexion à une base de données (plateformes Java SE) 1090
Connexion à une base de données (plateformes Tomcat) 1090
Résolution des problèmes de connexion à une base de données 1093
Ressources 1096
Déploiement de Node.js 1097
QuickStart pour Node.js 1097
Votre AWS compte 1098
Conditions préalables 1100
Étape 1 : Création d'une application Node.js 1101
Étape 2 : Exécutez votre application localement 1102
Étape 3 : Déployez votre application Node.js avec l'interface de ligne de commande EB ... 1102
Étape 4 : Exécutez votre application sur Elastic Beanstalk 1103
Étape 5 : nettoyer 887
AWS ressources pour votre application 1104
Étapes suivantes 1105

Déployez avec la console 1105
Environnement de développement 1105
Installation de Node.js. 1106
Confirmation de l'installation de npm 1106
Installez le AWS SDK pour Node.js 1106
Installation du générateur Express 1106
Configuration d'un cadre et d'un serveur Express 1108
Plateforme Node.js 1108
Configuration de votre environnement Node.js 1110
Espaces de noms de la configuration Node.js 1112
Procfile 1116
Configuration des dépendances 1117
Fichier npm shrinkwrap 1122
Configuration du proxy 1122
Exemples d'applications et de didacticiels 1126
Lancement d'un environnement avec un exemple d'application Node.js 1126
Étapes suivantes 1128
Didacticiel - Express 1128
Conditions préalables 1128
Créer un environnement Elastic Beanstalk 1129
Mise à jour de l'application pour utiliser Express 1131
Mise à jour de l'application pour utiliser Amazon RDS 1135
Nettoyage 1140
Didacticiel - Express avec mise en cluster 1140
Conditions préalables 1141
Créer un environnement Elastic Beanstalk 1141
Mise à jour de l'application pour utiliser Express 1144
Nettoyage 1158
Tutoriel - Node.js avec DynamoDB 1158
Conditions préalables 1159
Créer un environnement Elastic Beanstalk 1160
Ajout d'autorisations aux instances de votre environnement 1162
Déploiement de l'exemple d'application 1163
Créez une table DynamoDB 1166
Mise à jour des fichiers de configuration de l'application 1167
Configuration de votre environnement pour une haute disponibilité 1169

Nettoyage 1170
Étapes suivantes 1170
Ajout d'une base de données 1171
Ajout d'une instance de base de données à votre environnement 1171
Téléchargement d'un pilote 1173
Connexion à une base de données 1173
Ressources 1174
Déploiement de PHP 1175
QuickStart pour PHP 1175
Votre AWS compte 1176
Conditions préalables 1178
Étape 1 : Création d'une application PHP 1178
Étape 2 : Exécutez votre application localement 1179
Étape 3 : Initialisation et déploiement de votre application PHP 1179
Étape 4 : Parcourez votre application cloud 1180
Étape 5 : Mettez à jour et redéployez votre application 1180
Nettoyage 1181
Étapes suivantes 1182
Plateforme PHP 1182
Installation du AWS SDK pour PHP 1183
Considérations relatives à PHP 8.1 sur Amazon Linux 2 1183
Configuration de votre environnement PHP 1186
Espaces de noms pour la configuration 1188
Installation des dépendances 1189
Mise à jour de Composer 1191
Extension de php.ini 1192
Exemples avancés 1193
Ajout d'une base de données 1193
Didacticiel - Laravel 1197
Didacticiel - CakePHP 1207
Didacticiel - Symfony 1216
Didacticiel - Production haute disponibilité 1222
Tutoriel - HA WordPress 1234
Didacticiel - Drupal haute disponibilité 1251
Déploiement de Python 1271
QuickStart pour Python 1271

Votre AWS compte 1272
Conditions préalables 1274
Étape 1 : Création d'une application Python 1276
Étape 2 : Exécutez votre application localement 1276
Étape 3 : Déployez votre application Python avec l'EB CLI 1277
Étape 4 : Exécutez votre application sur Elastic Beanstalk 1278
Étape 5 : nettoyer 887
AWS ressources pour votre application 1278
Étapes suivantes 1279
Déployez à l'aide de la console 1280
Environnement de développement 1280
Conditions préalables 1281
Utilisation d’un environnement virtuel 1281
Configuration d'un projet Python pour Elastic Beanstalk 1283
Plateforme Python 1283
Configuration de votre environnement Python 1285
Espaces de noms de la configuration Python 1287
L'`python3`exécutable 1289
Procfile 1289
Spécification des dépendances 1290
Didacticiel - Flask 1293
Conditions préalables 1293
Configuration d'un environnement virtuel Python avec Flask 1294
Création d'une application Flask 1295
Déploiement de votre site avec l'interface de ligne de commande (CLI) EB 1297
Nettoyage 1301
Étapes suivantes 1301
Didacticiel - Django 1301
Conditions préalables 1302
Configuration d'un environnement virtuel Python et installation de Django 1302
Création d'un projet Django 1304
Configurer votre application Django pour Elastic Beanstalk 1306
Déploiement de votre site avec l'interface de ligne de commande (CLI) EB 1308
Mise à jour de votre application 1311
Nettoyage 1315
Étapes suivantes 1315

Ajout d'une base de données 1316
Ajout d'une instance de base de données à votre environnement 1316
Téléchargement d'un pilote 1317
Connexion à une base de données 1318
Ressources 1319
Déployer Ruby 1320
Environnement de développement 1320
Installation de Ruby 1321
Installation du AWS SDK pour Ruby 1322
Installation d'un IDE ou d'un éditeur de texte 1322
La plateforme Ruby 1323
Configuration de votre environnement Ruby 1324
Espaces de noms de configuration Ruby 1327
Gemfile 1328
Procfile 1329
Didacticiel - Rails 1331
Conditions préalables 1331
Connaissances de base sur Elastic Beanstalk 1331
Lancer un environnement Elastic Beanstalk 1332
Installation de Rails et génération d'un site web 1334
Configuration des paramètres Rails 1338
Déploiement de votre application 1339
Nettoyage 1339
Étapes suivantes 1340
Didacticiel - Sinatra 1340
Conditions préalables 1340
Lancer un environnement Elastic Beanstalk 1341
Écriture d'un site web Sinatra 1343
Déploiement de votre application 1344
Nettoyage 1345
Étapes suivantes 1345
Ajout d'une base de données 1346
Ajout d'une instance de base de données à votre environnement 1346
Téléchargement d'une carte 1348
Connexion à une base de données 1348
Déploiement avec Docker 1349

Branches de la plateforme Docker 1349
Branches de plateforme retirées exécutées sur l'AMI Amazon Linux (AL1) 1350
Branche de plateforme Docker 1351
QuickStart pour Docker 1351
QuickStart pour Docker Compose 1359
Configuration de l'image Docker 1371
Branche de plateforme gérée par ECS 1377
Présentation de la plateforme Docker gérée par ECS 1378
Ressources Amazon ECS créées par Elastic Beanstalk 1378
`Dockerrun.aws.json`fichier v2 1379
Images Docker 1380
Échec de déploiements de conteneurs 1380
Extension des plateformes Docker basées sur ECS 1381
Configuration Docker gérée par ECS pour Elastic Beanstalk 1381
Tutoriel : Docker géré par ECS 1394
Migration vers ECS en cours d'exécution sur AL2 023 1404
Authentification avec des référentiels d'images 1406
Référentiel Amazon ECR 1406
AWS Secrets Manager 1408
Fichier Dockerrun.aws.json 1410
Images publiques d'Amazon ECR 1412
Configuration de l'environnement 1413
Configuration des logiciels dans les environnements Docker 1414
Référencement de variables d'environnement dans les conteneurs 1416
Utilisation de la fonction d'interpolation pour les variables d'environnement avec Docker Compose 1418
Génération de journaux pour des rapports de santé améliorés avec Docker Compose 1419
Journalisation personnalisée des conteneurs Docker avec Docker Compose 1419
Images Docker 1421
Configuration des mises à jour gérées pour les environnements Docker 1421
Espaces de noms de la configuration Python 1422
Configuration Docker sur l'AMI Amazon Linux (antérieure à Amazon Linux 2) 1423
Plateformes traditionnelles 1425
(Hérité) Migration vers Docker s'exécutant sur Amazon Linux 2 1425
Conteneurs Docker GlassFish (anciens) 1431
Environnements de surveillance 1441

Console de surveillance 1441
Graphiques de surveillance 1442
Personnalisation de la console de surveillance 1443
Surveillance de l'état de santé avec la CLI 1444
Lecture du résultat 1448
Vue d'intégrité interactive 1450
Options d'affichage d'intégrité interactif 1452
Création de rapports d'intégrité de base 1452
Couleurs de l'intégrité 1453
Vérifications de l'état Elastic Load Balancing 1454
Vérifications de l'état d'un environnement à instance unique et d'un environnement de travail 1455
Contrôles supplémentaires 1455
CloudWatch Métriques Amazon 1456
Rapports d'intégrité améliorés et surveillance 1457
Agent de vérification de l'état Elastic Beanstalk 1460
Facteurs de détermination de l'intégrité de l'environnement et de l'instance 1461
Personnalisation d'une règle de vérification de l'état 1464
Rôles d'intégrité améliorée 1464
Autorisation de santé améliorée 1465
Événements d'intégrité améliorée 1466
Comportement de la création de rapports d'intégrité améliorée au cours des mises à jour, des déploiements et de la mise à l'échelle 1467
Activation des rapports d'intégrité améliorée 1467
Console de surveillance de l'état 1471
Couleurs et états utilisés dans les rapports d'intégrité 1476
Métriques des instances 1479
Règles d'intégrité améliorée 1482
CloudWatch 1488
Utilisateurs de l'API 1497
Format de journal d'intégrité améliorée 1499
Notifications et dépannage 1503
Analyse de l'environnement basée sur l'IA 1505
Gestion des alarmes 1509
Afficher l'historique des modifications 1511
Affichage des événements 1513

Console ... 1513
Ligne de commande ... 1514
Surveillance des instances ... 1514
Affichage des journaux d'instance ... 1516
Emplacement des journaux sur les instances Amazon EC2 ... 1518
Emplacement des journaux dans Amazon S3 ... 1519
Paramètres de rotation des journaux sous Linux ... 1520
Extension de la configuration de tâche de journal par défaut ... 1521
Streaming de fichiers journaux vers Amazon CloudWatch Logs ... 1524
Journaux de déploiement ... 1525
Opérations prises en charge ... 1526
Contenu des journaux ... 1526
Console ... 1528
Fichiers journaux d'instance ... 1529
Intégration des AWS services ... 1531
Présentation de l'architecture ... 1531
CloudFront ... 1532
CloudTrail ... 1533
Informations sur Elastic Beanstalk dans CloudTrail ... 1533
Présentation des entrées du fichier journal Elastic Beanstalk ... 1534
CloudWatch ... 1536
CloudWatch Journaux ... 1536
Conditions préalables à la diffusion des journaux d'instance vers Logs CloudWatch ... 1537
Comment Elastic Beanstalk configure les journaux CloudWatch ... 1538
Streaming des logs d'instance vers CloudWatch Logs ... 1544
Résolution des problèmes et intégration CloudWatch des journaux ... 1546
Diffusion des informations d'intégrité de l'environnement ... 1548
EventBridge ... 1551
Surveillez une ressource Elastic Beanstalk avec EventBridge ... 1552
Exemple de modèles d'événements Elastic Beanstalk ... 1555
Exemple d'événement Elastic Beanstalk ... 1557
Mappage des champs d'événement Elastic Beanstalk ... 1559
AWS Config ... 1561
Con AWS Config figuration ... 1562
Configuration AWS Config pour enregistrer les ressources Elastic Beanstalk ... 1562
Afficher les détails de configuration d'Elastic Beanstalk dans la console AWS Config ... 1563

Évaluation des ressources Elastic Beanstalk à l'aide de AWS Config règles 1567
DynamoDB 1568
ElastiCache 1569
Amazon EFS 1570
Fichiers de configuration 1570
Systèmes de fichiers chiffrés 1571
Exemples d’applications 1572
Nettoyage de systèmes de fichiers 1572
IAM 1573
Profils d’instance 1573
Rôles du service 1578
Utilisation des rôles liés à un service 1587
Stratégies utilisateur 1599
Format ARN 1607
Ressources et conditions 1609
Contrôle d’accès basé sur les étiquettes 1655
Exemples de stratégies gérées 1660
Exemple de stratégies spécifiques aux ressources 1664
Accès au compartiment S3 entre environnements 1676
Amazon RDS 1678
Amazon RDS dans un VPC par défaut 1680
Informations d'identification Amazon RDS et Secrets Manager 1686
Nettoyage d'une instance Amazon RDS externe 1686
Amazon S3 1687
Le compartiment de comptes client Amazon S3 d'Elastic Beanstalk 1687
Contenu du compartiment de comptes client Elastic Beanstalk Amazon S3 1687
Suppression d'objets dans le compartiment Elastic Beanstalk Amazon S3 1689
Suppression du compartiment Elastic Beanstalk Amazon S3 1689
Gestionnaire de Secrets et Responsable des systèmes Parameter Store 1691
Récupérez les secrets des variables d'environnement 1691
Autorisations IAM requises 1702
Utilisation de Secrets Manager et du magasin de paramètres de Systems Manager 1704
Secrets de résolution des problèmes et variables d'environnement 1707
Amazon VPC 1708
VPC public 1710
VPC public/privé 1711

VPC privé ... 1711
Hôtes bastions ... 1714
Amazon RDS ... 1719
Points de terminaison d'un VPC ... 1726
Politiques de point de terminaison d'un VPC ... 1731
Sécurité ... 1742
Protection des données ... 1743
Chiffrement des données ... 1744
Trafic inter-réseaux ... 1745
Gestion des identités et des accès ... 1746
AWS politiques gérées ... 1746
Journalisation et surveillance ... 1763
Création de rapports d'intégrité améliorée ... 1763
Journaux des EC2 instances Amazon ... 1763
Notifications de l'environnement ... 1764
CloudWatch Alarmes Amazon ... 1764
AWS CloudTrail journaux ... 1764
AWS X-Ray débogage ... 1764
Validation de conformité ... 1765
Résilience ... 1766
Sécurité de l'infrastructure ... 1766
Modèle de responsabilité partagée ... 1767
Bonnes pratiques de sécurité ... 1767
Bonnes pratiques de sécurité préventive ... 1767
Bonnes pratiques de sécurité de détection ... 1769
Permissions ... 1771
Rôles obligatoires ... 1771
Politiques et rôles facultatifs ... 1771
Rôle de service ... 1772
`AWSElasticBeanstalkEnhancedHealth` ... 1579
`AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy` ... 1580
Profil d'instance ... 1774
Politiques gérées ... 1775
Création d'un profil d'instance EC2 ... 1775
Stratégie utilisateur ... 1775
Tutoriels et exemples ... 1777

Migration des applications IIS 1780
Conditions préalables 1781
Glossaire des migrations 1783
Termes relatifs à Windows, IIS et .NET 1784
Termes relatifs à Elastic Beanstalk 1788
Termes de Python 1792
Migrations IIS de base 1793
Exploration de votre environnement IIS 1793
Préparation de la migration 1795
Votre première migration 1796
Contrôle de la migration 1797
Surveillance de la progression 1797
Vérification de la migration 1798
Gestion des artefacts de migration 1799
Configuration réseau 1799
Configuration VPC 1799
Déploiements multisites avec configurations de ports 1804
Configuration et dépendances partagées 1806
Bonnes pratiques 1807
Résolution des problèmes 1807
Configuration de sécurité 1808
Configuration du profil d'instance 1808
Gestion des rôles de service 1809
Configuration du groupe de sécurité 1810
Intégration des certificats SSL 1811
Authentification Windows 1811
Bonnes pratiques et dépannage 1812
Cartographie de la migration entre IIS et Elastic Beanstalk 1815
Sites et applications IIS dans Elastic Beanstalk 1815
Gestion des répertoires virtuels et des chemins d'application 1816
Réécriture d'URL et routage des demandes d'applications (ARR) 1817
Structure des artefacts de migration 1818
Scénarios de migration avancés 1820
Migrations multisites avec routage des demandes d'application (ARR) 1820
Migrations multisites sans ARR à l'aide d'un routage basé sur l'hôte 1823
Gestion des annuaires virtuels 1827

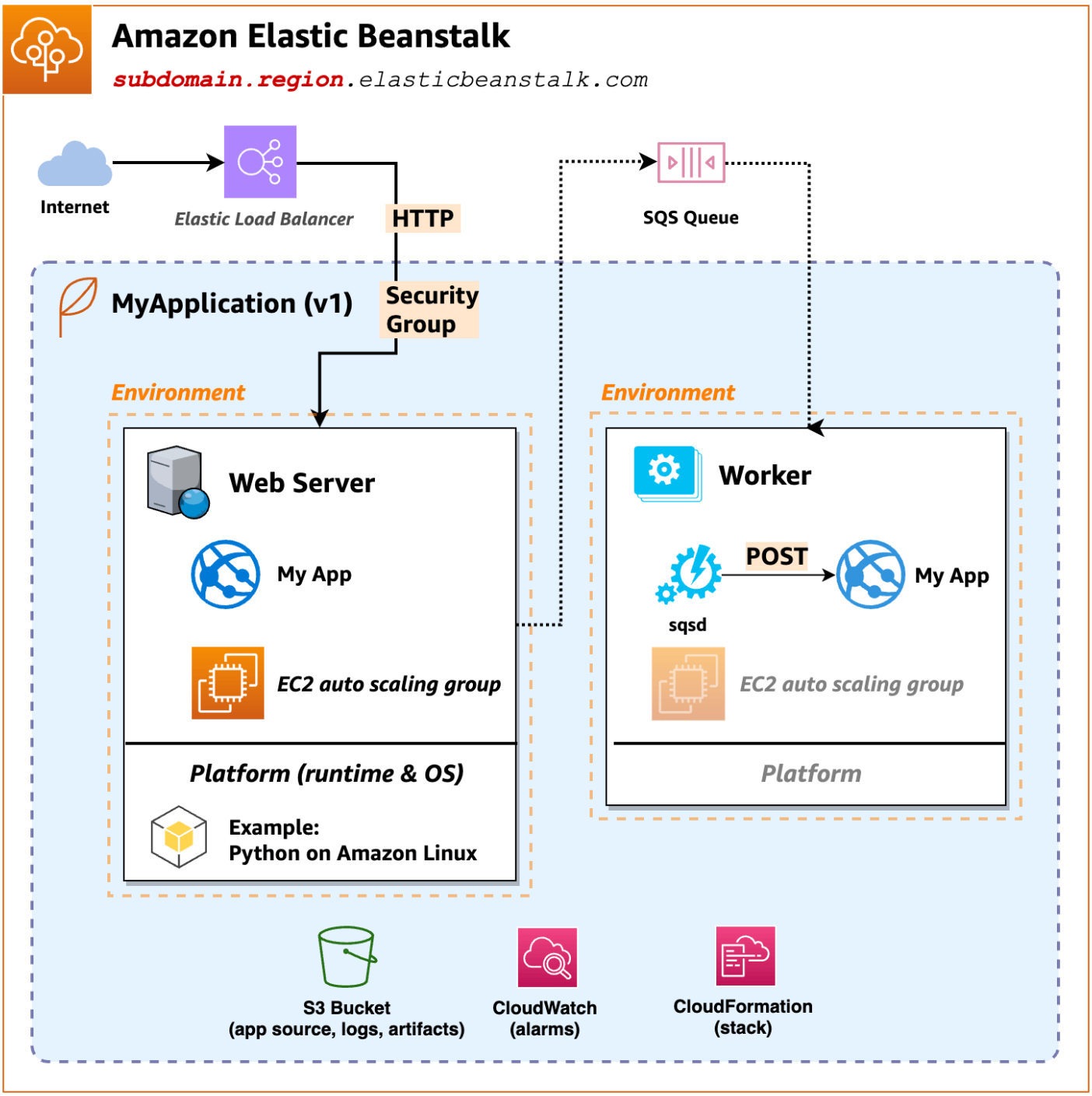
Paramètres personnalisés du pool d'applications 1831
Déploiement des versions précédentes 1833
Dépannage et diagnostic 1834
Associer une EC2 paire de clés à votre environnement 1834
Accès aux journaux 1835
Accès aux artefacts côté client 1835
Surveillance de la santé de l'environnement 1835
EC2 optimisation des performances 1836
Configuration du volume EBS 1837
Problèmes courants et solutions correspondantes 1837
Obtention de support 1838
Options de migration : EB CLI ou MGN 1838
Quand utiliser chaque option de migration 1840
Comparaison des flux de travail de migration 1840
Conclusion 1841
Résolution des problèmes 1842
Utilisation de l'analyse de l'environnement basée sur l'IA 1843
Utilisation de l'outil Systems Manager 1843
General guidance 1845
Variables d'environnement pour les secrets 1846
Création de l'environnement 1847
Déploiements 1847
Santé 1848
Équilibreurs de charge configurés à double pile 1850
Configuration 1851
Docker 1851
FAQ 1852
Résolution de problèmes de déploiement 1853
Erreurs de déploiement 1854
Ressources 1858
Exemples d’applications 1859
AWS SDK pour Java 1860
AWS SDK pour .NET 1860
AWS Toolkit for Visual Studio 1860
AWS SDK pour JavaScript Node.js 1861
AWS SDK pour PHP 1861

AWS SDK pour Python (Boto) 1861
AWS SDK pour Ruby 1861
Contenu archivé 1863
Plateformes personnalisées (retirées) 1863
Historique du document 1881
..... mdccclxxxviii

Qu'est-ce que c'est AWS Elastic Beanstalk ?

Avec Elastic Beanstalk, vous pouvez déployer des applications Web AWS dans le cloud sur diverses plateformes prises en charge. Vous créez et déployez vos applications. Elastic Beanstalk EC2 approvisionne les instances Amazon, configure l'équilibrage de charge, met en place le suivi de l'état de santé et adapte dynamiquement votre environnement.

Outre les environnements de serveur Web, Elastic Beanstalk propose également des environnements de travail que vous pouvez utiliser pour traiter les messages provenant d'une file d'attente Amazon SQS, utiles pour les tâches asynchrones ou de longue durée. Pour de plus amples informations, veuillez consulter Environnements de travail Elastic Beanstalk.



Plateformes prises en charge

Elastic Beanstalk prend en charge les applications développées dans Go, Java, .NET, Node.js, PHP, Python et Ruby. Elastic Beanstalk prend également en charge les conteneurs Docker, dans lesquels vous pouvez choisir votre propre langage de programmation et les dépendances des applications.

Lorsque vous déployez votre application, Elastic Beanstalk crée la version de plateforme prise en charge sélectionnée et fournit une AWS ou plusieurs ressources, telles que des instances EC2 Amazon, AWS dans votre compte pour exécuter votre application.

Vous pouvez interagir avec Elastic Beanstalk via la AWS Command Line Interface console Elastic Beanstalk, le () ou l'EB CLI AWS CLI, un outil de ligne de commande de haut niveau conçu spécifiquement pour Elastic Beanstalk.

Vous pouvez effectuer la plupart des tâches de déploiement, telles que la modification de la taille de votre parc d' EC2 instances Amazon ou le suivi de votre application, directement depuis l'interface Web d'Elastic Beanstalk (console).

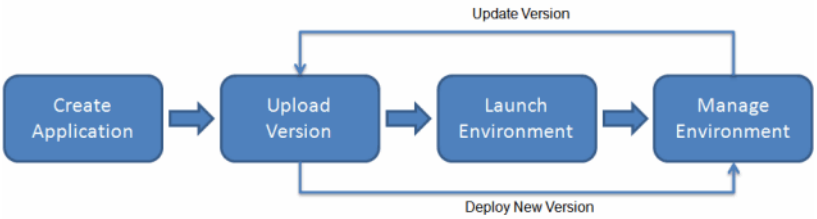
Pour en savoir plus sur le déploiement d'un exemple d'application Web à l'aide d'Elastic Beanstalk, consultez. Découvrez comment démarrer avec Elastic Beanstalk

Flux de travail de déploiement d'applications

Pour utiliser Elastic Beanstalk, vous devez créer une application, puis télécharger le bundle de sources de votre application dans Elastic Beanstalk. Ensuite, vous fournissez des informations sur l'application, et Elastic Beanstalk lance automatiquement un environnement et crée et AWS configure les ressources nécessaires pour exécuter votre code.

Une fois que vous avez créé et déployé votre application et que votre environnement est lancé, vous pouvez gérer votre environnement et déployer de nouvelles versions de l'application. Les informations relatives à l'application, notamment les métriques, les événements et l'état de l'environnement, sont mises à disposition via la console Elastic Beanstalk et les interfaces de ligne de commande. APIs

Le schéma suivant illustre le flux de travail d'Elastic Beanstalk :



Tarification

Il n'y a aucun frais supplémentaire pour Elastic Beanstalk. Vous ne payez que pour les AWS ressources sous-jacentes consommées par votre application. Pour de plus amples informations sur la tarification, veuillez consulter la page détaillée du service Elastic Beanstalk.

Étapes suivantes

Nous vous recommandons de suivre ce didacticiel pour commencer à utiliser Elastic Beanstalk. Didacticiel de démarrage Ce didacticiel vous explique comment créer, visualiser et mettre à jour un exemple d'application Elastic Beanstalk.

Découvrez comment démarrer avec Elastic Beanstalk

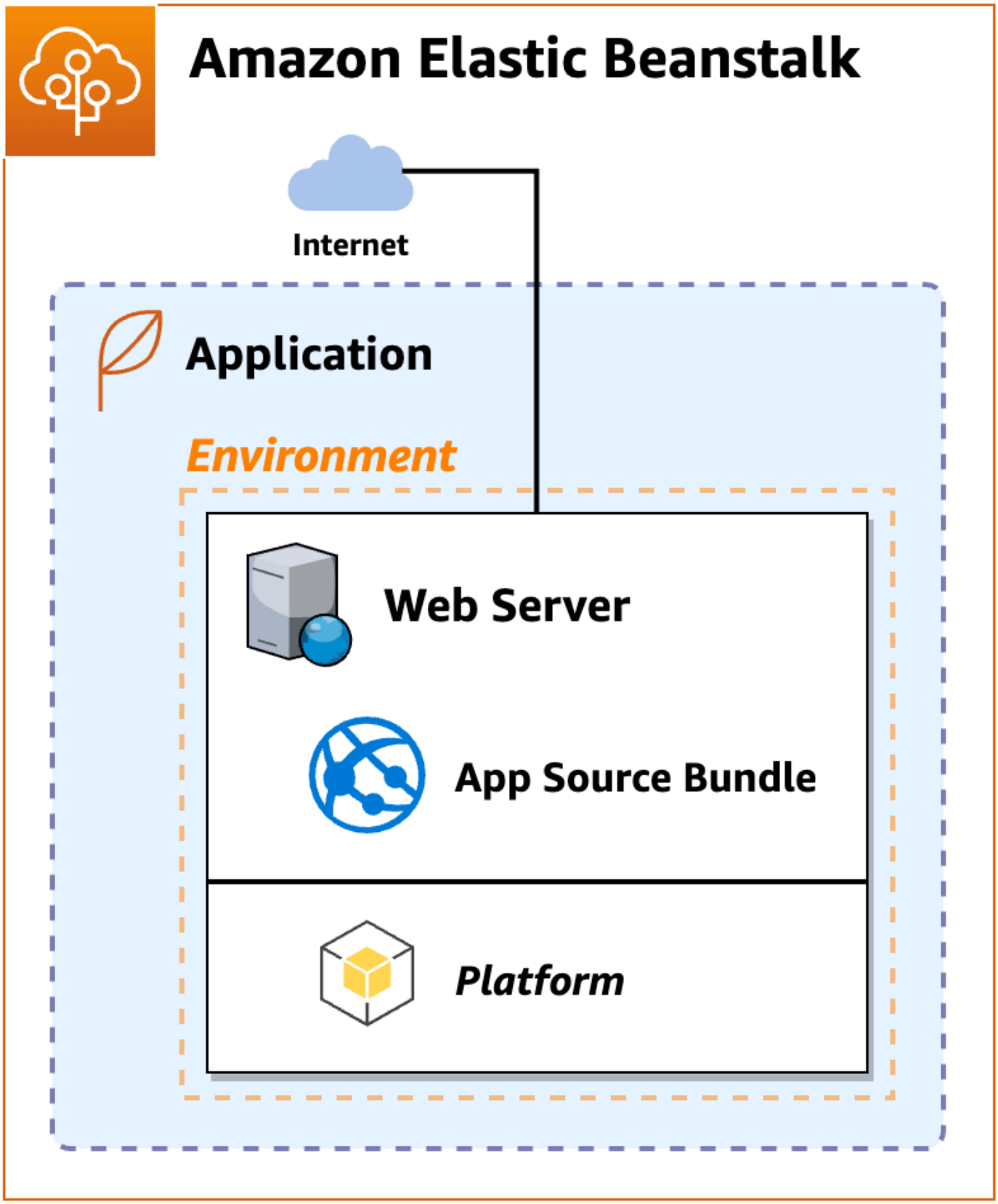
Avec Elastic Beanstalk, vous pouvez déployer, surveiller et dimensionner des applications et des services Web. Généralement, vous développez votre code localement, puis vous le déployez sur des instances de serveur Amazon EC2. Ces instances, également appelées environnements, s'exécutent sur des plateformes qui peuvent être mises à niveau via la AWS console ou la ligne de commande.

Pour commencer, nous vous recommandons de déployer un exemple d'application prédéfini directement depuis la console. Vous pouvez ensuite apprendre à développer localement et à déployer à partir de la ligne de commande duthe section called “QuickStart pour PHP”.

L'utilisation d'Elastic Beanstalk est gratuite, mais les frais standard AWS s'appliquent aux ressources que vous créez au cours de ce didacticiel jusqu'à ce que vous les supprimiez à la fin. Les frais totaux sont généralement inférieurs à un dollar. Pour en savoir sur la façon de réduire les frais, veuillez consulter offre gratuite AWS .

Après avoir suivi ce didacticiel, vous comprendrez les principes de base de la création, de la configuration, du déploiement, de la mise à jour et de la surveillance d'une application Elastic Beanstalk avec des environnements exécutés sur des instances Amazon EC2.

Durée estimée : 35 à 45 minutes


Amazon Elastic Beanstalk
Internet
Application
Environment
Web Server
App Source Bundle
Platform

Ce que vous allez construire

Votre première application Elastic Beanstalk consistera en un seul environnement Amazon EC2 exécutant l'exemple PHP sur une plateforme gérée par PHP.

Application Elastic Beanstalk

: Une application Elastic Beanstalk est un conteneur pour les composants Elastic Beanstalk, y compris les environnements dans lesquels le code de votre application s'exécute sur des plateformes fournies et gérées par Elastic Beanstalk, ou dans des conteneurs personnalisés que vous fournissez.

Environnement

: Un environnement Elastic Beanstalk est un ensemble de ressources fonctionnant AWS ensemble, y compris une instance Amazon EC2. Lorsque vous créez un environnement, Elastic Beanstalk fournit les ressources nécessaires à votre compte. AWS

Plateforme

: Une plate-forme est une combinaison d'un système d'exploitation, d'un environnement d'exécution d'un langage de programmation, d'un serveur Web, d'un serveur d'applications et de composants Elastic Beanstalk supplémentaires. Elastic Beanstalk fournit des plateformes gérées, mais vous pouvez également fournir votre propre plate-forme dans un conteneur.

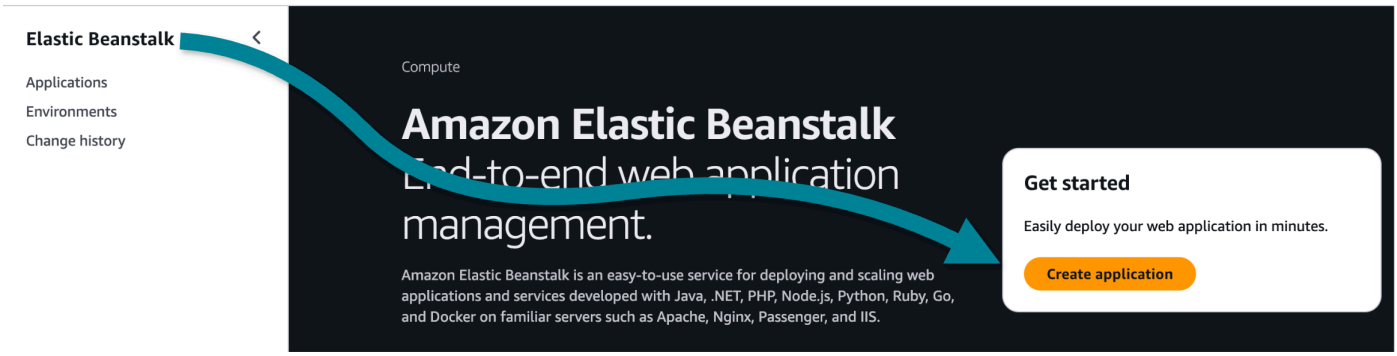
Elastic Beanstalk prend en charge les plateformes pour différents langages de programmation, serveurs d'applications et conteneurs Docker. Lorsque vous créez un environnement, vous devez choisir la plateforme. Vous pouvez mettre à niveau la plate-forme, mais vous ne pouvez pas la modifier pour un environnement.

> **Plateformes de commutation**
>
> Si vous devez changer de langage de programmation, vous devez créer un nouvel environnement sur une autre plateforme et passer à un nouvel environnement.

Étape 1 - Création d'une application

Pour créer votre exemple d'application, vous allez utiliser l'assistant de console Create application (Créer une application). Il crée une application Elastic Beanstalk et y lance un environnement.

Rappel : un environnement est un ensemble de AWS ressources nécessaires à l'exécution du code de votre application.



Pour créer une application

1. Ouvrez la console Elastic Beanstalk.
2. Choisissez Create application (Créer une application).
3. Dans Application name (Nom de l'application), saisissez **`getting-started-app`**.

La console propose un processus en six étapes pour créer une application et configurer un environnement. Pour ce démarrage rapide, vous n'aurez qu'à vous concentrer sur les deux premières étapes, puis vous pourrez passer à la révision et à la création de votre application et de votre environnement.

Pour configurer un environnement

1. Dans Informations sur l'environnement, dans le champ Nom de l'environnement, entrez :**`gs-app-web-env`**.
2. Pour Platform, choisissez la plateforme PHP.
3. Pour le code d'application et les préréglages, acceptez les valeurs par défaut (exemple d'application et instance unique), puis choisissez Next.

Pour configurer l'accès au service

Ensuite, vous avez besoin de deux rôles. Un rôle de service permet à Elastic Beanstalk de surveiller vos instances EC2 et de mettre à niveau la plateforme de votre environnement. Un rôle de profil

d'instance EC2 permet des tâches telles que la rédaction de journaux et l'interaction avec d'autres services.

Pour créer ou sélectionner le rôle de service

1. Si vous avez déjà créé un rôle de service et que vous souhaitez en choisir un existant, sélectionnez la valeur dans le menu déroulant Rôle de service et ignorez le reste de ces étapes pour créer un rôle de service.
2. Si aucune valeur n'est répertoriée pour le rôle de service, ou si vous souhaitez en créer un nouveau, passez aux étapes suivantes.
3. Pour Rôle de service, choisissez Créer un rôle.
4. Pour Type d’entité de confiance, choisissez Service AWS .
5. Dans le cas d'utilisation, choisissez Elastic Beanstalk — Environment.
6. Choisissez Suivant.
7. Vérifiez que les politiques d'autorisations incluent les éléments suivants, puis choisissez Next :
   - `AWSElasticBeanstalkEnhancedHealth`
   - `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy`
8. Choisissez Créer un rôle.
9. Retournez à l'onglet Configurer l'accès au service, actualisez la liste, puis sélectionnez le rôle de service nouvellement créé.

Pour créer ou sélectionner un profil d'instance EC2

1. Si vous avez déjà créé un profil d'instance EC2 et que vous souhaitez en choisir un existant, sélectionnez la valeur dans le menu déroulant du profil d'instance EC2 et ignorez le reste de ces étapes pour créer un profil d'instance EC2.
2. Si aucune valeur n'est répertoriée pour le profil d'instance EC2, ou si vous souhaitez en créer un nouveau, passez aux étapes suivantes.
3. Choisissez Créer un rôle.
4. Pour Type d’entité de confiance, choisissez Service AWS .
5. Dans le cas d'utilisation, choisissez Elastic Beanstalk — Compute.
6. Choisissez Suivant.
7. Vérifiez que les politiques d'autorisations incluent les éléments suivants, puis choisissez Next :

- `AWSElasticBeanstalkWebTier`
- `AWSElasticBeanstalkWorkerTier`
- `AWSElasticBeanstalkMulticontainerDocker`

8. Choisissez Créer un rôle.
9. Retournez à l'onglet Configurer l'accès au service, actualisez la liste, puis sélectionnez le profil d'instance EC2 nouvellement créé.

Pour terminer la configuration et la création de votre application

1. Ignorez la paire de clés EC2.

   Nous allons vous montrer d'autres moyens de vous connecter à vos instances Amazon EC2 via la console.
2. Choisissez Passer à la révision pour passer en revue plusieurs étapes facultatives.

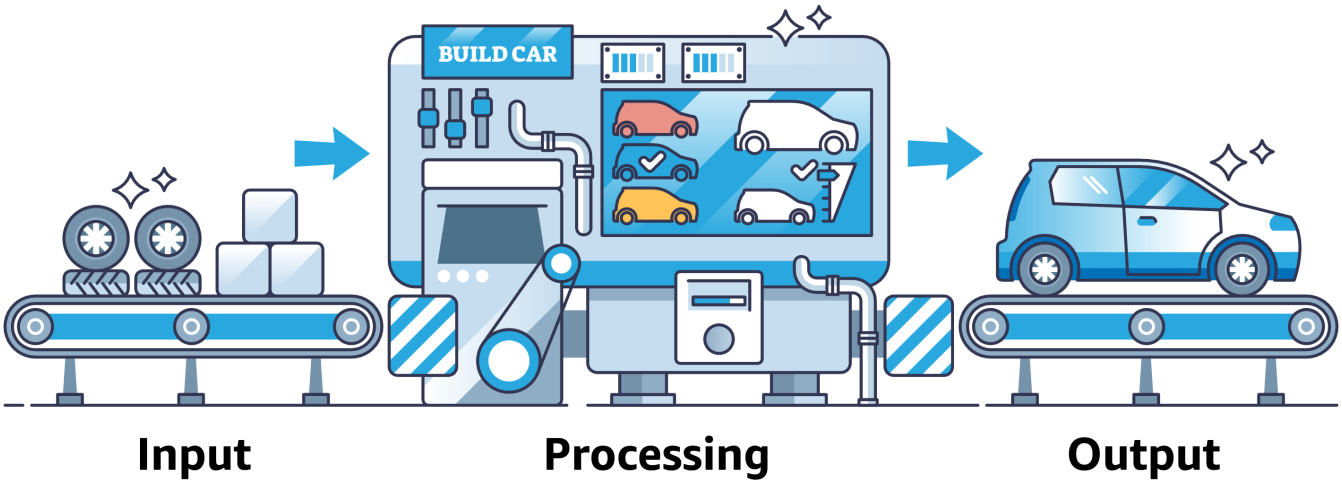
   Étapes facultatives : mise en réseau, bases de données, paramètres de dimensionnement, configuration avancée pour les mises à jour, surveillance et journalisation.
3. Sur la page Révision qui affiche un résumé de vos choix, choisissez Soumettre.

> Félicitations !
>
> Vous avez créé une application et configuré un environnement ! Vous devez maintenant attendre que les ressources soient déployées.

Étape 2 - Déployez votre application

Lorsque vous créez une application, Elastic Beanstalk configure les environnements pour vous. Il vous suffit de vous asseoir et d'attendre.

Le déploiement initial peut prendre jusqu'à cinq minutes pour créer les ressources. Les mises à jour prendront moins de temps car seules les modifications seront déployées sur votre stack.

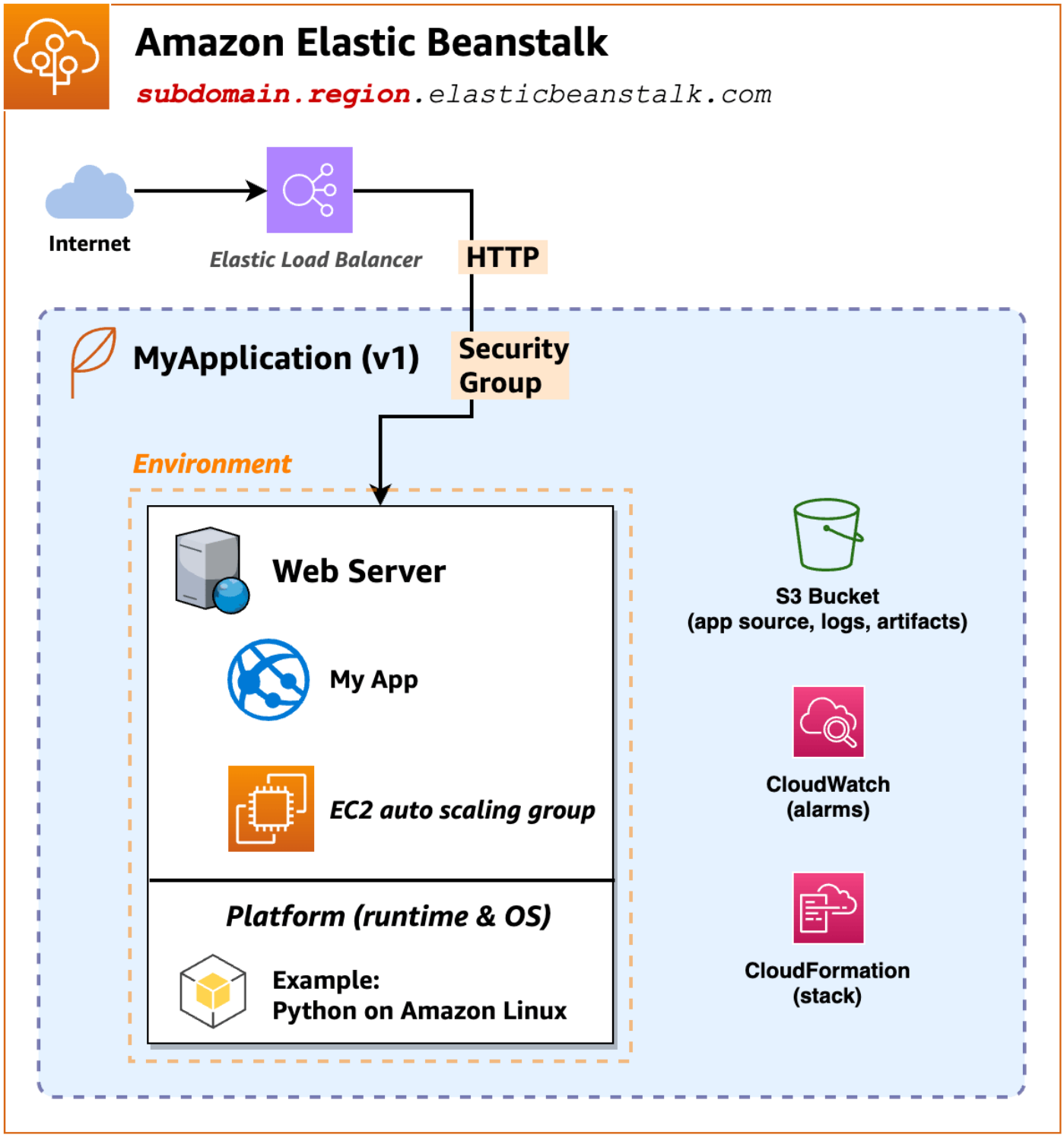


Lorsque vous créez l'exemple d'application, Elastic Beanstalk crée les ressources suivantes :

- Instance EC2 : machine virtuelle Amazon EC2 configurée pour exécuter des applications Web sur la plate-forme que vous avez sélectionnée.

  Chaque plate-forme exécute un ensemble différent de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version linguistique, un framework, un conteneur Web ou une combinaison des deux. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse pour transférer le trafic Web vers votre application Web, diffuser des ressources statiques et générer des journaux d'accès et d'erreurs. Vous pouvez vous connecter à vos instances Amazon EC2 pour consulter la configuration et les journaux.
- Groupe de sécurité d'instance : un groupe de sécurité Amazon EC2 sera créé pour autoriser les demandes entrantes sur le port 80, afin que le trafic entrant sur un équilibreur de charge puisse atteindre votre application Web.
- Compartiment Amazon S3 : emplacement de stockage pour votre code source, vos journaux et autres artefacts.
- CloudWatch Alarmes Amazon — Deux CloudWatch alarmes sont créées pour surveiller la charge sur vos instances et les augmenter ou les réduire selon les besoins.
- CloudFormation stack : Elastic CloudFormation Beanstalk utilise pour déployer les ressources dans votre environnement et apporter des modifications à la configuration. Vous pouvez consulter le modèle de définition de ressource dans la CloudFormation console.
- Nom de domaine — Un nom de domaine qui mène à votre application Web sous la forme :*subdomain*.*region*.elasticbeanstalk.com.

Elastic Beanstalk crée votre application, lance un environnement, crée une version de l'application, puis déploie votre code dans l'environnement. Au cours du processus, la console suit la progression et affiche le statut de l'événement dans l'onglet Événements.



Une fois que toutes les ressources auront été déployées, l'état de santé de l'environnement devrait passer à**0k**.

> Votre candidature est prête !
>
> Une fois que l'état de santé de votre application est **0k** passé à, vous pouvez accéder au site Web de votre application Web.

Étape 3 - Explorez l'environnement Elastic Beanstalk

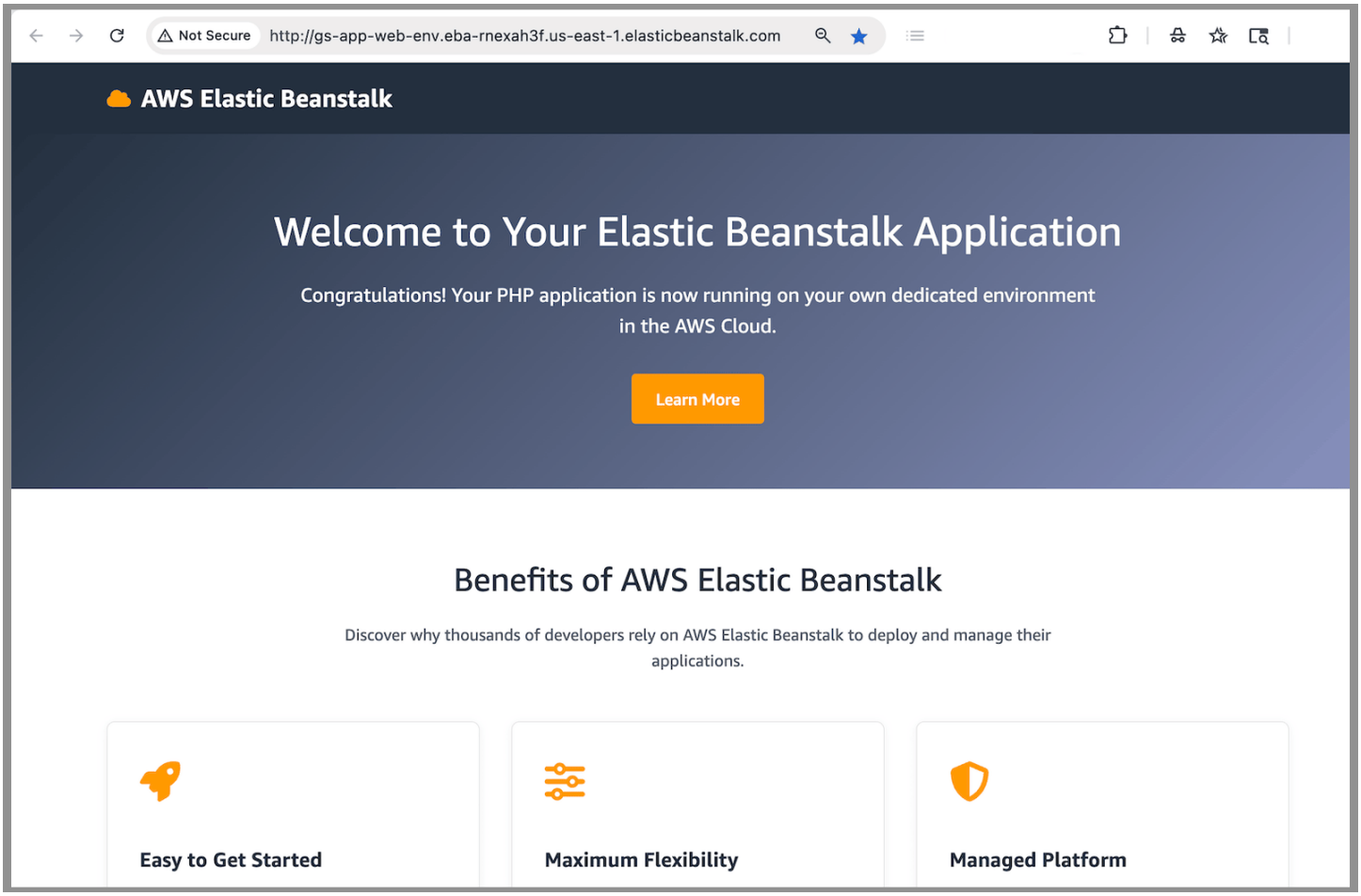
Vous allez commencer à explorer votre environnement applicatif déployé à partir de la page de présentation de l'environnement de la console.

Pour consulter l'environnement et votre application

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Accéder à l'environnement pour parcourir votre application !

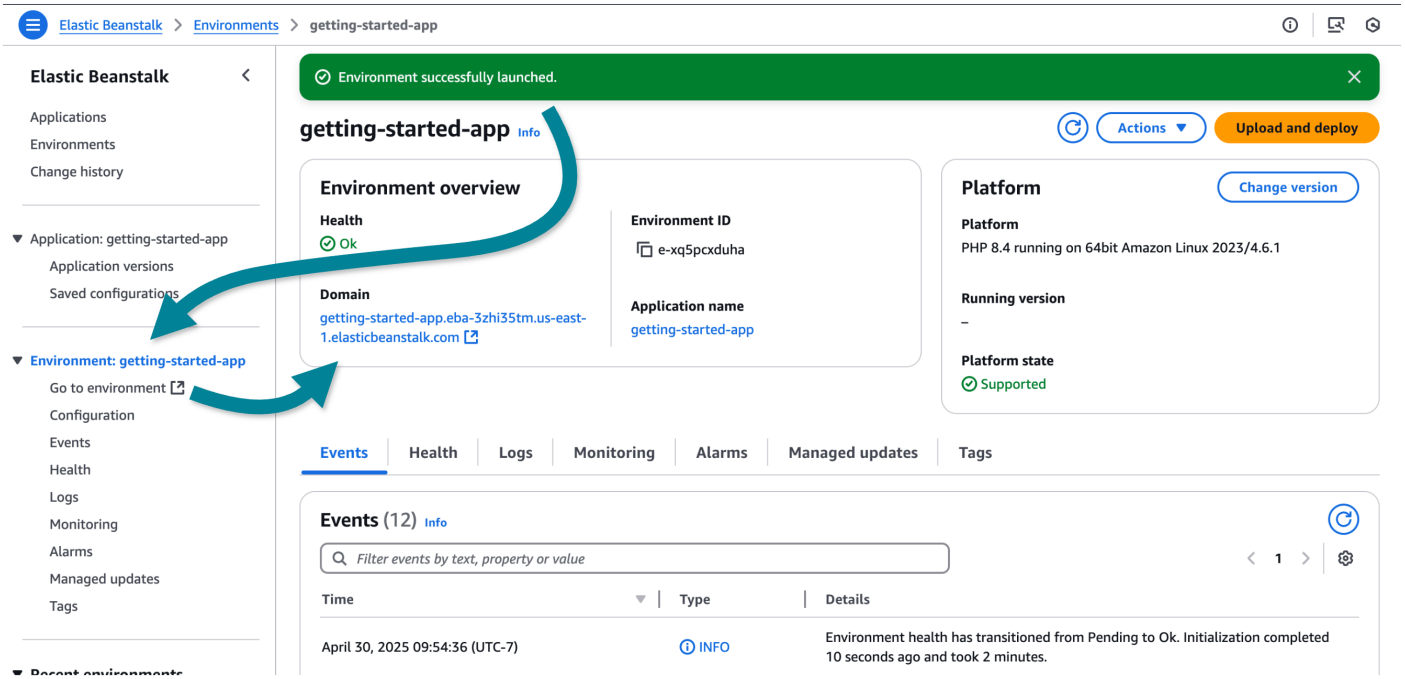
   (Vous pouvez également choisir le lien URL répertorié pour le domaine afin de parcourir votre application.)

   La connexion sera HTTP (et non HTTPS), il est donc possible qu'un avertissement apparaisse dans votre navigateur.



De retour dans la console Elastic Beanstalk, la partie supérieure présente la vue d'ensemble de l'environnement avec des informations de haut niveau sur votre environnement, notamment le nom, l'URL du domaine, l'état de santé actuel, la version en cours d'exécution et la plate-forme sur laquelle l'application s'exécute. La version et la plate-forme en cours d'exécution sont essentielles pour le dépannage de votre application actuellement déployée.

Après le volet de présentation, vous verrez l'activité récente de l'environnement dans l'onglet Événements.



Pendant qu'Elastic Beanstalk AWS crée vos ressources et lance votre application, l'environnement est dans un état normal. `Pending` Les messages d'état relatifs aux événements de lancement sont ajoutés en permanence à la liste des événements.

Le domaine de l'environnement est l'URL de votre application Web déployée. Dans le volet de navigation de gauche, l'option Accéder à l'environnement vous permet également d'accéder à votre domaine. De même, le volet de navigation de gauche contient des liens qui correspondent aux différents onglets.

Prenez note du lien Configuration dans le volet de navigation de gauche, qui affiche un résumé des valeurs des options de configuration de l'environnement, regroupées par catégorie.

> **Paramètres de configuration de l'environnement**
>
> Prenez note du lien Configuration dans le volet de navigation de gauche. Vous pouvez consulter et modifier les paramètres d'environnement détaillés, tels que les rôles de service, le réseau, la base de données, le dimensionnement, les mises à jour de plateformes gérées, la mémoire, le suivi de l'état de santé, le déploiement progressif, la journalisation, etc.

Les différents onglets contiennent des informations détaillées sur votre environnement :

- Événements : consultez une liste actualisée d'informations et de messages d'erreur provenant du service Elastic Beanstalk et d'autres services relatifs aux ressources de votre environnement.
- Santé : consultez l'état et les informations de santé détaillées des instances Amazon EC2 exécutant votre application.
- Logs(Journaux) : récupérez et téléchargez les journaux depuis Amazon EC2 dans votre environnement. Vous pouvez récupérer les journaux complets ou les activités récentes. Les journaux récupérés sont disponibles pendant 15 minutes.
- Surveillance : consultez les statistiques relatives à l'environnement, telles que la latence moyenne et l'utilisation du processeur.
- Alarmes : affichez et modifiez les alarmes configurées pour les métriques environnementales.
- Mises à jour gérées : consultez les informations sur les mises à jour de plateforme gérées à venir et achevées et sur le remplacement des instances.
- Balises : affichez et modifiez les paires clé-valeur appliquées à votre environnement.

> **Note**
>
> Les liens du volet de navigation de la console afficheront l'onglet correspondant.

Résolution des problèmes liés aux journaux

Pour résoudre les problèmes liés à des comportements inattendus ou déboguer des déploiements, vous pouvez consulter les journaux de vos environnements.

Vous pouvez demander 100 lignes de tous les fichiers journaux sous l'onglet Logs de la console Elastic Beanstalk. Vous pouvez également vous connecter directement à l'instance Amazon EC2 et suivre les journaux en temps réel.

Pour demander les journaux (console Elastic Beanstalk)

1. Accédez à votre environnement dans la console Elastic Beanstalk.
2. Choisissez l'onglet Logs ou le menu de navigation gauche, puis choisissez Request logs.
3. Sélectionnez les 100 dernières lignes.
4. Une fois les journaux créés, cliquez sur le lien Télécharger pour afficher les journaux dans le navigateur.

Dans les journaux, recherchez le journal et notez le répertoire du journal d'accès nginx.

Ajouter une politique pour activer les connexions à Amazon EC2

Avant de pouvoir vous connecter, vous devez ajouter une politique qui autorise les connexions à Amazon EC2 avec le gestionnaire de session.

1. Accédez à la Console IAM.
2. Recherchez et sélectionnez le **`aws-elasticbeanstalk-ec2-role`** rôle.
3. Choisissez Ajouter une autorisation, puis Joindre des politiques.
4. Recherchez une politique par défaut qui commence par le texte suivant :**`AmazonSSMManagedEC2Instance`**, puis ajoutez-la au rôle.

Pour vous connecter à votre Amazon EC2 à l'aide du gestionnaire de session

1. Accédez à la console Amazon EC2.
2. Choisissez Instances, puis sélectionnez votre `gs-app-web-env` instance.
3. Choisissez Connect, puis Gestionnaire de session.
4. Choisissez Se connecter.

Une fois connecté à l'instance, lancez un shell bash et suivez les logs :

1. Exécutez la commande **`bash`**.
2. Exécutez la commande **`cd /var/log/nginx`**.
3. Exécutez la commande **`tail -f access.log`**.
4. Dans votre navigateur, accédez à l'URL du domaine de l'application. Rafraîchir

> Félicitations, vous êtes connecté !
>
> Les entrées du journal de votre instance devraient être mises à jour chaque fois que vous actualisez la page.

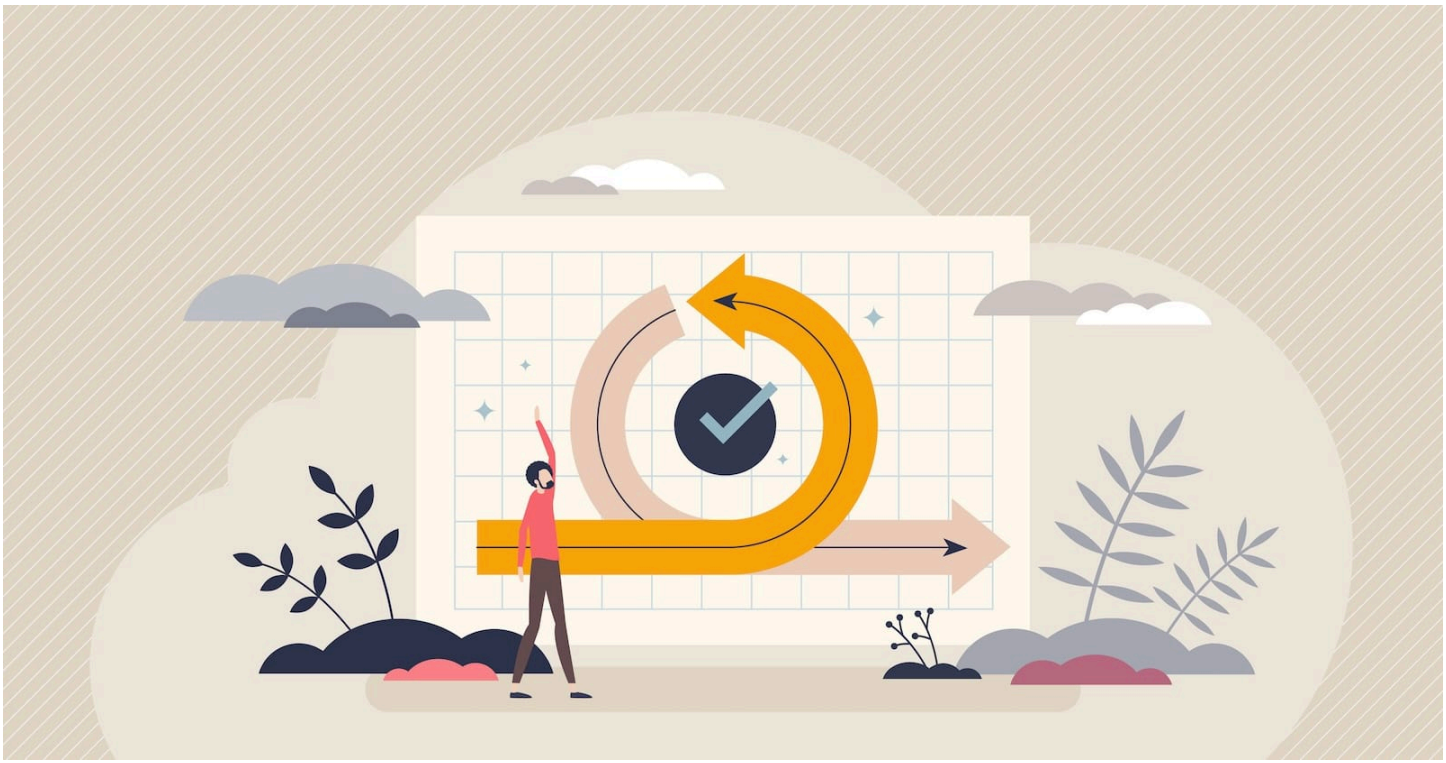
⚠ Le bouton Connect ne fonctionne pas ?

Si le bouton de connexion n'est pas disponible, retournez dans IAM et vérifiez que vous avez ajouté la politique nécessaire au rôle.

Étape 4 - Mettez à jour votre application

À terme, vous souhaiterez mettre à jour votre application. Vous pouvez déployer une nouvelle version à tout moment, tant qu'aucune autre opération de mise à jour n'est en cours sur votre environnement.

La version de l'application avec laquelle vous avez démarré ce didacticiel est nommée Exemple d'application.



Pour mettre à jour la version de votre application

1. Téléchargez l'exemple d'application PHP suivant :

   PHP — php-v2.zip

2. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS

3. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
4. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
5. Sélectionnez Choisir un fichier, puis chargez l'exemple de bundle de sources d'applications que vous avez téléchargé.

   La console remplit automatiquement l'étiquette Version avec une nouvelle étiquette unique, en incrémentant automatiquement un entier final. Si vous choisissez votre propre étiquette de version, assurez-vous qu'elle est unique.
6. Choisissez Déployer.

Pendant qu'Elastic Beanstalk déploie votre fichier sur vos instances Amazon EC2, vous pouvez consulter l'état du déploiement sur la page de présentation de l'environnement. Lorsque la version de l'application est mise à jour, l'état de santé de l'environnement est gris. Une fois le déploiement terminé, Elastic Beanstalk effectue une vérification de l'état de l'application. Lorsque l'application répond à la vérification de l'état, elle est considérée comme saine et le statut redevient vert. La présentation de l'environnement affiche la nouvelle Running version (Version en cours d'exécution), c'est-à-dire le nom que vous avez fourni dans le champ Version label (Étiquette de version).

Elastic Beanstalk télécharge également la nouvelle version de votre application et l'ajoute au tableau des versions de l'application. Pour afficher le tableau, choisissez Versions de l'application dans getting-started-apple volet de navigation.

> **Mise à jour réussie !**
>
> Vous devriez voir un message « v2 » mis à jour après avoir actualisé votre navigateur.
> Si vous souhaitez modifier la source vous-même, décompressez, modifiez, puis compressez à nouveau le bundle de sources. Sur macOS, utilisez la commande suivante depuis votre répertoire php avec le -X pour exclure les attributs de fichier supplémentaires :
> `zip -X -r ../php-v2.zip .`

Étape 5 - Faites évoluer votre application

Vous pouvez configurer votre environnement afin de mieux répondre aux besoins de votre application. Par exemple, si vous avez une application gourmande en calcul, vous pouvez modifier

le type d'instance Amazon Elastic Compute Cloud (Amazon EC2) qui exécute votre application. Pour appliquer des modifications de configuration, Elastic Beanstalk effectue une mise à jour de l'environnement.

Certains changements de configuration sont simples et rapidement effectifs. Certaines modifications nécessitent de supprimer et de recréer AWS des ressources, ce qui peut prendre plusieurs minutes. Lorsque vous modifiez les paramètres de configuration, Elastic Beanstalk vous avertit des temps d'arrêt éventuels de l'application.

Augmenter les paramètres de capacité

Dans cet exemple de modification de la configuration, vous modifiez les paramètres de capacité de votre environnement. Vous configurez un environnement évolutif à la charge équilibrée qui a entre deux et quatre instances Amazon EC2 dans son groupe Auto Scaling, puis vous vérifiez que la modification s'est produite. Elastic Beanstalk crée une instance Amazon EC2 supplémentaire, qui s'ajoute à l'instance unique qu'il a initialement créée. Elastic Beanstalk associe ensuite les deux instances à l'équilibreur de charge de l'environnement. Par conséquent, la réactivité de votre application est améliorée et sa disponibilité est augmentée.

Pour modifier la capacité de votre environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Instances (Instances), choisissez Edit (Modifier).
5. Réduisez la section Instances (Instances) pour voir plus facilement la section Capacity (Capacité). Dans la section Auto Scaling group (Groupe Auto Scaling), remplacez Environment type (Type d'environnement) par Load balanced (Équilibrage de charge).
6. Dans la ligne Instances, remplacez Min par **2** et Max par**4**.
7. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

   Si vous êtes averti que la mise à jour remplacera toutes vos instances actuelles. Choisissez Confirmer.

La mise à jour de l'environnement peut prendre quelques minutes. Vous devriez voir plusieurs mises à jour dans la liste des événements. Surveillez l'événement Déploiement réussi de la nouvelle configuration dans l'environnement.

Vérifiez l'augmentation de la capacité

Une fois la mise à jour de l'environnement terminée et l'environnement prêt, Elastic Beanstalk a automatiquement lancé une deuxième instance pour répondre à votre nouveau paramètre de capacité minimale.

Pour vérifier l'augmentation de la capacité

1. Choisissez Health dans la liste d'onglets ou dans le volet de navigation de gauche.
2. Consultez la section Amélioration de l'état de santé de l'instance.

> Vous venez de passer à la vitesse supérieure !
>
> Avec deux instances Amazon EC2, la capacité de votre environnement a doublé, et cela n'a pris que quelques minutes.

Nettoyage de votre environnement Elastic Beanstalk

Pour vous assurer que les services que vous n'utilisez pas ne vous seront pas facturés, supprimez toutes les versions de l'application et mettez fin aux environnements, ce qui supprime également les AWS ressources que l'environnement a créées pour vous.

Pour supprimer l'application et toutes les ressources associées

1. Supprimez toutes les versions de l'application.

    a. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS

    b. Dans le volet de navigation, choisissez Applications, puis choisissez getting-started-app.

    c. Dans le volet de navigation, recherchez le nom de votre application et choisissez Application versions (Versions d'application).

    d. Sur la page Application versions (Versions de l’application), sélectionnez toutes les versions de l’application que vous souhaitez supprimer.

e. Choisissez Actions, puis Supprimer.

f. Activez Delete versions from Amazon S3 (Supprimer les versions Amazon S3).

g. Choisissez Delete (Supprimer), puis Done (Terminé).

2. Terminer l'environnement.

   a. Dans le volet de navigation, choisissez getting-started-app, puis choisissez GettingStartedApp-env dans la liste des environnements.

   b. Choisissez Actions, puis Terminate Environment (Résilier l'environnement).

   c. Confirmez que vous souhaitez terminer GettingStartedApp-env en saisissant le nom de l'environnement, puis en choisissant Terminate.

3. Supprimez l'application getting-started-app.

   a. Dans le volet de navigation, sélectionnez getting-started-app.

   b. Choisissez Actions, puis Delete application (Supprimer l’application).

   c. Confirmez que vous souhaitez supprimer getting-started-appen saisissant le nom de l'application, puis en choisissant Supprimer.

> Félicitations !
>
> Vous avez déployé avec succès un exemple d'application AWS dans le Cloud, téléchargé une nouvelle version, modifié sa configuration pour ajouter une deuxième instance Auto Scaling et nettoyé vos AWS ressources !

Étapes suivantes

Pour savoir comment utiliser l'outil de ligne de **eb** commande pour automatiser le déploiement de votre code sur Elastic Beanstalk, nous vous conseillons de continuer avec le. the section called “QuickStart pour PHP”

Ensuite, vous voudrez peut-être examiner comment configurer une connexion HTTPS, voirthe section called “HTTPS”.

Configuration de l'interface de ligne de commande EB (EB CLI) pour gérer Elastic Beanstalk

L'EB CLI est une interface de ligne de commande qui fournit des commandes interactives pour créer, mettre à jour et surveiller des environnements dans AWS Elastic Beanstalk. Le projet open source EB CLI est sur Github : aws/aws-elastic-beanstalk-cli

Après avoir installé l'interface de ligne de commande EB et configuré un répertoire de projet, vous pouvez créer des environnements via une seule commande :

```
$ eb create my-beanstalk-environment
```

Nous vous recommandons d'effectuer l'installation à l'aide du script d'installation. Découvrez comment procéder dansthe section called “Installation de la CLI EB”.

AWS CLI Fournit un accès direct à Elastic APIs Beanstalk de bas niveau. Bien que puissant, il est également verbeux et moins préféré à l'EB CLI. Par exemple, la création d'un environnement avec le AWS CLI nécessite les séries de commandes suivantes :

```
$ aws elasticbeanstalk check-dns-availability \
    --cname-prefix my-cname
$ aws elasticbeanstalk create-application-version \
    --application-name my-application \
    --version-label v1 \
    --source-bundle S3Bucket=amzn-s3-demo-bucket,S3Key=php-proxy-sample.zip
$ aws elasticbeanstalk create-environment \
    --cname-prefix my-cname \
    --application-name my-app \
    --version-label v1 \
    --environment-name my-env \
    --solution-stack-name "64bit Amazon Linux 2023 v4.5.0 running Ruby 3.4"
```

Installation de la CLI EB avec un script de configuration (recommandé)

> **Nous recommandons le script d'installation**
>
> Nous vous recommandons d'utiliser le script d'installation pour configurer l'interface de ligne de commande EB et ses dépendances et éviter les conflits potentiels avec d'autres packages Python.

Pré-requis : Git, Python et virtualenv

Pour télécharger et utiliser le script d'installation

1. Pour cloner le référentiel.

   ```
   git clone https://github.com/aws/aws-elastic-beanstalk-cli-setup.git
   ```

2. Installez ou mettez à niveau l'interface de ligne de commande EB.

   macOS/Linux dans Bash ou Zsh

   ```
   python ./aws-elastic-beanstalk-cli-setup/scripts/ebcli_installer.py
   ```

   Fenêtre dans PowerShell ou fenêtre de commande

   ```
   python .\aws-elastic-beanstalk-cli-setup\scripts\ebcli_installer.py
   ```

3. Vérifiez que l'interface de ligne de commande EB est installée correctement.

   ```
   $ eb --version
   EB CLI 3.21.0 (Python 3.12)
   ```

Pour obtenir des instructions d'installation complètes, consultez le aws/aws-elastic-beanstalk-cli-setupréférentiel sur GitHub.

Installation manuelle de l'interface de ligne de commande EB

Vous pouvez installer l'interface de ligne de commande EB sous Linux, macOS et Windows à l'aide `pip` du gestionnaire de packages pour Python qui permet l'installation, la mise à niveau et la suppression des packages Python et de leurs dépendances.

> **Nous recommandons le script d'installation**
>
> Nous vous recommandons d'utiliser le Installation de la CLI EB pour configurer l'interface de ligne de commande EB et éviter les conflits de dépendance.

Prérequis : une version compatible de Python doit être installée. Vous pouvez le télécharger depuis la page de téléchargement de Python sur le site Web de Python.

Pour installer l'interface de ligne de commande EB (manuellement)

1. Exécutez la commande suivante.

   ```
   $ pip install awsebcli --upgrade --user
   ```

   L'option `--upgrade` demande à `pip` de mettre à niveau toutes les exigences qui sont déjà installées. L'option `--user` demande à `pip` d'installer le programme dans un sous-répertoire de votre répertoire utilisateur pour éviter la modification des bibliothèques utilisées par votre système d'exploitation.

   > **Résolution des problèmes**
   >
   > Si vous rencontrez des problèmes lorsque vous tentez d'installer l'interface de ligne de commande EB avec `pip`, vous pouvez installer l'interface de ligne de commande EB dans un environnement virtuel pour isoler l'outil et ses dépendances, ou utiliser une version de Python autre que celle que vous utilisez normalement.

2. Ajoutez le chemin d'accès au fichier exécutable à votre variable `PATH` :
   - Sous Linux et macOS :

     Linux : `~/.local/bin`

     macOS : `~/Library/Python/3.12/bin`

Pour modifier votre variable PATH (Linux, Unix ou macOS) :

a. Recherchez le script de profil de votre shell dans votre dossier utilisateur. Si vous n'êtes pas certain du shell utilisé, exécutez `echo $SHELL`.

```
$ ls -a ~
.  ..  .bash_logout  .bash_profile  .bashrc  Desktop  Documents  Downloads
```

- Bash : `.bash_profile`, `.profile` ou `.bash_login`.
- Zsh : `.zshrc`
- Tcsh : `.tcshrc`, `.cshrc` ou `.login`.

b. Ajoutez une commande d'exportation à votre script de profil. L'exemple suivant ajoute le chemin représenté par *LOCAL_PATH* à la PATH variable en cours.

```
export PATH=LOCAL_PATH:$PATH
```

c. Chargez le script de profil décrit à la première étape dans votre session actuelle. L'exemple suivant charge le script de profil représenté par*PROFILE_SCRIPT*.

```
$ source ~/PROFILE_SCRIPT
```

- Sous Windows :

  Python 3.12 — `%USERPROFILE%\AppData\Roaming\Python\Python312\Scripts`

  Versions antérieures de Python : `%USERPROFILE%\AppData\Roaming\Python\Scripts`

  Pour modifier votre variable PATH (Windows) :

  a. Appuyez sur la touche Windows et entrez **`environment variables`**.

  b. Choisissez Modifier les variables d'environnement pour votre compte.

  c. Choisissez PATH, puis Modifier.

  d. Ajoutez des chemins d'accès dans le champ Valeur de la variable, en les séparant par des points virgules. Par exemple: *`C:\item1\path`***`;C:\item2\path`**

  e. Choisissez OK deux fois pour appliquer les nouveaux paramètres.

f. Fermez toutes les fenêtres d'invite de commande en cours d'exécution, puis rouvrez une fenêtre d'invite de commande.

3. Vérifiez que l'interface de ligne de commande EB est installée correctement en exécutant eb --version.

```
$ eb --version
EB CLI 3.21.0 (Python 3.12)
```

L'interface de ligne de commande Elastic Beanstalk est mise à jour régulièrement pour ajouter une fonctionnalité prenant en charge les dernières fonctions Elastic Beanstalk. Pour effectuez une mise à jour vers la dernière version de l'interface de ligne de commande EB, exécutez à nouveau la commande d'installation.

```
$ pip install awsebcli --upgrade --user
```

Si vous devez désinstaller l'interface de ligne de commande EB, utilisez `pip uninstall`.

```
$ pip uninstall awsebcli
```

Installation de l'interface de ligne de commande EB dans un environnement virtuel

Pour éviter les conflits entre exigences des versions et autres packages `pip`, vous pouvez installer l'interface de ligne de commande EB dans un environnement virtuel.

Pour installer l'interface de ligne de commande EB dans un environnement virtuel

1. Tout d'abord, installez `virtualenv` avec`pip`.

```
$ pip install --user virtualenv
```

2. Créez un environnement virtuel.

```
$ virtualenv ~/eb-ve
```

L'option `-p` vous permet d'utiliser un exécutable Python autre que l'exécutable par défaut.

```
$ virtualenv -p /usr/bin/python3.12 ~/eb-ve
```

3. Activez l'environnement virtuel.

   Linux, Unix ou macOS

   ```
   $ source ~/eb-ve/bin/activate
   ```

   Windows

   ```
   $ %USERPROFILE%\eb-ve\Scripts\activate
   ```

4. Installation de l'interface de ligne de commande EB.

   ```
   (eb-ve)$ pip install awsebcli --upgrade
   ```

5. Vérifiez que l'interface de ligne de commande EB est installée correctement.

   ```
   $ eb --version
   EB CLI 3.23.0 (Python 3.12)
   ```

La commande `deactivate` vous permet de quitter l'environnement virtuel. Exécutez à nouveau la commande d'activation chaque fois que vous démarrez une nouvelle session.

Pour effectuer une mise à niveau vers la dernière version, exécutez à nouveau la commande d'installation.

```
(eb-ve)$ pip install awsebcli --upgrade
```

Installation de l'interface de ligne de commande EB avec Homebrew

La dernière version de l'interface de ligne de commande EB est généralement disponible sur `Homebrew` quelques jours après qu'elle s'est affichée dans `pip`.

> ⚠ Nous recommandons le script d'installation
>
> Nous vous recommandons d'utiliser le Installation de la CLI EB pour configurer l'interface de ligne de commande EB et éviter les conflits de dépendance.

Pour installer l'interface de ligne de commande EB avec **`Homebrew`**

1. Vérifiez que vous utilisez la dernière version de `Homebrew`.

```
$ brew update
```

2. Exécutez `brew install awsebcli`.

```
$ brew install awsebcli
```

3. Vérifiez que l'interface de ligne de commande EB est installée correctement.

```
$ eb --version
EB CLI 3.21.0 (Python 3.12)
```

Configuration de l'interface de ligne de commande EB

Après avoir installé l'EB CLI, vous êtes prêt à configurer le répertoire de votre projet et l'EB CLI eb init en exécutant. L'exemple suivant montre les étapes de configuration lors de la première exécution eb init dans un dossier de projet nomméeb.

Pour initialiser un projet d'interface de ligne de commande EB

1. Tout d'abord, l'interface de ligne de commande EB vous invite à sélectionner une région. Choisissez votre région préférée.

```
~/eb $ eb init
Select a default region
1) us-east-1 : US East (N. Virginia)
2) us-west-1 : US West (N. California)
3) us-west-2 : US West (Oregon)
4) eu-west-1 : Europe (Ireland)
5) eu-central-1 : Europe (Frankfurt)
```

```
6) ap-south-1 : Asia Pacific (Mumbai)
7) ap-southeast-1 : Asia Pacific (Singapore)
...
(default is 3): 3
```

2. Si vous y êtes invité, fournissez votre clé d'accès et votre clé secrète afin que l'EB CLI puisse gérer les ressources pour vous. Les clés d'accès sont créées dans la Gestion des identités et des accès AWS console. Si vous n'avez pas de clés, consultez Comment puis-je obtenir les informations d'identification de sécurité ? dans le manuel Référence générale d'Amazon Web Services.

```
You have not yet set up your credentials or your credentials are incorrect.
You must provide your credentials.
(aws-access-id): AKIAJOUAASEXAMPLE
(aws-secret-key): 5ZRIrtTM4ciIAvd4EXAMPLEDtm+PiPSzpoK
```

3. Une application Elastic Beanstalk est une ressource qui contient un ensemble de versions d'applications (source), d'environnements et de configurations enregistrées associés à une seule application Web. Chaque fois que vous déployez votre code source dans Elastic Beanstalk via l'interface de ligne de commande EB, une nouvelle version d'application est créée et ajoutée à la liste.

```
Select an application to use
1) [ Create new Application ]
(default is 1): 1
```

4. Le nom de l'application par défaut correspond au nom du dossier dans lequel vous exécutez eb init. Saisissez un nom décrivant votre projet.

```
Enter Application Name
(default is "eb"): eb
Application eb has been created.
```

5. Sélectionnez une plateforme qui correspond à l'infrastructure ou au langage dans lequel votre application web est développée. Si vous n'avez pas encore commencé à développer votre application, choisissez la plateforme qui vous intéresse. Vous saurez bientôt comment lancer un exemple d'application et vous pourrez modifier ce paramètre ultérieurement.

```
Select a platform.
1) .NET Core on Linux
2) .NET on Windows Server
```

```
3) Docker
4) Go
5) Java
6) Node.js
7) PHP         <== select platform by number
8) Packer
9) Python
10) Ruby
11) Tomcat
(make a selection):7
```

6. Sélectionnez une branche de plateforme spécifique.

```
Select a platform branch.
1) PHP 8.4 running on 64bit Amazon Linux 2023
2) PHP 8.3 running on 64bit Amazon Linux 2023
3) PHP 8.2 running on 64bit Amazon Linux 2023
4) PHP 8.1 running on 64bit Amazon Linux 2023
5) PHP 8.1 running on 64bit Amazon Linux 2
(default is 1):1
```

7. Choisissez yes (oui) pour attribuer une paire de clés SSH aux instances de votre environnement Elastic Beanstalk. Cela vous permet de vous connecter directement à des fins de dépannage.

```
Do you want to set up SSH for your instances?
(y/n): y
```

8. Choisissez une paire de clés existante ou créez-en une. Pour utiliser eb init afin de créer une nouvelle paire de clés, l'outil ssh-keygen doit être installé sur votre ordinateur local et disponible à partir de la ligne de commande. L'EB CLI enregistre EC2 pour vous la nouvelle paire de clés auprès d'Amazon et stocke la clé privée localement dans un dossier nommé `.ssh` dans votre répertoire utilisateur.

```
Select a keypair.
1) [ Create new KeyPair ]
(default is 1): 1
```

Votre interface de ligne de commande EB est maintenant configurée et prête à être utilisée.

Configuration avancée

- Utilisation d'un fichier .ebignore pour ignorer des fichiers
- Utilisation de profils nommés
- Déploiement d'un artefact à la place du dossier de projet
- Configuration des paramètres et des priorités
- Métadonnées de l’instance

Utilisation d'un fichier .ebignore pour ignorer des fichiers

Vous pouvez donner instruction à l'interface de ligne de commande EB d'ignorer certains fichiers du répertoire de votre projet en ajoutant le fichier `.ebignore` au répertoire. Ce fichier fonctionne comme un fichier `.gitignore`. Lorsque vous déployez le répertoire de votre projet dans Elastic Beanstalk et que vous créez une nouvelle version d'application, l'interface de ligne de commande EB n'inclut pas les fichiers spécifiés par `.ebignore` dans le bundle source qu'elle crée.

Si `.ebignore` n'est pas présent, mais qu'il existe `.gitignore`, l'interface de ligne de commande EB ignore les fichiers spécifiés dans `.gitignore`. Si `.ebignore` est présent, l'interface de ligne de commande EB ne lit pas `.gitignore`.

Quand `.ebignore` est présent, l'interface de ligne de commande EB n'utilise pas de commandes Git pour créer votre bundle de fichiers source. Autrement dit, l'interface de ligne de commande (CLI) EB ignore les fichiers spécifiés dans `.ebignore` et inclut tous les autres fichiers. Plus particulièrement, elle comprend des fichiers source non validés.

> **Note**
>
> Dans Windows, si vous ajoutez `.ebignore`, l'interface de ligne de commande EB suit les liens symboliques et inclut le fichier lié lors de la création d'un bundle de fichiers source. Il s'agit d'un problème connu, qui sera corrigé lors d'une future mise à jour.

Utilisation de profils nommés

Si vous stockez vos informations d'identification sous la forme d'un profil nommé dans un fichier `credentials` ou `config`, vous pouvez utiliser l'option `--profile` pour spécifier explicitement un profil. Par exemple, la commande suivante crée une nouvelle application à l'aide du profil `user2`.

```
$ eb init --profile user2
```

Vous pouvez également modifier le profil par défaut en définissant la variable d'environnement `AWS_EB_PROFILE`. Lorsque cette variable est définie, l'interface de ligne de commande EB lit les informations d'identification à partir du profil spécifié au lieu de `default` ou eb-cli.

Linux, macOS ou Unix

```
$ export AWS_EB_PROFILE=user2
```

Windows

```
> set AWS_EB_PROFILE=user2
```

Déploiement d'un artefact à la place du dossier de projet

Vous pouvez demander à l'interface de ligne de commande EB de déployer un fichier ZIP ou WAR que vous générez dans le cadre d'un processus de création distinct en ajoutant les lignes suivantes à `.elasticbeanstalk/config.yml` dans votre dossier de projet.

```
deploy:
  artifact: path/to/buildartifact.zip
```

Si vous configurez l'interface de ligne de commande EB dans votre référentiel Git, et ne validez pas l'artefact dans la source, utilisez l'option `--staged` pour déployer la version la plus récente.

```
~/eb$ eb deploy --staged
```

Configuration des paramètres et des priorités

L'EB CLI utilise une chaîne de fournisseurs pour rechercher des AWS informations d'identification à différents endroits, notamment des variables d'environnement système ou utilisateur et des fichiers AWS de configuration locaux.

L'interface de ligne de commande EB recherche les informations d'identification et les paramètres de configuration dans l'ordre suivant :

1. Options de ligne de commande – Spécifiez un profil nommé avec `--profile` pour remplacer les paramètres par défaut.
2. Variables d'environnement – `AWS_ACCESS_KEY_ID` et `AWS_SECRET_ACCESS_KEY`.

3. Le fichier AWS d'informations d'identification : situé `~/.aws/credentials` sur les systèmes Linux et OS X, ou `C:\Users\`*`USERNAME`*`\.aws\credentials` sur les systèmes Windows. Ce fichier peut contenir plusieurs profils nommés en plus d'un profil par défaut.
4. Fichier de configuration de l'AWS CLI – situé dans `~/.aws/config` sous Linux et OS X ou dans `C:\Users\`*`USERNAME`*`\.aws\config` sous Windows. Ce fichier peut contenir un profil par défaut, des profils nommés et des paramètres de AWS CLI configuration spécifiques pour chacun d'entre eux.
5. Fichier de configuration existant de l'interface de ligne de commande EB – Situé dans `~/.elasticbeanstalk/config` sous Linux et OS X ou dans `C:\Users\`*`USERNAME`*`\.elasticbeanstalk\config` sous Windows.
6. Informations d'identification du profil d'instance : ces informations d'identification peuvent être utilisées sur EC2 les instances Amazon auxquelles un rôle d'instance a été attribué et sont fournies via le service de EC2 métadonnées Amazon. Le profil d'instance doit avoir l'autorisation d'utiliser Elastic Beanstalk.

Si le fichier d'informations d'identification contient un profil nommé « eb-cli », l'interface de ligne de commande EB le privilégie par rapport au profil par défaut. Si aucun profil n'est trouvé, ou si le profil trouvé n'est pas autorisé à utiliser –, l'interface de ligne de commande EB vous invite à saisir les clés.

Métadonnées de l’instance

Pour utiliser l'EB CLI à partir d'une EC2 instance Amazon, créez un rôle ayant accès aux ressources nécessaires et attribuez ce rôle à l'instance lors de son lancement. Lancez l'instance et installez l'interface de ligne de commande EB avec `pip`.

```
~$ sudo pip install awsebcli
```

`pip` est pré-installé sur Amazon Linux.

L'interface de ligne de commande EB lit les informations d'identification à partir des métadonnées de l'instance. Pour plus d'informations, consultez la section Accorder l'accès aux AWS ressources aux applications exécutées sur des EC2 instances Amazon dans le guide de l'utilisateur IAM.

Utilisation de l'interface de ligne de commande EB avec Git

L'EB CLI fournit une intégration Git qui vous permet d'associer des branches de code à des environnements spécifiques dans Elastic Beanstalk afin d'organiser vos déploiements.

Pour installer Git et initialiser votre référentiel Git

1. Téléchargez la version la plus récente de Git sur le site http://git-scm.com.
2. Initialisez votre référentiel Git en saisissant la chaîne suivante :

   ```
   ~/eb$ git init
   ```

   L'interface de ligne de commande EB reconnaît désormais que votre application est configurée avec Git.
3. Si vous n'avez pas encore exécuté eb init, faites-le maintenant :

   ```
   ~/eb$ eb init
   ```

Associer des environnements Elastic Beanstalk à des branches Git

Vous pouvez associer un environnement différent à chaque branche de votre code. Lorsque vous procédez à l'extraction d'une branche, les modifications sont déployées pour l'environnement associé. Vous pouvez par exemple saisir les informations suivantes pour associer votre environnement de production à votre branche principale, et un environnement de développement distinct à une branche de développement :

```
~/eb$ git checkout mainline
~/eb$ eb use prod
~/eb$ git checkout develop
~/eb$ eb use dev
```

Déploiement des modifications

Par défaut, l'interface de ligne de commande EB déploie la dernière validation (commit) dans la branche actuelle, en utilisant respectivement l'ID de commit et le message comme libellé de version et comme description. Si vous souhaitez effectuer un déploiement dans votre environnement sans validation, vous pouvez utiliser l'option `--staged` afin de déployer les modifications qui ont été ajoutées à la zone de transit.

Pour déployer les modifications sans validation

1. Ajoutez les fichiers nouveaux et modifiés à la zone de transit :

```
~/eb$ git add .
```

2. Déployez les modifications indexées avec eb deploy :

```
~/eb$ eb deploy --staged
```

Si vous avez configuré l'interface de ligne de commande EB pour déployer un artefact et que vous ne le validez pas dans le référentiel git, utilisez l'option `--staged` pour déployer la dernière version.

Utilisation des sous-modules Git

Certains projets de code tirent parti des sous-modules Git, des référentiels dans le référentiel de niveau supérieur. Lorsque vous déployez votre code à l'aide de eb create ou de eb deploy, l'interface de commande EB peut inclure des sous-modules dans le fichier zip de la version de l'application et les charger avec le reste du code.

Vous pouvez contrôler l'inclusion de sous-modules à l'aide de l'option `include_git_submodules` dans la section `global` du fichier de configuration de l'interface de ligne de commande EB, le fichier `.elasticbeanstalk/config.yml` dans votre dossier de projet.

Pour inclure des sous-modules, définissez cette option sur `true` :

```
global:
  include_git_submodules: true
```

Lorsque l'option `include_git_submodules` est manquante ou définie sur `false`, l'interface de ligne de commande EB n'inclut pas de sous-modules dans le fichier zip chargé.

Consultez Git Tools - Submodules pour obtenir plus d'informations sur les sous-modules Git.

> **Comportement par défaut**
>
> Lorsque vous exécutez eb init pour configurer votre projet, l'interface de ligne de commande EB ajoute l'option `include_git_submodules` et la définit sur `true`. De cette manière, tous les sous-modules de votre projet sont inclus dans vos déploiements.
> L'interface de ligne de commande EB n'a pas toujours pris en charge l'inclusion de sous-modules. Pour éviter toute modification accidentelle ou indésirable aux projets qui existaient avant l'ajout de la prise en charge des sous-modules, l'interface de ligne de commande

EB n'inclut pas les sous-modules pour lesquels l'option `include_git_submodules` est manquante. Si vous disposez d'un tel projet et que vous souhaitez inclure des sous-modules dans vos déploiements, ajoutez l'option et définissez-la sur `true`, comme expliqué dans cette section.

CodeCommit comportement

L'intégration d'Elastic Beanstalk avec CodeCommit ne prend pas en charge les sous-modules pour le moment. Si vous avez activé l'intégration de votre environnement CodeCommit, les sous-modules ne sont pas inclus dans vos déploiements.

Affectation de balises Git à votre version de l'application

Vous pouvez utiliser une balise Git comme libellé de version afin d'identifier la version d'application en cours d'exécution dans votre environnement. Par exemple, saisissez la chaîne suivante :

```
~/eb$ git tag -a v1.0 -m "My version 1.0"
```

Guide de référence des commandes de l'interface de ligne de commande (CLI) EB

Vous pouvez utiliser l'interface de ligne de commande Elastic Beanstalk (EB CLI) pour effectuer diverses opérations de déploiement et de gestion de vos applications et environnements Elastic Beanstalk. L'interface de ligne de commande (CLI) EB s'intègre à Git si vous souhaitez déployer le code source de l'application qui est sous le contrôle source Git. Pour de plus amples informations, veuillez consulter Configuration de l'interface de ligne de commande EB (EB CLI) pour gérer Elastic Beanstalk et Utilisation de l'interface de ligne de commande EB avec Git.

Commandes

- Options courantes
- eb abort
- eb appversion
- eb clone
- eb codesource
- eb config
- eb console
- eb create
- eb deploy
- eb events
- eb health
- eb init
- eb labs
- eb list
- eb local
- eb logs
- eb migrate
- eb open
- eb platform

- eb printenv
- eb restore
- eb scale
- eb setenv
- eb ssh
- eb status
- eb swap
- eb tags
- eb terminate
- eb upgrade
- eb use

Options courantes

Vous pouvez utiliser les options suivantes avec toutes les commandes de l'interface de ligne de commande (CLI) EB.

Nom	Description
`--debug`	Imprimez des informations pour le débogage.
`-h`, `--help`	Affichez le message d'aide. Type : chaîne Par défaut : aucun
`--no-verify-ssl`	Ignorez la vérification du certificat SSL. Utilisez cette option si vous avez des problèmes pour utiliser l'interface de ligne de commande (CLI) avec un proxy.
`--profile`	Utilisez un profil spécifique de votre fichier AWS d'informations d'identification.
`--quiet`	Supprimez toutes les données de la commande.

Nom	Description
`--region`	Utilisez la région spécifiée.
`-v`, `--verbose`	Affichez des informations détaillées.

eb abort

Description

Annule une mise à niveau lorsque les modifications de la configuration de l'environnement apportées à des instances sont toujours en cours.

> **Note**
>
> Si vous avez plus de deux environnements qui font l'objet d'une mise à jour, vous êtes invité à sélectionner le nom de l'environnement pour lequel vous souhaitez annuler les modifications.

Syntaxe

eb abort

eb abort ***environment-name***

Options

Nom	Description
Options courantes	

Output

La commande affiche une liste d'environnements actuellement mis à jour et vous invite à choisir la mise à jour que vous souhaitez annuler. Si un seul environnement est actuellement en cours de

mise à jour, vous n'avez pas besoin de spécifier le nom de l'environnement. En cas de réussite, la commande annule les changements de configuration d'environnement. Le processus de restauration continue jusqu'à ce que toutes les instances de l'environnement aient la configuration de l'environnement précédent ou jusqu'à ce que le processus de restauration échoue.

Exemple

L'exemple suivant annule la mise à niveau de la plateforme.

```
$ eb abort
Aborting update to environment "tmp-dev".
<list of events>
```

eb appversion

Description

La commande `appversion` de l'interface de ligne de commande (CLI) EB gère les versions de votre application Elastic Beanstalk. Vous pouvez créer une nouvelle version de l'application sans déployer, supprimer une version de l'application ou créer la stratégie de cycle de vie de la version de l'application. Si vous appelez la commande sans ajouter d'options, elle passe en mode interactif.

Utilisez l'option `--create` pour créer une nouvelle version de l'application.

Utilisez l'option `--delete` pour supprimer une version de l'application.

Utilisez l'option `lifecycle` pour afficher ou créer la stratégie de cycle de vie d'une version de l'application. Pour de plus amples informations, veuillez consulter the section called “Cycle de vie des versions”.

Syntaxe

eb appversion

eb appversion [-c | --create]

eb appversion [-d | --delete] ***version-label***

eb appversion lifecycle [-p | --print]

Options

Nom	Description Type : Chaîne
-a *application-name* or --nom_application *application-name*	Nom de l'application . Si une application portant le nom spécifié n'est pas trouvée, l'interface de ligne de commande (CLI) EB crée une version d'application pour une nouvelle application. Applicable uniquement avec l'option `--create`. Type : chaîne
-c ou --create	Créez une nouvelle version de l'application.
-d *version-label* or --supprimer *version-label*	Supprimez la version de l'application étiquetée*version-label* .
`-l` *version_label* or `--label` *version_label*	Spécifiez une étiquette à utiliser pour la version créée par l'interface de ligne de commande (CLI) EB. Si vous n'utilisez pas cette option, l'interface de ligne de commande (CLI) EB génère une nouvelle étiquette unique. Si vous fournissez une étiquette de version, assurez-vous qu'elle est unique. Applicable uniquement avec l'option `--create`. Type : chaîne
lifecycle	Appeler l'éditeur par défaut pour créer une stratégie de cycle de vie d'une version de l'application. Utilisez cette stratégie pour éviter d'atteindre le quota de version de l'application.

Nom	Description Type : Chaîne
lifecycle -p ou lifecycle --print	Afficher la stratégie de cycle de vie actuelle de l'application.

`-m "`*`version_description`*` "`
 	
or
 	
`--message "`*`version_description`*`  "`
	Description de la version de l'application. Elle est entourée de guillemets doubles.
	
	Applicable uniquement avec l'option `--create`.
	
	Type : chaîne

`-p`
 	
ou
 	
`--process`
	Prétraitez et validez les fichiers de configuration et le manifeste d'environnement dans le groupe source. La validation des fichiers de configuration peut identifier des problèmes. Nous vous recommandons de le faire avant de déployer la version de l'application dans l'environnement.
	
	Applicable uniquement avec l'option `--create`.

`--source codecommit/ `*`repository-name`*`/`*`branch-name`*	CodeCommit référentiel et branche. Applicable uniquement avec l'option `--create`.
`--staged`	Utilisez les fichiers mis en scène dans l'index git, au lieu du commit HEAD, pour créer la version de l'application. Applicable uniquement avec l'option `--create`.
`--timeout `*`minutes`*	Le nombre de minutes avant que la commande expire. Applicable uniquement avec l'option `--create`.
Options courantes	

Utilisation interactive de la commande

Si vous utilisez la commande sans arguments, la sortie affiche les versions de l'application. Elles sont répertoriées par ordre chronologique inverse, la dernière version étant répertoriée en premier. Consultez la section Examples (Exemples) pour obtenir des exemples de ce à quoi ressemble l'écran. Notez que la ligne d'état s'affiche en bas. La ligne d'état affiche des informations contextuelles.

Appuyez sur d pour supprimer une version de l'application, appuyez sur `l` pour gérer la stratégie de cycle de vie de votre application, ou appuyez sur q pour quitter sans apporter de modifications.

> **Note**
>
> Si la version est déployée sur un environnement, vous ne pouvez pas la supprimer.

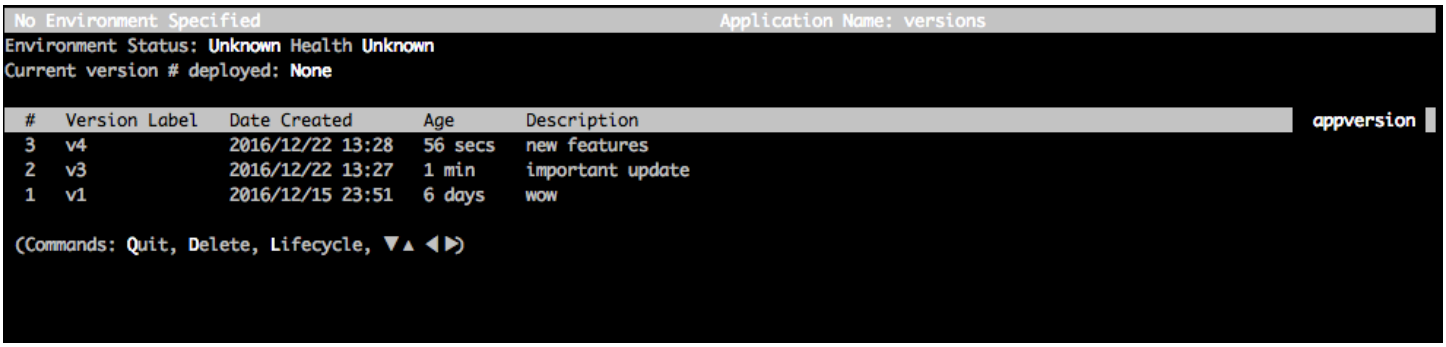
Output

La commande avec l'option `--create` affiche un message qui confirme que la version d'application a été créée.

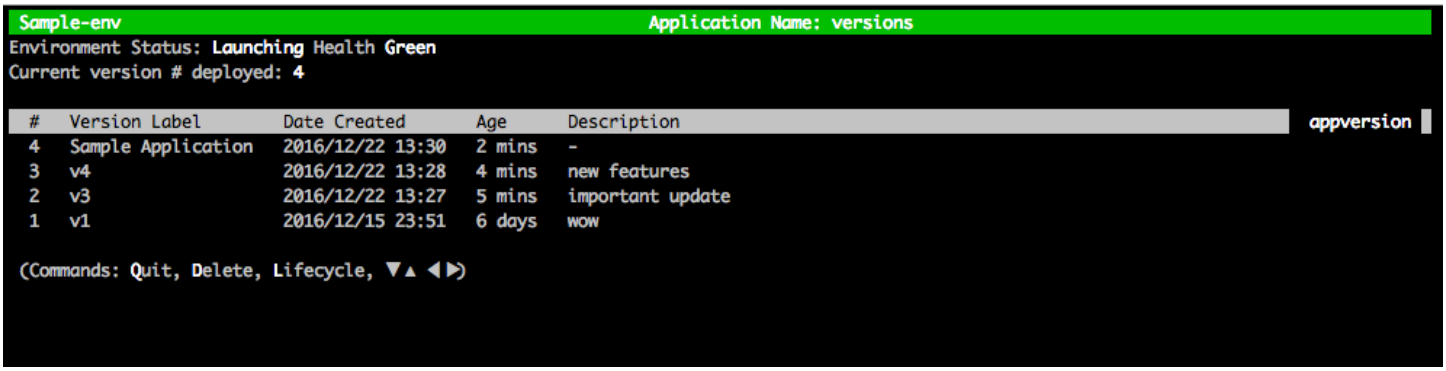
La commande avec l'`--delete`*`version-label`*option affiche un message confirmant que la version de l'application a été supprimée.

Exemples

L'exemple suivant montre la fenêtre interactive pour une application sans déploiement.



L'exemple suivant montre la fenêtre interactive pour une application avec la quatrième version, avec l'étiquette de version Sample Application (Exemple d'application), déployée.



L'exemple suivant montre le résultat d'une eb appversion lifecycle -p commande, où se *ACCOUNT-ID* trouve l'ID de compte de l'utilisateur :

```
Application details for: lifecycle
  Region: sa-east-1
  Description: Application created from the EB CLI using "eb init"
  Date Created: 2016/12/20 02:48 UTC
  Date Updated: 2016/12/20 02:48 UTC
  Application Versions: ['Sample Application']
  Resource Lifecycle Config(s):
    VersionLifecycleConfig:
      MaxCountRule:
        DeleteSourceFromS3: False
        Enabled: False
        MaxCount: 200
      MaxAgeRule:
        DeleteSourceFromS3: False
        Enabled: False
        MaxAgeInDays: 180
    ServiceRole: arn:aws:iam::ACCOUNT-ID:role/aws-elasticbeanstalk-service-role
```

eb clone

Description

Clone un environnement dans un nouvel environnement afin que les deux aient des paramètres d'environnement identiques.

> **Note**
>
> Par défaut, quelle que soit la version de pile de solutions de l'environnement à partir de laquelle vous créez le clone, la commande eb clone crée l'environnement clone avec la pile de solutions plus récente. Vous pouvez supprimer cela en incluant l'option `--exact` lorsque vous exécutez la commande.

> **Important**
>
> Les environnements Elastic Beanstalk clonés ne transfèrent pas les groupes de sécurité à des fins d'entrée, laissant l'environnement ouvert à tout le trafic Internet. Vous devrez rétablir les groupes de sécurité d'entrée pour l'environnement cloné.
> Vous pouvez voir les ressources qui ne peuvent pas être clonées en vérifiant l'état de dérive de la configuration de votre environnement. Pour plus d'informations, consultez la section Détecter la dérive sur une CloudFormation pile entière dans le Guide de AWS CloudFormation l'utilisateur.

Syntaxe

eb clone

eb clone ***environment-name***

Options

Nom	Description
`-n` *`string`* or `--clone_name` *`string`*	Nom souhaité pour l'environnement cloné.
`-c` *`string`* or	Préfixe CNAME souhaité pour l'environnement cloné.

Nom	Description
`--cname` *`string`*	
`--envvars`	Propriétés de l'environnement dans une liste séparée par des virgules au format *`name`* =. *`value`* Type : Chaîne Contraintes : • Les paires clé-valeur doivent être séparées par des virgules. • Les clés et les valeurs peuvent contenir tout caractère alphabétique dans n'importe quelle langue, tout caractère numérique, espace blanc, séparateur invisible et les symboles suivants : _ . : / + \ - @ • Les clés peuvent contenir jusqu'à 128 caractères. Les valeurs peuvent contenir jusqu'à 256 caractères. • Les clés et les valeurs sont sensibles à la casse. • Les valeurs ne peuvent pas correspondre au nom de l'environnement. • Les valeurs ne peuvent pas inclure `aws:` ni `elasticbeanstalk:` . • La taille totale de toutes les propriétés de l'environnement ne peut pas dépasser 4 096 octets.
`--exact`	Empêche Elastic Beanstalk de mettre à jour la version de pile de solutions pour le nouvel environnement clone vers la version la plus récente disponible (pour la plateforme de l'environnement d'origine).
`--scale` *`number`*	Le nombre d'instances à exécuter dans l'environnement clone au moment du lancement.
`--tags` *`name`*=*`value`*	Balises pour les ressources de votre environnement dans une liste séparée par des virgules au format *`name`* =. *`value`*
`--timeout`	Le nombre de minutes avant que la commande expire.

Nom	Description
Options courantes	

Output

En cas d'aboutissement, la commande crée un environnement qui a les mêmes paramètres que l'environnement d'origine ou avec des modifications à l'environnement comme indiqué par toutes les options eb clone.

Exemple

L'exemple suivant clone l'environnement spécifié.

```
$ eb clone
Enter name for Environment Clone
(default is tmp-dev-clone):
Enter DNS CNAME prefix
(default is tmp-dev-clone):
Environment details for: tmp-dev-clone
  Application name: tmp
  Region: us-west-2
  Deployed Version: app-141029_144740
  Environment ID: e-vjvrqnn5pv
  Platform: 64bit Amazon Linux 2014.09 v1.0.9 running PHP 5.5
  Tier: WebServer-Standard-1.0
  CNAME: tmp-dev-clone.elasticbeanstalk.com
  Updated: 2014-10-29 22:00:23.008000+00:00
Printing Status:
2018-07-11 21:04:20    INFO: createEnvironment is starting.
2018-07-11 21:04:21    INFO: Using elasticbeanstalk-us-west-2-888888888888 as Amazon S3
 storage bucket for environment data.
...
2018-07-11 21:07:10    INFO: Successfully launched environment: tmp-dev-clone
```

eb codesource

Description

Configure l'interface de ligne de commande EB pour le déployer à partir d'un CodeCommit référentiel, ou désactive CodeCommit l'intégration et télécharge le bundle source depuis votre machine locale.

> **Note**
>
> Certaines AWS régions n'en proposent pas CodeCommit. L'intégration entre Elastic CodeCommit Beanstalk et Elastic ne fonctionne pas dans ces régions.
> Pour plus d'informations sur les AWS services offerts dans chaque région, consultez le tableau des régions.

Syntaxe

eb codesource

eb codesource codecommit

eb codesource local

Options

Nom	Description
Options courantes	

Output

eb codesourcevous invite à choisir entre l' CodeCommit intégration et les déploiements standard.

eb codesource codecommit démarre la configuration interactive du référentiel pour l'intégration CodeCommit .

eb codesource local affiche la configuration d'origine et désactive l'intégration CodeCommit.

Exemples

eb codesource codecommitÀ utiliser pour configurer CodeCommit l'intégration pour la branche actuelle.

```
~/my-app$ eb codesource codecommit
Select a repository
1) my-repo
2) my-app
3) [ Create new Repository ]
(default is 1): 1

Select a branch
1) mainline
2) test
3) [ Create new Branch with local HEAD ]
(default is 1): 1
```

Permet eb codesource local de désactiver CodeCommit l'intégration pour la branche actuelle.

```
~/my-app$ eb codesource local
Current CodeCommit setup:
  Repository: my-app
  Branch: mainline
Default set to use local sources
```

eb config

Description

Gère les paramètres de configuration actifs et les configurations enregistrées de votre environnement. Vous pouvez utiliser cette commande pour charger, télécharger ou répertorier les configurations enregistrées de votre environnement. Vous pouvez également l'utiliser pour télécharger, afficher ou mettre à jour ses paramètres de configuration actifs.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande modifie également les paramètres de configuration du générateur. Cela se fait en fonction des valeurs définies dans `platform.yaml`.

> **Note**
>
> eb config n'affiche pas les propriétés de l'environnement. Pour définir les propriétés de l'environnement que vous pouvez lire à partir de votre application, utilisez eb setenv à la place.

Syntaxe

Les éléments suivants font partie de la syntaxe utilisée pour que la commande eb config fonctionne avec les paramètres de configuration actifs de votre environnement. Pour obtenir des exemples spécifiques, consultez la section Exemples plus loin dans cette rubrique.

- eb config– Affiche les paramètres de configuration actifs de votre environnement dans un éditeur de texte que vous avez configuré en tant que variable d'environnement EDITOR. Lorsque vous enregistrez des modifications dans le fichier et que vous fermez l'éditeur, l'environnement est mis à jour avec les paramètres d'option que vous avez enregistrés dans le fichier.

  > **Note**
  >
  > Si vous n'avez pas configuré de variable d'environnement EDITOR, l'interface de ligne de commande (CLI) EB affiche vos paramètres d'options dans votre éditeur par défaut pour les fichiers YAML.

- eb config ***environment-name*** – Affiche et met à jour la configuration de l'environnement nommé. La configuration s'affiche soit dans un éditeur de texte que vous avez configuré, soit dans les fichiers YAML de votre éditeur par défaut.
- eb config save – Enregistre les paramètres de configuration actifs de l'environnement actuel dans `.elasticbeanstalk/saved_configs/` avec le nom de fichier `[configuration-name].cfg.yml`. Par défaut, l'interface de ligne de commande EB enregistre les paramètres de configuration *`configuration-name`* en fonction du nom de l'environnement. Vous pouvez spécifier un nom de configuration différent en incluant l'option `--cfg` avec votre nom de la configuration souhaité lorsque vous exécutez la commande.

  Vous pouvez baliser votre configuration enregistrée à l'aide de l'option `--tags`.
- eb config **`--display`** – Écrit les paramètres de configuration actifs d'un environnement dans stdout au lieu d'un fichier. Par défaut, cela affiche les paramètres de configuration dans le terminal.

- eb config **--update** *configuration_string* **|** *file_path*— Met à jour les paramètres de configuration actifs de l'environnement actuel avec les informations spécifiées dans *configuration_string* ou dans le fichier identifié par*file_path*.

> **Note**
>
> Les options `--display` et `--update` offrent une flexibilité pour lire et réviser les paramètres de configuration d'un environnement par programmation.

Ce qui suit décrit la syntaxe pour l'utilisation de la commande eb config permettant de travailler avec des configurations enregistrées. Pour obtenir des exemples, consultez la section Exemples plus loin dans cette rubrique.

- eb config get ***config-name***– Télécharge la configuration enregistrée nommée depuis Amazon S3.
- eb config delete ***config-name*** – Supprime la configuration enregistrée nommée d'Amazon S3. Elle la supprime également localement, si vous l'avez déjà téléchargée.
- eb config list – Répertorie les configurations enregistrées que vous avez dans Amazon S3.
- eb config put ***filename*** – Télécharge la nommé configuration enregistrée nommée dans un compartiment Amazon S3. Ils *filename* doivent avoir l'extension de fichier `.cfg.yml`. Pour spécifier le nom du fichier sans chemin d'accès, vous pouvez enregistrer le fichier dans le dossier `.elasticbeanstalk` ou dans le dossier `.elasticbeanstalk/saved_configs/` avant d'exécuter la commande. Vous pouvez également le spécifier *filename* en fournissant le chemin complet.

Options

Nom	Description
`--cfg` *config-name*	Nom à utiliser pour une configuration enregistrée. Cette option fonctionne avec eb config save uniquement.
`-d` ou	Affiche les paramètres de configuration de l'environnement actuel (écrit dans stdout).

Nom	Description
`--display`	Utilisez l'option `--format` pour spécifier que la sortie doit être au format JSON ou YAML. Si vous ne le spécifiez pas, la sortie est au format YAML. Cette option ne fonctionne que si vous utilisez la commande eb config sans aucune autre sous-commande.
`-f` *`format_type`* ou `--format` *`format_type`*	Spécifie le format d'affichage. Les valeurs valides sont JSON ou YAML. La valeur par défaut est YAML. Cette option fonctionne uniquement avec l'option `--display` .
`--tags` *`key1`*`=`*`value1`*`[,`*`ke`	Les balises à ajouter à votre configuration enregistrée. Lorsque vous spécifiez des balises dans la liste, spécifiez-les en tant que paires clé=valeur et séparez-les par une virgule. Pour plus d'informations, consultez Balisage de configurations enregistrées. Cette option fonctionne avec eb config save uniquement.
`--timeout` *`timeout`*	Le nombre de minutes avant que la commande expire.

Nom	Description

`-u configuration_string | file_path`
 	
ou
 	
`--update configuration_string | file_path`
	Met à jour les paramètres de configuration actifs de l'environnement actuel.
	
	Cette option ne fonctionne que si vous utilisez la commande eb config sans aucune autre sous-commande.
	
	Le paramètre *configuration_string | file_path* est de type chaîne. La chaîne fournit la liste des espaces de noms et des options correspondantes pour ajouter, mettre à jour ou supprimer des paramètres de configuration de votre environnement. La chaîne d'entrée peut également représenter un fichier contenant les mêmes informations.
	
	Pour spécifier un nom de fichier, la chaîne d'entrée doit suivre le format "file://< *path*><*filename*>". Pour spécifier le nom de fichier sans un*path*, enregistrez-le dans le dossier dans lequel vous exécutez la commande. Vous pouvez sinon spécifier le nom de fichier en fournissant le chemin d'accès complet.
	
	Les informations de configuration doivent répondre aux conditions suivantes. Au moins une des sections,OptionSettings ou OptionsToRemove, est requise. Utilisez OptionSettings pour ajouter ou modifier des options. Utilisez OptionsToRemove pour supprimer des options d'un espace de noms. Pour obtenir des exemples spécifiques, consultez la section Exemples plus loin dans cette rubrique.
	
	Example
	
	Format YAML
	
	`OptionSettings:`
	`  namespace1:`
	`    option-name-1: option-value-1`
	`    option-name-2: option-value-2`
	`    ...`
	`OptionsToRemove:`
	`  namespace1:`

Nom	Description

	option-name-1
	option-name-2
	...
	
	Example
	
	Format JSON
	
	{
	"OptionSettings": {
	"namespace1": {
	"option-name-1": " *option-value-1* ",
	"option-name-2": " *option-value-2* ",
	...
	}
	},
	"OptionsToRemove": {
	"namespace1": {
	"option-name-1",
	"option-name-2",
	...
	}
	}
	}

Options courantes	

Output

Si la commande eb config ou eb config ***environment-name***est exécutée avec succès sans ajout d'une sous-commande ou d'une option, la commande affiche vos paramètres d'options actuels dans l'éditeur de texte que vous avez configuré en tant que variable d'environnement EDITOR. Si vous n'avez pas configuré de variable d'environnement EDITOR, l'interface de ligne de commande (CLI) EB affiche vos paramètres d'options dans votre éditeur par défaut pour les fichiers YAML.

Lorsque vous enregistrez des modifications dans le fichier et que vous fermez l'éditeur, l'environnement est mis à jour avec les paramètres d'option que vous avez enregistrés dans le fichier. La sortie suivante s'affiche pour confirmer la mise à jour de la configuration.

```
$ eb config myApp-dev
    Printing Status:
    2021-05-19 18:09:45    INFO    Environment update is starting.
    2021-05-19 18:09:55    INFO    Updating environment myApp-dev's configuration
 settings.
    2021-05-19 18:11:20    INFO    Successfully deployed new configuration to
 environment.
```

Si la commande s'exécute avec succès avec l'option `--display`, elle affiche les paramètres de configuration de l'environnement actuel (écrit dansstdout).

Si la commande s'exécute avec succès avec le paramètre `get`, la commande affiche l'emplacement de la copie locale que vous avez téléchargée.

Si la commande s'exécute avec succès avec le paramètre `save`, la commande affiche l'emplacement du fichier enregistré.

Exemples

Cette section décrit comment modifier l'éditeur de texte que vous utilisez pour afficher et modifier votre fichier de paramètres d'option.

Pour Linux et UNIX, l'exemple suivant remplace l'éditeur par vim :

```
$ export EDITOR=vim
```

Pour Linux et UNIX, l'exemple suivant remplace l'éditeur par ce qui est installé à l'emplacement `/usr/bin/kate`.

```
$ export EDITOR=/usr/bin/kate
```

Pour Windows, l'exemple suivant remplace l'éditeur par Notepad++.

```
> set EDITOR="C:\Program Files\Notepad++\Notepad++.exe
```

Cette section fournit des exemples de la commande eb config lorsqu'elle est exécutée avec des sous-commandes.

L'exemple suivant supprime la configuration enregistrée nommée `app-tmp`.

```
$ eb config delete app-tmp
```

L'exemple suivant télécharge la configuration enregistrée avec le nom app-tmp depuis votre compartiment Amazon S3.

```
$ eb config get app-tmp
```

L'exemple suivant répertorie les noms des configurations enregistrées qui sont stockées dans votre compartiment Amazon S3.

```
$ eb config list
```

L'exemple suivant télécharge la copie locale de la configuration enregistrée nommée app-tmp dans votre compartiment Amazon S3.

```
$ eb config put app-tmp
```

L'exemple suivant enregistre les paramètres de configuration de l'environnement en cours d'exécution actuel. Si vous ne fournissez pas un nom à utiliser pour la configuration enregistrée, alors Elastic Beanstalk nomme le fichier de configuration en fonction du nom de l'environnement. Par exemple, un environnement nommé tmp-dev serait appelé `tmp-dev.cfg.yml`. Elastic Beanstalk enregistre le fichier dans le dossier `/.elasticbeanstalk/saved_configs/`.

```
$ eb config save
```

Dans l'exemple suivant, l'option `--cfg` est utilisée pour enregistrer les paramètres de configuration de l'environnement tmp-dev dans un fichier appelé `v1-app-tmp.cfg.yml`. Elastic Beanstalk enregistre le fichier dans le dossier `/.elasticbeanstalk/saved_configs/`. Si vous ne spécifiez pas un nom d'environnement, Elastic Beanstalk enregistre les paramètres de configuration de l'environnement en cours d'exécution actuel.

```
$ eb config save tmp-dev --cfg v1-app-tmp
```

Cette section fournit des exemples de commande eb config lorsqu'elle est exécutée sans sous-commande.

La commande suivante affiche les paramètres d'options de votre environnement actuel dans un éditeur de texte.

```
$ eb config
```

La commande suivante affiche les paramètres d'options de l'environnement my-env dans un éditeur de texte.

```
$ eb config my-env
```

L'exemple suivant affiche les paramètres d'options de votre environnement actuel. Il sort au format YAML car aucun format spécifique n'a été spécifié avec l'option `--format`.

```
$ eb config --display
```

L'exemple suivant met à jour les paramètres d'options de votre environnement actuel avec les spécifications du fichier nommé `example.txt`. Le fichier a un format YAML ou JSON. L'interface de ligne de commande (CLI) EB détecte automatiquement le format de fichier.

- L'option Minsize est définie sur 1 pour l'espace de noms `aws:autoscaling:asg`.
- La taille du lot de l'espace de noms `aws:elasticbeanstalk:command` est définie sur 30 %.
- Il supprime le paramètre d'option IdleTimeout: None de l'espace de nomsAWSEBV2LoadBalancer.aws:elbv2:loadbalancer.

```
$ eb config --update "file://example.txt"
```

Example- nom de fichier :example.txt - format YAML

```
OptionSettings:
  'aws:elasticbeanstalk:command':
    BatchSize: '30'
    BatchSizeType: Percentage
  'aws:autoscaling:asg':
    MinSize: '1'
OptionsToRemove:
  'AWSEBV2LoadBalancer.aws:elbv2:loadbalancer':
    IdleTimeout
```

Example- nom de fichier :example.txt - format JSON

```
{
```

```
    "OptionSettings": {
        "aws:elasticbeanstalk:command": {
            "BatchSize": "30",
            "BatchSizeType": "Percentage"
        },
        "aws:autoscaling:asg": {
            "MinSize": "1"
        }
    },
    "OptionsToRemove": {
        "AWSEBV2LoadBalancer.aws:elbv2:loadbalancer": {
            "IdleTimeout"
        }
    }
}
```

Les exemples suivants mettent à jour les paramètres d'options de votre environnement actuel. La commande définit l'option Minsize sur 1 pour l'espace de noms `aws:autoscaling:asg`.

> **Note**
>
> Ces exemples sont spécifiques à Windows PowerShell. Ils échappent aux occurrences littérales du caractère double guillemet (") en le faisant précéder d'une barre oblique (\). Différents systèmes d'exploitation et environnements de ligne de commande peuvent comporter des séquences d'échappement différentes. Pour cette raison, nous vous recommandons d'utiliser l'option de fichier reprise dans les exemples précédents. La spécification des options de configuration dans un fichier ne nécessite pas l'échappement de caractères et est cohérente entre différents systèmes d'exploitation.

L'exemple suivant est au format JSON. L'interface de ligne de commande (CLI) EB détecte si le format est en JSON ou YAML.

```
PS C:\Users\myUser\EB_apps\myApp-env>eb config --update '{\"OptionSettings\":
{\"aws:autoscaling:asg\":{\"MaxSize\":\"1\"}}}'
```

L'exemple suivant est au format YAML. Pour saisir la chaîne YAML dans le format correct, la commande inclut l'espacement et les end-of-line retours requis dans un fichier YAML.

- Terminez chaque ligne avec la touche « Entrée » ou « retour ».

- Commencez la deuxième ligne par deux espaces et commencez la troisième ligne par quatre espaces.

```
PS C:\Users\myUser\EB_apps\myApp-env>eb config --update 'OptionSettings:
>>  aws:autoscaling:asg:
>>    MinSize: \"1\"'
```

eb console

Description

Ouvre un navigateur pour afficher le tableau de bord de configuration de l'environnement d'Elastic Beanstalk Management Console.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande affiche également la configuration de l'environnement de génération, telle qu'indiquée dans `platform.yaml`, dans Elastic Beanstalk Management Console.

Syntaxe

eb console

eb console ***environment-name***

Options

Nom	Description
Options courantes	

eb create

Description

Crée un environnement et y déploie une version de l'application.

> **Note**
>
> - Pour utiliser eb create sur une application .NET, vous devez créer un package de déploiement comme décrit dans Création d'une solution groupée source pour une application .NET, puis définir la configuration de l'interface de ligne de commande (CLI) pour déployer le package sous forme d'artefact comme décrit dans Déploiement d'un artefact à la place du dossier de projet.
> - Créer des environnements avec l'interface de ligne de commande (CLI) EB nécessite un rôle de service. Vous pouvez créer un rôle de service en créant un environnement dans la console Elastic Beanstalk. Si vous n'avez pas de rôle de service, l'interface de ligne de commande (CLI) EB essaie d'en créer un lorsque vous exécutez `eb create`.

Vous pouvez déployer la version de l'application à partir de différentes sources :

- Par défaut : depuis le code source de l'application dans le répertoire de projet local.
- À l'aide de l'option `--version` : à partir d'une version de l'application qui existe déjà dans votre application.
- Lorsque votre projet de répertoire ne dispose pas d'un code d'application ou que vous utilisez l'option `--sample` : déployée à partir d'un exemple d'application spécifique à la plateforme de votre environnement.

Syntaxe

eb create

eb create ***environment-name***

La longueur d'un nom d'environnement doit être comprise entre 4 et 40 caractères. Il ne peut contenir que des lettres, des chiffres et des traits d'union (-). Un nom d'environnement ne peut pas commencer ou se terminer par un trait d'union.

Si vous incluez un nom d'environnement dans la commande, l'interface de ligne de commande (CLI) EB ne vous invite pas à sélectionner ou créer un rôle de service.

Si vous exécutez la commande sans argument de nom d'environnement, elle s'exécute dans un flux interactif et vous invite à saisir ou sélectionner des valeurs pour certains paramètres. Dans ce flux

interactif, si vous déployez un exemple d'application, l'interface de ligne de commande (CLI) EB vous demande aussi si vous souhaitez télécharger cet exemple d'application sur votre répertoire de projet local. En la téléchargeant, vous pouvez utiliser l'interface de ligne de commande (CLI) EB avec le nouvel environnement pour exécuter des opérations qui nécessitent le code de l'application, comme eb deploy.

Certaines invites de flux interactives ne sont affichées que dans certaines conditions. Par exemple, si vous choisissez d'utiliser un équilibreur de charge Application Load Balancer et que votre compte dispose d'au moins un équilibreur de charge Application Load Balancer partageable, Elastic Beanstalk affiche une invite qui vous demande si vous souhaitez utiliser un équilibreur de charge partagé. Si aucun équilibreur de charge Application Load Balancer partageable n'existe dans votre compte, cette invite ne s'affiche pas.

Options

Aucune de ces options n'est obligatoire. Si vous exécutez eb create sans spécifier d'options, l'interface de ligne de commande (CLI) EB vous invite à entrer ou à sélectionner une valeur pour chaque paramètre.

Nom	Description

`-d`
 	
ou
 	
`--branch_default`
	Définissez l'environnement en tant qu'environnement par défaut pour le référentiel actuel.

`--cfg` *`config-name`*	Utilisez des paramètres de plateforme à partir d'une configuration enregistrée dans `.elasticbeanstalk/saved_configs/` ou votre compartiment Amazon S3. Spécifiez le nom du fichier uniquement, sans l'extension `.cfg.yml`.
`-c` *`subdomain-name`* or `--cname` *`subdomain-name`*	Nom du sous-domaine devant préfixer l'entrée DNS CNAME qui permet d'accéder à votre site web. Type : chaîne Par défaut : le nom de l'environnement

Nom	Description

`-db`
 	
ou
 	
`--database`
	Attache une base de données à l'environnement. Si vous exécutez eb create avec l'option `--database` , mais sans les options `--database.username` et `--database.password` , l'interface de ligne de commande (CLI) EB vous invite à saisir le nom et le mot de passe de l'utilisateur principal de la base de données.

`-db.engine` *`engine`* or `--database.engine` *`engine`*	Le type de moteur de base de données. Si vous exécutez eb create avec cette option, alors l'interface de ligne de commande (CLI) EB lance l'environnement avec une base de données attachée. C'est le cas même si vous n'avez pas exécuté la commande avec l'option `--database` . Type : chaîne Valeurs valides: `mysql`, `oracle-se1` , `postgres`, `sqlserver-ex` , `sqlserver-web` , `sqlserver-se`

Nom	Description
`-db.i` *`instance_type`* or `--database.instance` *`instance_type`*	Type d' EC2 instance Amazon à utiliser pour la base de données. Si vous exécutez eb create avec cette option, alors l'interface de ligne de commande (CLI) EB lance l'environnement avec une base de données attachée. C'est le cas même si vous n'avez pas exécuté la commande avec l'option `--database` . Type : chaîne Valeurs valides : Amazon RDS prend en charge un ensemble standard d'instances de base de données. Pour sélectionner une instance de base de données appropriée pour votre moteur de base de données, vous devez prendre en compte certaines considérations spécifiques. Pour plus d'informations, consultez Classes d'instance de base de données dans le Guide de l'utilisateur Amazon RDS.
`-db.pass` *`password`* or `--database.password` *`password`*	Le mot de passe pour la base de données. Si vous exécutez eb create avec cette option, alors l'interface de ligne de commande (CLI) EB lance l'environnement avec une base de données attachée. C'est le cas même si vous n'avez pas exécuté la commande avec l'option `--database` .

Nom	Description
`-db.size` *`number_of_gigabytes`* or `--database.size` *`number_of_gigabytes`*	Le nombre de gigaoctets (Go) à allouer pour le stockage de base de données. Si vous exécutez eb create avec cette option, alors l'interface de ligne de commande (CLI) EB lance l'environnement avec une base de données attachée. C'est le cas même si vous n'avez pas exécuté la commande avec l'option `--database` . Type : nombre Valeurs valides : • MySQL – `5` à `1024`. La valeur par défaut est `5`. • Postgres – `5` sur `1024`. La valeur par défaut est `5`. • Oracle – `10` à `1024`. La valeur par défaut est `10`. • Microsoft SQL Server Express Edition – `30`. • Microsoft SQL Server Web Edition – `30`. • Microsoft SQL Server Standard Edition – `200`.
`-db.user` *`username`* or `--database.username` *`username`*	Le nom utilisateur pour la base de données. Si vous exécutez eb create avec cette option, alors l'interface de ligne de commande (CLI) EB lance l'environnement avec une base de données attachée même si vous n'avez pas exécuté la commande avec l'option `--database` . Si vous exécutez eb create avec l'option `--database` , mais sans les options `--database.username` et `--database.password` , alors l'interface de ligne de commande (CLI) EB vous invite pour le mot de passe et le nom utilisateur de base de données principale.
`-db.version` *`version`* or `--database.version` *`version`*	Utilisé pour spécifier la version du moteur de base de données. Si cet indicateur est présent, l'environnement se lance avec une base de données avec le numéro de version spécifié, même si l'indicateur `--database` n'est pas présent.

Nom	Description
`--elb-type` *type*	Type de l'équilibreur de charge. Type : chaîne Valeurs valides: `classic`, `application` , `network` Par défaut: `application`

`-es`
 	
ou
 	
`--enable-spot`
	Activez les demandes d'instances Spot pour votre environnement. Pour plus d'informations, consultez Groupe Auto Scaling.
	
	Options connexes :
	
	• `--instance-types`
	• `--on-demand-base-capacity`
	• `--on-demand-above-base-capacity`
	• `--spot-max-price`

`--env-group-suffix` *groupname*	Le nom du groupe à ajouter au nom de l'environnement. A utiliser uniquement avec Compose Environments.
`--envvars`	Propriétés de l'environnement dans une liste séparée par des virgules au format *name* =. *value* Pour connaître les limites, consultez Configuration des propriétés de l'environnement (variables d'environnement).
`-ip` *profile_name* or `--instance_profile` *profile_name*	Le profil d'instance avec le rôle IAM avec les informations d'identification de sécurité temporaires dont votre applicati on a besoin pour accéder aux AWS ressources.

Nom	Description

`-it`
 	
or
 	
`--instance-types` *`type1`*`[,`*`type2`* `...]`
	Liste séparée par des virgules des types d' EC2 instances Amazon que vous souhaitez que votre environnement utilise. Si vous ne spécifiez pas cette option, Elastic Beanstalk fournit les types d'instance par défaut.
	
	Pour de plus amples informations, veuillez consulter EC2 Instances Amazon et Groupe Auto Scaling.
	
	Important
	L'interface de ligne de commande (CLI) EB applique uniquement cette option aux instances Spot. À moins que cette option soit utilisée avec l'option `--enable-spot` , l'interface de ligne de commande (CLI) EB l'ignore. Pour spécifier un type d'instance pour une instance à la demande, utilisez l'option `--intance-type` (pas de « s ») à la place.

Nom	Description

`-i`
 	
or
 	
`--instance_type`
	Le type d' EC2 instance Amazon que vous souhaitez que votre environnement utilise. Si vous ne spécifiez pas cette option, Elastic Beanstalk fournit un type d'instance par défaut.
	
	Pour plus d'informations, consultez EC2 Instances Amazon.
	
	Important
	L'interface de ligne de commande (CLI) EB applique uniquement cette option aux instances à la demande. N'utilisez pas cette option avec l'option `--enable-spot` , car l'interface de ligne de commande (CLI) EB l'ignore lorsque vous le faites. Pour spécifier des types d'instances pour une instance Spot, utilisez l'option `--intance-types` (avec un « s ») à la place.

`-k` *`key_name`* or `--keyname` *`key_name`*	Le nom de la paire de EC2 clés Amazon à utiliser avec le client Secure Shell (SSH) pour se connecter en toute sécurité aux EC2 instances Amazon qui exécutent votre application Elastic Beanstalk. Si vous incluez cette option avec la commande eb create, la valeur que vous fournissez remplace n'importe quel nom de clé que vous pourriez avoir spécifié avec eb init. Valeurs valides : nom de clé existant enregistré auprès d'Amazon EC2

Nom	Description
`-im` *number-of-instances* or `--min-instances` *number-of-instances*	Le nombre minimum d' EC2 instances Amazon dont vous avez besoin pour votre environnement. Type : Nombre (entier) Par défaut: 1 Valeurs valides : de 1 à 10000
`-ix` *number-of-instances* or `--max-instances` *number-of-instances*	Le nombre maximum d' EC2 instances Amazon que vous autorisez votre environnement à avoir. Type : Nombre (entier) Par défaut: 4 Valeurs valides : de 1 à 10000
`--modules` *component-a component-b*	Une liste des environnements de composants à créer. Cela doit être utilisé uniquement avec Compose Environments (Composer des environnements).

`-sb`
 	
ou
 	
`--on-demand-base-capacity`
	Nombre minimal d'instances à la demande que votre groupe Auto Scaling alloue avant de considérer les instances Spot à mesure que votre environnement augmente.
	
	Cette option ne peut être spécifiée qu'avec l'option `--enable-spot` . Pour plus d'informations, consultez Groupe Auto Scaling.
	
	Type : Nombre (entier)
	
	Par défaut: `0`
	
	Valeurs valables : `0` à `--max-instances` (en son absence : `MaxSize` option dans l'espace de noms `aws:autoscaling:asg`)

Nom	Description

`-sp`
 	
ou
 	
`--on-demand-above-base-capacity`
	Pourcentage d'instances à la demande dans le cadre de la capacité supplémentaire allouée par votre groupe Auto Scaling supérieur au nombre d'instances qui est spécifié par l'option `--on-demand-base-capacity` .
	
	Cette option ne peut être spécifiée qu'avec l'option `--enable-spot` . Pour en savoir plus, consultez Groupe Auto Scaling.
	
	Type : Nombre (entier)
	
	Par défaut : `0` pour un environnement à instance unique ; `70` pour un environnement à charge équilibrée
	
	Valeurs valides : de `0` à `100`

Nom	Description
`-p` *platform-version* or `--platform` *platform-version*	La version de la plateforme à utiliser. Vous pouvez spécifier un nom de plateforme, une plateforme et une version de plateforme, une branche de plateforme, un nom de pile de solutions ou un ARN de pile de solutions. Exemples : • `php`, `PHP`, `node.js` – Dernière version pour la plateforme spécifiée • `php-7.2`, `"PHP 7.2"` – Version recommandée (généralement la plus récente) de la plateforme PHP 7.2 • `"PHP 7.2 running on 64bit Amazon Linux"` – Version recommandée (généralement la plus récente) de la plateforme PHP dans cette branche de plateforme • `"64bit Amazon Linux 2017.09 v2.6.3 running PHP 7.1"` – Version de la plateforme PHP spécifiée par ce nom de pile de solutions • `"arn:aws:elasticbeanstalk:us-east-2::platform/PHP 7.1 running on 64bit Amazon Linux/2.6.3"` – Version de la plateforme PHP spécifiée par cet ARN de pile de solutions Utilisez `eb platform list` pour obtenir la liste des configurations disponibles. Si vous spécifiez l'option `--platform` , elle remplace la valeur fournie au cours d'`eb init`.

`-pr`
 	
ou
 	
`--process`
	Prétraitez et validez les fichiers de configuration et le manifeste d'environnement dans le groupe source. La validation des fichiers de configuration permet d'identifier les problèmes avant de déployer la version de l'applica tion dans un environnement.

Nom	Description
`-r` *region* or `--region` *region*	AWS Région dans laquelle vous souhaitez déployer l'application. Pour obtenir la liste des valeurs que vous pouvez spécifier pour cette option, consultez Points de terminaison et quotas AWS Elastic Beanstalk dans Références générales AWS.
`--sample`	Déployez l'exemple d'application dans le nouvel environnement au lieu du code dans votre référentiel.
`--scale` *number-of-instances*	Démarrez avec le nombre d'instances spécifié
`--service-role` *servicerole*	Attribuez un rôle de service autre que celui par défaut à l'environnement. **Note** Ne saisissez pas d'ARN. Saisissez uniquement le nom du rôle. Elastic Beanstalk fait précéder le nom de rôle des valeurs correctes pour créer l'ARN en interne.

Nom	Description
`-ls` *load-balancer* or `--shared-lb` *load-balancer*	Configurez l'environnement pour utiliser un équilibreur de charge partagé. Indiquez le nom ou l'ARN d'un équilibreur de charge partageable dans votre compte, un équilibreur de charge Application Load Balancer que vous avez créé explicitement, et non pas un autre environnement Elastic Beanstalk. Pour plus d'informations, consultez Application Load Balancer partagé. Exemples de paramètres : • `FrontEndLB` – Nom de l'équilibreur de charge • `arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/app/FrontEndLB/0dbf78d8ad96abbc` – ARN de l'équilibreur de charge Application Load Balancer. Vous pouvez spécifier cette option uniquement avec `--elb-type application`. Si vous spécifiez cette option et que vous ne spécifiez pas `--shared-lb`, Elastic Beanstalk crée un équilibreur de charge dédié pour l'environnement.
`-lp` *port* or `--shared-lb-port` *port*	Port d'écoute par défaut de l'équilibreur de charge partagé pour cet environnement. Elastic Beanstalk ajoute une règle d'écoute qui achemine tout le trafic de ce processus d'écoute vers le processus d'environnement par défaut. Pour plus d'informations, consultez Application Load Balancer partagé. Type : Nombre (entier) Par défaut: `80` Valeurs valides : entier représentant un port d'écoute de l'équilibreur de charge partagé.

Nom	Description
`--single`	Créez l'environnement avec une seule EC2 instance Amazon et sans équilibreur de charge. **Warning** Un environnement d'instance unique n'est pas prêt pour la production. Si l'instance devient instable lors du déploiement, ou si Elastic Beanstalk résilie l'instance lors d'une mise à jour de configuration, votre application peut être indisponible pendant un certain temps. Utilisez des environnements d'instance unique pour le développement ou les tests, ou comme environnement intermédiaire. Utilisez des environnements à charge équilibrée pour la production.

`-sm`
 	
ou
 	
`--spot-max-price`
	Prix maximum par heure d'unité, en dollars américains, que vous êtes prêt à payer pour une instance Spot.
	
	Cette option ne peut être spécifiée qu'avec l'option `--enable-spot` . Pour en savoir plus, consultez Groupe Auto Scaling.
	
	Type : nombre (flottant)
	
	Par défaut : prix à la demande, par type d'instance. La valeur de l'option dans ce cas est `null`.
	
	Valeurs valides : de `0.001` à `20.0`
	
	Pour obtenir des recommandations concernant les options de prix maximum pour les instances Spot, consultez l'historique des tarifs des instances Spot dans le guide de EC2 l'utilisateur Amazon.

Nom	Description
`--tags key1=value1[,key2=va`	Ajouter des balises aux ressources de votre environnement. Les balises sont spécifiées sous la forme d'une liste séparée par des virgules et composée de paires `key=value` . Pour de plus amples informations, veuillez consulter Environnements de balisage.

`-t worker`
 	
ou
 	
`--tier worker`
	Créez un environnement de travail. Ignorez cette option pour créer un environnement de serveur web.

`--timeout minutes`	Définissez le nombre de minutes avant que la commande expire.
`--version version_label`	Spécifie la version de l'application que vous voulez voir déployée dans l'environnement à la place du code source d'application dans le répertoire du projet local. Type : chaîne Valeurs valides : une étiquette de version de l'application existante
`--vpc`	Configurez un VPC pour votre environnement. Lorsque vous incluez cette option, l'interface de ligne de commande (CLI) EB vous invite à entrer tous les paramètres obligatoires avant de lancer l'environnement.
`--vpc.dbsubnets subnet1,subnet2`	Spécifie les sous-réseaux pour les instances de base de données dans un VPC. Obligatoire lorsque `--vpc.id` est spécifié.
`--vpc.ec2subnets subnet1,subnet2`	Spécifie les sous-réseaux pour les EC2 instances Amazon dans un VPC. Obligatoire lorsque `--vpc.id` est spécifié.

Nom	Description
`--vpc.elbpublic`	Lance votre équilibreur de charge Elastic Load Balancing dans un sous-réseau public de votre VPC. Vous ne pouvez pas spécifier cette option avec les options `--tier worker` ou `--single`.
`--vpc.elbsubnets` *`subnet1,subnet2`*	Spécifie des sous-réseaux pour l'équilibreur de charge Elastic Load Balancing dans un VPC. Vous ne pouvez pas spécifier cette option avec les options `--tier worker` ou `--single`.
`--vpc.id` *`ID`*	Lance votre environnement dans le VPC spécifié.
`--vpc.publicip`	Lance vos EC2 instances Amazon dans un sous-réseau public de votre VPC. Vous ne pouvez pas spécifier cette option avec l'option `--tier worker`.
`--vpc.securitygroups` *`securitygroup1,securitygroup2`*	Spécifie le groupe de sécurité IDs. Obligatoire lorsque `--vpc.id` est spécifié.
Options courantes	

Output

En cas de succès, la commande vous soumet des questions, puis renvoie le statut de l'opération de création. Si des problèmes se sont produits lors du lancement, vous pouvez utiliser l'opération eb events pour obtenir plus d'informations.

Si vous avez activé le CodeBuild support dans votre application, eb create affiche les informations au CodeBuild fur et à mesure de la création de votre code. Pour plus d'informations sur le CodeBuild support d'Elastic Beanstalk, consultez. Utilisation de l'interface de ligne de commande EB avec AWS CodeBuild

Exemples

L'exemple suivant crée un environnement en mode interactif.

```
$ eb create
Enter Environment Name
(default is tmp-dev): ENTER
Enter DNS CNAME prefix
(default is tmp-dev): ENTER
Select a load balancer type
1) classic
2) application
3) network
(default is 2): ENTER
Environment details for: tmp-dev
  Application name: tmp
  Region: us-east-2
  Deployed Version: app-141029_145448
  Environment ID: e-um3yfrzq22
  Platform: 64bit Amazon Linux 2014.09 v1.0.9 running PHP 5.5
  Tier: WebServer-Standard-1.0
  CNAME: tmp-dev.elasticbeanstalk.com
  Updated: 2014-10-29 21:54:51.063000+00:00
Printing Status:
...
```

L'exemple suivant crée aussi un environnement en mode interactif. Dans cet exemple, votre projet de répertoire ne dispose pas d'un code d'application. La commande déploie un exemple d'application et le télécharge dans votre répertoire de projet local.

```
$ eb create
Enter Environment Name
(default is tmp-dev): ENTER
Enter DNS CNAME prefix
(default is tmp-dev): ENTER
Select a load balancer type
1) classic
2) application
3) network
(default is 2): ENTER
NOTE: The current directory does not contain any source code. Elastic Beanstalk is
 launching the sample application instead.
```

```
Do you want to download the sample application into the current directory?
(Y/n): ENTER
INFO: Downloading sample application to the current directory.
INFO: Download complete.
Environment details for: tmp-dev
  Application name: tmp
  Region: us-east-2
  Deployed Version: Sample Application
  Environment ID: e-um3yfrzq22
  Platform: 64bit Amazon Linux 2014.09 v1.0.9 running PHP 5.5
  Tier: WebServer-Standard-1.0
  CNAME: tmp-dev.elasticbeanstalk.com
  Updated: 2017-11-08 21:54:51.063000+00:00
Printing Status:
...
```

La commande suivante crée un environnement sans afficher d'invites.

```
$ eb create dev-env
Creating application version archive "app-160312_014028".
Uploading test/app-160312_014028.zip to S3. This may take a while.
Upload Complete.
Application test has been created.
Environment details for: dev-env
  Application name: test
  Region: us-east-2
  Deployed Version: app-160312_014028
  Environment ID: e-6fgpkjxyyi
  Platform: 64bit Amazon Linux 2015.09 v2.0.8 running PHP 5.6
  Tier: WebServer-Standard
  CNAME: UNKNOWN
  Updated: 2016-03-12 01:40:33.614000+00:00
Printing Status:
...
```

La commande suivante crée un environnement dans un VPC personnalisé.

```
$ eb create dev-vpc --vpc.id vpc-0ce8dd99 --vpc.elbsubnets subnet-
b356d7c6,subnet-02f74b0c --vpc.ec2subnets subnet-0bb7f0cd,subnet-3b6697c1 --
vpc.securitygroup sg-70cff265
Creating application version archive "app-160312_014309".
Uploading test/app-160312_014309.zip to S3. This may take a while.
Upload Complete.
```

```
Environment details for: dev-vpc
  Application name: test
  Region: us-east-2
  Deployed Version: app-160312_014309
  Environment ID: e-pqkcip3mns
  Platform: 64bit Amazon Linux 2015.09 v2.0.8 running Java 8
  Tier: WebServer-Standard
  CNAME: UNKNOWN
  Updated: 2016-03-12 01:43:14.057000+00:00
Printing Status:
...
```

eb deploy

Description

Déploie le groupe source de l'application depuis le répertoire du projet initialisé vers l'application en cours d'exécution.

Si git est installé, l'interface de ligne de commande (CLI) EB utilise la commande `git archive` pour créer un fichier `.zip` à partir du contenu de la commande `git commit` la plus récente.

Cependant, quand `.ebignore` est présent dans le répertoire de votre projet, l'interface de ligne de commande (CLI) EB n'utilise pas de commandes et de sémantiques Git pour créer votre bundle de fichiers source. Autrement dit, l'interface de ligne de commande (CLI) EB ignore les fichiers spécifiés dans `.ebignore` et inclut tous les autres fichiers. Plus particulièrement, elle comprend des fichiers source non validés.

> **Note**
>
> Vous pouvez configurer l'interface de ligne de commande (CLI) EB pour déployer un artefact de votre processus de construction au lieu de créer un fichier ZIP de votre dossier de projet. Consultez Déploiement d'un artefact à la place du dossier de projet pour plus de détails.

Syntaxe

eb deploy

eb deploy ***environment-name***

Options

Nom	Description
`-l` *version_label* or `--label` *version_label*	Spécifiez une étiquette à utiliser pour la version créée par l'interface de ligne de commande (CLI) EB. Si l'étiquette a déjà été utilisée, l'interface de ligne de commande (CLI) EB redéploie la version précédente avec cette étiquette. Type : chaîne
`--env-group-suffix` *groupname*	Nom du groupe à ajouter au nom de l'environnement. A utiliser uniquement avec Compose Environments.

`-m "`*version_description* `"`
 	
or
 	
`--message "`*version_description* `"`
	La description de la version de l'application, entourée de guillemets doubles.
	
	Type : Chaîne

`--modules` *component-a component-b*	Liste des composants à mettre à jour. A utiliser uniquement avec Compose Environments.

`-p`
 	
ou
 	
`--process`
	Prétraitez et validez les fichiers de configuration et le manifeste d'environnement dans le groupe source. La validation des fichiers de configuration permet d'identifier les problèmes avant de déployer la version de l'application dans un environnement.

`--source codecommit/` *repository-name*`/`*branch-name*	CodeCommit référentiel et branche.
`--staged`	Déployez des fichiers mis en lots dans l'index git au lieu de la validation HEAD.
`--timeout` *minutes*	Le nombre de minutes avant que la commande expire.

Nom	Description
`--version` *version_label*	Une version de l'application existante à déployer. Type : chaîne
Options courantes	

Output

En cas de réussite, la commande renvoie l'état de l'opération `deploy`.

Si vous avez activé le CodeBuild support dans votre application, eb deploy affiche les informations à partir CodeBuild de la création de votre code. Pour plus d'informations sur le CodeBuild support d'Elastic Beanstalk, consultez. Utilisation de l'interface de ligne de commande EB avec AWS CodeBuild

Exemple

L'exemple suivant déploie l'application en cours.

```
$ eb deploy
2018-07-11 21:05:22    INFO: Environment update is starting.
2018-07-11 21:05:27    INFO: Deploying new version to instance(s).
2018-07-11 21:05:53    INFO: New application version was deployed to running EC2
 instances.
2018-07-11 21:05:53    INFO: Environment update completed successfully.
```

eb events

Description

Renvoie les événements les plus récents pour l'environnement.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande renvoie également les événements les plus récents pour l'environnement de génération.

Syntaxe

eb events

eb events ***environment-name***

Options

Nom	Description

`-f`
 	
ou
 	
`--follow`
	Événements de flux. Pour annuler, appuyez sur CTRL+C.

Output

En cas de succès, la commande renvoie des événements récents.

Exemple

L'exemple suivant renvoie les événements les plus récents.

```
$ eb events
2014-10-29 21:55:39     INFO    createEnvironment is starting.
2014-10-29 21:55:40     INFO    Using elasticbeanstalk-us-west-2-111122223333 as Amazon
 S3 storage bucket for environment data.
2014-10-29 21:55:57     INFO    Created load balancer named: awseb-e-r-AWSEBLoa-
NSKUOK5X6Z9J
2014-10-29 21:56:16     INFO    Created security group named: awseb-e-rxgrhjr9bx-stack-
AWSEBSecurityGroup-1UUHU5LZ20ZY7
2014-10-29 21:57:18     INFO    Waiting for EC2 instances to launch. This may take a
 few minutes.
2014-10-29 21:57:18     INFO    Created Auto Scaling group named: awseb-e-rxgrhjr9bx-
stack-AWSEBAutoScalingGroup-1TE320ZCJ9RPD
2014-10-29 21:57:22     INFO    Created Auto Scaling group policy named:
 arn:aws:autoscaling:us-east-2:11122223333:scalingPolicy:2cced9e6-859b-421a-
be63-8ab34771155a:autoScalingGroupName/awseb-e-rxgrhjr9bx-stack-
AWSEBAutoScalingGroup-1TE320ZCJ9RPD:policyName/awseb-e-rxgrhjr9bx-stack-
AWSEBAutoScalingScaleUpPolicy-1I2ZSNVU4APRY
```

```
2014-10-29 21:57:22     INFO    Created Auto Scaling group policy named:
 arn:aws:autoscaling:us-east-2:11122223333:scalingPolicy:1f08b863-
bf65-415a-b584-b7fa3a69a0d5:autoScalingGroupName/awseb-e-rxgrhjr9bx-stack-
AWSEBAutoScalingGroup-1TE320ZCJ9RPD:policyName/awseb-e-rxgrhjr9bx-stack-
AWSEBAutoScalingScaleDownPolicy-1E3G7PZKZPSOG
2014-10-29 21:57:25     INFO    Created CloudWatch alarm named: awseb-e-rxgrhjr9bx-
stack-AWSEBCloudwatchAlarmLow-VF5EJ549FZBL
2014-10-29 21:57:25     INFO    Created CloudWatch alarm named: awseb-e-rxgrhjr9bx-
stack-AWSEBCloudwatchAlarmHigh-LA9YEW3O6WJO
2014-10-29 21:58:50     INFO    Added EC2 instance 'i-c7ee492d' to Auto ScalingGroup
 'awseb-e-rxgrhjr9bx-stack-AWSEBAutoScalingGroup-1TE320ZCJ9RPD'.
2014-10-29 21:58:53     INFO    Successfully launched environment: tmp-dev
2014-10-29 21:59:14     INFO    Environment health has been set to GREEN
2014-10-29 21:59:43     INFO    Adding instance 'i-c7ee492d' to your environment.
```

eb health

Description

Renvoie l'intégrité la plus récente pour l'environnement.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande renvoie également l'état le plus récent pour l'environnement de génération.

Syntaxe

eb health

eb health ***environment-name***

Options

Nom	Description

`-r`
 	
ou
 	
`--refresh`
	Affichez des informations d'état de santé interactivement et mettez à jour toutes les 10 secondes à mesure que de nouvelles informations sont rapportées.

Nom	Description
`--mono`	N'affichez pas la couleur dans le résultat.

Output

En cas de réussite, la commande renvoie l'intégrité récente.

Exemple

L'exemple suivant renvoie les informations d'intégrité les plus récentes pour un environnement Linux.

```
~/project $ eb health
 elasticBeanstalkExa-env                                Ok
 2015-07-08 23:13:20
WebServer
 Ruby 2.1 (Puma)
  total      ok    warning  degraded  severe    info   pending  unknown
    5        5        0        0        0        0        0        0

  instance-id   status     cause
                                    health
    Overall     Ok
  i-d581497d    Ok
  i-d481497c    Ok
  i-136e00c0    Ok
  i-126e00c1    Ok
  i-8b2cf575    Ok

  instance-id   r/sec    %2xx   %3xx   %4xx   %5xx      p99      p90      p75     p50
   p10                               requests
    Overall     671.8   100.0    0.0    0.0    0.0    0.003    0.002    0.001   0.001
 0.000
  i-d581497d    143.0    1430      0      0      0    0.003    0.002    0.001   0.001
 0.000
  i-d481497c    128.8    1288      0      0      0    0.003    0.002    0.001   0.001
 0.000
  i-136e00c0    125.4    1254      0      0      0    0.004    0.002    0.001   0.001
 0.000
  i-126e00c1    133.4    1334      0      0      0    0.003    0.002    0.001   0.001
 0.000
```

```
 i-8b2cf575    141.2    1412      0      0      0    0.003    0.002    0.001   0.001
0.000

 instance-id   type       az   running     load 1  load 5      user%  nice%  system%
idle%   iowait%                            cpu
 i-d581497d    t2.micro   1a   12 mins       0.0    0.04        6.2    0.0      1.0
92.5       0.1
 i-d481497c    t2.micro   1a   12 mins       0.01    0.09        5.9    0.0      1.6
92.4       0.1
 i-136e00c0    t2.micro   1b   12 mins       0.15    0.07        5.5    0.0      0.9
93.2       0.0
 i-126e00c1    t2.micro   1b   12 mins       0.17    0.14        5.7    0.0      1.4
92.7       0.1
 i-8b2cf575    t2.micro   1c   1 hour        0.19    0.08        6.5    0.0      1.2
92.1       0.1

 instance-id   status     id   version              ago
                                 deployments
 i-d581497d    Deployed   1    Sample Application   12 mins
 i-d481497c    Deployed   1    Sample Application   12 mins
 i-136e00c0    Deployed   1    Sample Application   12 mins
 i-126e00c1    Deployed   1    Sample Application   12 mins
 i-8b2cf575    Deployed   1    Sample Application   1 hour
```

eb init

Description

Définit les valeurs par défaut pour les applications Elastic Beanstalk créées avec l'interface de ligne de commande (CLI) EB en vous soumettant une série de questions.

> **Note**
>
> Les valeurs que vous définissez s'eb initappliquent au répertoire et au référentiel actuels de l'ordinateur en cours.
> La commande crée une application Elastic Beanstalk dans votre compte. Pour créer un environnement Elastic Beanstalk, exécutez eb create après l'exécution de eb init.

Syntaxe

eb init

eb init *application-name*

Options

Si vous exécutez eb init sans spécifier l'option `--platform`, l'interface de ligne de commande (CLI) EB vous invite à entrer une valeur pour chaque paramètre.

> **Note**
>
> Pour utiliser eb init afin de créer une nouvelle paire de clés, l'outil `ssh-keygen` doit être installé sur votre ordinateur local et disponible à partir de la ligne de commande.

Nom	Description	

`-i`
 	
`--interactive`
	Force l'interface de ligne de commande (CLI) EB à vous inviter à fournir une valeur pour chaque option de commande eb init.
	
	Note
	La commande `init` vous invite à fournir des valeurs pour les options de commande eb init qui n'ont pas une valeur (par défaut). Après la première fois que vous exécutez la commande eb init dans un répertoire, l'interface de ligne de commande (CLI) EB ne vous soumettra peut-être aucune options de commande. Par conséquent, utilisez l'option `--interactive` lorsque vous voulez modifier un paramètre que vous avez préalablement défini.	

`-k` *keyname*	Le nom de la paire de EC2 clés Amazon à utiliser avec le client Secure Shell (SSH) pour se connecter en	

Nom	Description	
`--keyname` *keyname*	toute sécurité aux EC2 instances Amazon exécutant votre application Elastic Beanstalk.	
`--modules` *folder-1 folder-2*	Liste des répertoires enfants à initialiser. A utiliser uniquement avec Compose Environments.	

Nom	Description	
`-p` *platform-version* `--platform` *platform-version*	La version de la plateforme à utiliser. Vous pouvez spécifier un nom de plateforme, une plateforme et une version de plateforme, une branche de plateform e, un nom de pile de solutions ou un ARN de pile de solutions. Exemples : • `php`, `PHP`, `node.js` – Dernière version pour la plateforme spécifiée • `php-7.2`, `"PHP 7.2"` – Version recommandée (généralement la plus récente) de la plateforme PHP 7.2 • `"PHP 7.2 running on 64bit Amazon Linux"` – Version recommandée (généralement la plus récente) de la plateforme PHP dans cette branche de plateforme • `"64bit Amazon Linux 2017.09 v2.6.3 running PHP 7.1"` – Version de la plateforme PHP spécifiée par ce nom de pile de solutions • `"arn:aws:elasticbeanstalk:us-east-2::platform/PHP 7.1 running on 64bit Amazon Linux/2.6.3"` – Version de la plateform e PHP spécifiée par cet ARN de pile de solutions Utilisez `eb platform list` pour obtenir la liste des configurations disponibles. Spécifiez l'option `--platform` pour éviter la configuration interactive. **Note** Si vous spécifiez cette option, l'interface de ligne de commande (CLI) EB ne vous invite pas à saisir les valeurs d'autres options. Au	

Nom	Description	
	lieu de cela, il est seul responsable des valeurs par défaut pour chaque option. Vous pouvez spécifier des options pour tout ce pour quoi vous ne souhaitez pas utiliser les valeurs par défaut.	
`--source codecommit/` *repository-name*`/`*branch-name*	CodeCommit référentiel et branche.	
`--tags` *key1*`=`*value1*	Balisez votre application. Les balises sont spécifiées sous la forme d'une liste séparée par des virgules et composée de paires `key=value` . Pour en savoir plus, consultez L'étiquette des applications.	
Options courantes		

CodeBuild soutien

Si vous exécutez eb init dans un dossier contenant un fichier buildspec.yml, Elastic Beanstalk analyse le fichier pour une entrée eb_codebuild_settings avec des options spécifiques à Elastic Beanstalk. Pour plus d'informations sur le CodeBuild support d'Elastic Beanstalk, consultez. Utilisation de l'interface de ligne de commande EB avec AWS CodeBuild

Output

En cas de succès, la commande vous guide à travers la mise en place d'une nouvelle application Elastic Beanstalk grâce à une série d'invites.

Exemple

L'exemple de demande suivant initialise l'interface de ligne de commande (CLI) EB et vous invite à saisir des informations relatives à votre application. Remplacez *placeholder* le texte par vos propres valeurs.

```
$ eb init -i
Select a default region
1) us-east-1 : US East (N. Virginia)
2) us-west-1 : US West (N. California)
3) us-west-2 : US West (Oregon)
4) eu-west-1 : Europe (Ireland)
5) eu-central-1 : Europe (Frankfurt)
6) ap-south-1 : Asia Pacific (Mumbai)
7) ap-southeast-1 : Asia Pacific (Singapore)
...
(default is 3): 3

Select an application to use
1) HelloWorldApp
2) NewApp
3) [ Create new Application ]
(default is 3): 3

Enter Application Name
(default is "tmp"):
Application tmp has been created.

It appears you are using PHP. Is this correct?
(y/n): y

Select a platform branch.
1) PHP 7.2 running on 64bit Amazon Linux
2) PHP 7.1 running on 64bit Amazon Linux (Deprecated)
3) PHP 7.0 running on 64bit Amazon Linux (Deprecated)
4) PHP 5.6 running on 64bit Amazon Linux (Deprecated)
5) PHP 5.5 running on 64bit Amazon Linux (Deprecated)
6) PHP 5.4 running on 64bit Amazon Linux (Deprecated)
(default is 1): 1
Do you want to set up SSH for your instances?
(y/n): y

Select a keypair.
1) aws-eb
2) [ Create new KeyPair ]
(default is 2): 1
```

eb labs

Description

Sous-commandes de eb labs support work-in-progress ou de fonctionnalité expérimentale. Ces commandes peuvent être supprimées ou retravaillées dans des versions futures de l'interface de ligne de commande (CLI) EB et ne sont pas garanties pour être compatibles.

Pour une liste des descriptions et des sous-commandes disponibles, exécutez eb labs --help.

eb list

Description

Répertorie tous les environnements dans l'application actuelle ou tous les environnements dans toutes les applications, comme indiqué par l'option `--all`.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande répertorie également les environnements de génération.

Syntaxe

eb list

Options

Nom	Description

`-a`
ou
`--all`
	Répertorie tous les environnements de toutes les applications.

`-v`
ou
`--verbose`
	Fournit des informations plus détaillées sur tous les environnements, y compris des instances.

Nom	Description
Options courantes	

Output

En cas de réussite, la commande renvoie une liste de noms d'environnements dans laquelle votre environnement actuel est marqué d'un astérisque (*).

Exemple 1

L'exemple suivant répertorie vos environnements et indique que tmp-dev est votre environnement par défaut.

```
$ eb list
* tmp-dev
```

Exemple 2

L'exemple suivant répertorie vos environnements avec plus de détails.

```
$ eb list --verbose
Region: us-west-2
Application: tmp
    Environments: 1
        * tmp-dev : ['i-c7ee492d']
```

eb local

Description

Utilisez eb local run pour exécuter des conteneurs de votre application localement dans Docker. Vérifiez l'état du conteneur de l'application avec eb local status. Ouvrez l'application dans un navigateur web avec eb local open. Récupérez l'emplacement des journaux de l'application avec eb local logs.

eb local setenv et eb local printenv vous permettent de définir et d'afficher des variables d'environnement qui sont fournies par des conteneurs Docker que vous exécutez localement avec eb local run.

Vous devez exécuter toutes les commandes eb local dans le répertoire de projet d'une application Docker ayant fait l'objet d'une initialisation en tant que référentiel de l'interface de ligne de commande (CLI) EB en utilisant eb init.

> **Note**
>
> Utilisez eb local sur un ordinateur local exécutant les systèmes d'exploitation Linux ou macOS. La commande ne prend pas en charge Windows.
> Avant d'utiliser la commande sur macOS, installez Docker pour Mac et assurez-vous que boot2docker n'est pas installé (ou n'est pas dans le chemin d'exécution). La commande eb local essaie d'utiliser boot2docker s'il est présent, mais ne fonctionne pas bien avec elle sur macOS.

Syntaxe

eb local run

eb local status

eb local open

eb local logs

eb local setenv

eb local printenv

Options

eb local run

Nom	Description
`--envvars key1=value1,key2=value2`	Définit des variables d'environnement que l'interface de ligne de commande (CLI) EB va passer aux conteneurs Docker locaux.

Nom	Description
	Dans des environnements multicontainer, toutes les variables sont passées à tous les conteneurs.
`--port` *`hostport`*	Mappe un port sur l'hôte au port exposé sur le conteneur. Si vous ne spécifiez pas cette option, l'interface de ligne de commande (CLI) EB utilise le même port sur l'hôte et sur le conteneur. Cette option fonctionne uniquement avec les applications de plateforme Docker. Elle ne s'applique pas à la plateforme multi-conteneurs Docker.
Options courantes	

eb local status

eb local open

eb local logs

eb local setenv

eb local printenv

Nom	Description
Options courantes	

Output

eb local run

Messages d'état de Docker. Reste actif aussi longtemps que l'application est en cours d'exécution. Appuyez sur Ctrl+C pour arrêter l'application.

eb local status

L'état de chaque conteneur utilisé par l'application, en cours d'exécution ou non.

eb local open

Ouvre l'application dans un navigateur web puis se ferme.

eb local logs

L'emplacement des journaux générés dans votre répertoire de projet par des applications qui s'exécutent localement sous eb local run.

eb local setenv

Aucune

eb local printenv

Le nom et les valeurs de variables d'environnement définies avec eb local setenv.

Exemples

eb local run

```
~/project$ eb local run
Creating elasticbeanstalk_phpapp_1...
Creating elasticbeanstalk_nginxproxy_1...
Attaching to elasticbeanstalk_phpapp_1, elasticbeanstalk_nginxproxy_1
phpapp_1     | [23-Apr-2015 23:24:25] NOTICE: fpm is running, pid 1
phpapp_1     | [23-Apr-2015 23:24:25] NOTICE: ready to handle connections
```

eb local status

Affichez le statut de vos conteneurs locaux :

```
~/project$ eb local status
Platform: 64bit Amazon Linux 2014.09 v1.2.1 running Multi-container Docker 1.3.3
 (Generic)
Container name: elasticbeanstalk_nginxproxy_1
Container ip: 127.0.0.1
Container running: True
Exposed host port(s): 80
Full local URL(s): 127.0.0.1:80
```

```
Container name: elasticbeanstalk_phpapp_1
Container ip: 127.0.0.1
Container running: True
Exposed host port(s): None
Full local URL(s): None
```

eb local logs

Affichez le chemin d'accès au journal pour le projet actuel :

```
~/project$ eb local logs
Elastic Beanstalk will write logs locally to /home/user/project/.elasticbeanstalk/logs/
local.
Logs were most recently created 3 minutes ago and written to /home/user/
project/.elasticbeanstalk/logs/local/150420_234011665784.
```

eb local setenv

Définissez des variables d'environnement à utiliser avec eb local run.

```
~/project$ eb local setenv PARAM1=value
```

Imprimez des variables d'environnement définies avec eb local setenv.

```
~/project$ eb local printenv
Environment Variables:
PARAM1=value
```

eb logs

Description

La eb logs commande a trois objectifs distincts : activer ou désactiver le streaming des CloudWatch journaux vers Logs, récupérer les journaux d'instance ou les journaux CloudWatch des journaux, et demander une analyse de votre environnement basée sur l'IA. Avec l'option `--cloudwatch-logs` (`-cw`), la commande active ou désactive le streaming de journaux. Avec l'option `--analyze` (`-ai`), la commande demande une analyse basée sur l'IA des journaux, des événements et de l'état de l'instance de votre environnement. Sans aucune de ces options, il récupère les journaux.

En cas de récupération des journaux, spécifiez l'option `--all` ou `--zip`, `--stream` pour récupérer les journaux complets. Si vous ne spécifiez aucune de ces options, Elastic Beanstalk récupère les journaux de queue.

La commande traite les journaux associés à l'environnement par défaut ou l'environnement spécifié. Les journaux concernés varient selon le type de conteneur. Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande traite également les journaux associés à l'environnement de génération.

Pour de plus amples informations, veuillez consulter the section called “CloudWatch Journaux”. Pour plus d'informations sur l'analyse basée sur l'IA, consultez. Analyse de l'environnement basée sur l'IA

Syntaxe

Pour activer ou désactiver le streaming des CloudWatch journaux vers Logs, procédez comme suit :

```
eb logs --cloudwatch-logs [enable | disable] [--cloudwatch-log-source instance |
 environment-health | all] [environment-name]
```

Pour récupérer des journaux d'instance :

```
eb logs [-all | --zip | --stream] [--cloudwatch-log-source instance] [--
instance instance-id] [--log-group log-group] [environment-name]
```

Pour extraire les journaux d'intégrité de l'environnement :

```
eb logs [-all | --zip | --stream] --cloudwatch-log-source environment-health
 [environment-name]
```

Pour demander une analyse basée sur l'IA :

```
eb logs --analyze [environment-name]
```

Options

Nom	Description
`-cw [enable | disable]` or	Active ou désactive le streaming des journaux vers CloudWatch Logs. Si aucun argument n'est fourni, le streaming de journaux

Nom	Description
`--cloudwatch-logs [enable | disable]`	est activé. Si l'option `--cloudwatch-log-source (-cls)` n'est pas ajoutée, le streaming des journaux d'instance est activé ou désactivé.

`-cls instance | environment-health | all`
 	
or
 	
`--cloudwatch-log-source instance | environment-health | all`
	Spécifie la source des journaux lorsque vous travaillez avec CloudWatch des journaux. Avec la forme d'activation ou de désactivation de la commande, il s'agit des journaux pour lesquels vous pouvez activer ou désactiver le streaming CloudWatch des journaux. Avec le formulaire de récupération de la commande, il s'agit des journaux à récupérer à partir de CloudWatch Logs.
	
	Valeurs valides :
	
	• Avec `--cloudwatch-logs` (activation ou désactivation) - `instance | environment-health | all`
	• Sans `--cloudwatch-logs` (extraction) - `instance | environment-health`
	
	Significations des valeurs :
	
	• `instance` (valeur par défaut) - Journaux d'instance
	• `environment-health` - Journaux d'intégrité d'environnement (uniquement pris en charge lorsque la fonction d'intégrité améliorée est activée dans l'environnement)
	• `all` - Les deux sources de journaux

`-a`
 	
ou
 	
`--all`
	Récupère les journaux complets et les enregistre dans le répertoire `.elasticbeanstalk/logs` .

Nom	Description

`-z`
 	
ou
 	
`--zip`
	Récupère les journaux complets, les compresse dans un fichier `.zip`, puis enregistre le fichier dans le répertoire `.elasticbeanstalk/logs` .

`--stream`	Diffuse (sorties en continu) les journaux complets. Avec cette option, la commande poursuit son exécution jusqu'à ce que vous l'interrompiez (appuyez sur **`Ctrl+C`**).
`-i` *`instance-id`* ou `--instance` *`instance-id`*	Extrait les journaux de l'instance spécifiée uniquement.

Nom	Description

`-g log-group`
 	
or
 	
`--log-group log-group`
	Spécifie le groupe de CloudWatch journaux à partir duquel les journaux doivent être récupérés. L'option n'est valide que lorsque le streaming du journal d'instance vers CloudWatch Logs est activé.
	
	Si le streaming des journaux d'instance est activé et que vous ne spécifiez pas l'option `--log-group` , le groupe de journaux par défaut est l'un des suivants :
	
	• Amazon Linux 2 : `/aws/elasticbeanstalk/ environment-name /var/log/eb-engine.log`
	• Plateformes Windows – `/aws/elasticbeanstalk/ environment-name /EBDeploy-Log`
	• AMI Amazon Linux (AL1) — `/aws/elasticbeanstalk/ environment-name /var/log/eb-activity.log`
	
	Note
	Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.
	
	Pour plus d'informations sur le groupe de journaux correspondant à chaque fichier journal, consultez Comment Elastic Beanstalk configure les journaux CloudWatch.

Nom	Description

`-ai`
 	
or
 	
`--analyze`
	Demande une analyse basée sur l'IA des journaux, des événements et de l'état de l'instance de votre environnement. L'analyse utilise Amazon Bedrock pour identifier les causes profondes et recommander des solutions aux problèmes de santé environnementale. La commande envoie la demande, attend que l'analyse soit terminée, puis affiche les résultats.
	
	Cette option n'est pas compatible avec `--instance --all`,`--zip`,`--log-group` , ou`--cloudwatch-logs` .
	
	Pour les prérequis et les autorisations requises, voirAnalyse de l'environnement basée sur l'IA.

Options courantes	

Output

Par défaut, affiche les journaux directement sur le terminal. Utilise un programme de pagination pour afficher la sortie. Appuyez sur **`Q`** ou **`q`** pour quitter.

Avec `--stream`, affiche les journaux existants sur le terminal et poursuit son exécution. Appuyez sur **`Ctrl+C`** pour quitter.

Avec `--all` et `--zip`, enregistre les journaux dans des fichiers locaux et affiche l'emplacement des fichiers.

Avec`--analyze`, affiche l'analyse générée par l'IA directement dans le terminal une fois l'analyse terminée.

Exemples

L'exemple suivant active le streaming des journaux d'instance vers CloudWatch Logs.

```
$ eb logs -cw enable
Enabling instance log streaming to CloudWatch for your environment
After the environment is updated you can view your logs by following the link:
https://console.aws.amazon.com/cloudwatch/home?region=us-east-1#logs:prefix=/aws/
elasticbeanstalk/environment-name/
```

```
Printing Status:
2018-07-11 21:05:20    INFO: Environment update is starting.
2018-07-11 21:05:27    INFO: Updating environment environment-name's configuration
 settings.
2018-07-11 21:06:45    INFO: Successfully deployed new configuration to environment.
```

L'exemple suivant extrait les journaux d'instance dans un fichier `.zip`.

```
$ eb logs --zip
Retrieving logs...
Logs were saved to /home/workspace/environment/.elasticbeanstalk/logs/150622_173444.zip
```

L'exemple suivant demande une analyse de l'environnement basée sur l'IA.

```
$ eb logs --analyze
Analyzing environment...
No critical issues detected. Environment appears healthy.

---
Note: This analysis was generated by AI.
```

eb migrate

Description

Fait migrer les sites et applications Internet Information Services (IIS) d'un serveur Windows vers Elastic Beanstalk. La commande empaquète vos applications, préserve leurs configurations et les déploie dans un nouvel environnement Elastic Beanstalk.

Pour plus d'informations sur la migration de vos sites et applications IIS, consultezMigration des applications IIS.

> **Note**
>
> Avant d'utiliser cette commande, assurez-vous que votre système répond aux exigences suivantes :
>
> - Internet Information Services (IIS) version 7.0 ou ultérieure
> - Web Deploy 3.6 ou version ultérieure installé
> - Privilèges administratifs sur le serveur Windows

- AWS informations d'identification configurées avec les autorisations appropriées
- Votre serveur source dispose d'un accès Internet sortant aux AWS services.

Les étapes suivantes résument le processus de migration :

1. Découvrez les sites IIS et leurs configurations.
2. Package du contenu et de la configuration de l'application.
3. Créez l'environnement et l'application Elastic Beanstalk.
4. Déployez l'application avec des paramètres préservés.

La commande crée des artefacts de migration dans un répertoire structuré, comme indiqué dans la liste suivante :

```
C:\migration_workspace\
### .\migrations\latest\
    ### upload_target.zip
    ### upload_target\
        ### [SiteName].zip                    # One ZIP per default application of
 IIS site
        ### [SiteName-ApplicationName].zip     # One ZIP per additional application
        ### aws-windows-deployment-manifest.json
        ### ebmigrateScripts\
            ### site_installer.ps1             # Site installation scripts
            ### permission_handler.ps1         # Permission management
            ### other helper scripts
```

eb migrate cleanupÀ utiliser pour gérer ces artefacts.

Syntaxe

eb migrate ***[options]***

eb migrate explore ***[options]***

eb migrate cleanup ***[options]***

Lorsqu'il est exécuté sans arguments, eb migrate fonctionne en mode non interactif. Pour l'exécuter en mode interactif, exécutezeb migrate --interactive.

La commande du mode interactif demande les informations suivantes :

- Sélection des sites IIS à migrer
- Noms de l'environnement et des applications
- Sélection de la version de la plateforme
- Type d'instance et autres options de configuration

Sous-commandes

explorer

La eb migrate explore sous-commande examine votre serveur IIS et répertorie les sites disponibles.

Utilisez cette commande pour afficher les informations suivantes :

- Afficher tous les sites IIS sur le serveur
- Avec `--verbose`, inspectez la configuration détaillée, y compris :
  - Liaisons et ports du site
  - Pools d'applications
  - Répertoires virtuels et leurs chemins physiques
  - Paramètres d'authentification

```
PS C:\migrations_workspace >  eb migrate explore
Default Web Site
Site2
site3
router
```

```
PS C:\migrations_workspace >  eb migrate explore --verbose
1: Default Web Site:
  - Bindings:
    - *:8083:
  - Application '/':
    - Application Pool: DefaultAppPool
    - Enabled Protocols: http
    - Virtual Directories:
      - /:
```

```
        - Physical Path: C:\inetpub\wwwroot
        - Logon Method: ClearText
  - Application '/dotnet-6-0':
    - Application Pool: DefaultAppPool
    - Enabled Protocols: http
    - Virtual Directories:
      - /:
        - Physical Path: C:\inetpub\AspNetCoreWebApps\CoreWebApp-6-0
        - Logon Method: ClearText
  - Application '/dotnet-8-0':
    - Application Pool: DefaultAppPool
    - Enabled Protocols: http
    - Virtual Directories:
      - /:
        - Physical Path: C:\inetpub\AspNetCoreWebApps\CoreWebApp-8-0
        - Logon Method: ClearText
2: Site2:
  - Bindings:
    - *:8081:
...
```

nettoyage

La eb migrate cleanup sous-commande gère les artefacts de migration à l'aide des actions suivantes :

- Préserver la dernière migration réussie dans `./migrations/latest`
- Suppression des anciens répertoires de migration
- Gestion des fichiers de configuration critiques

```
PS C:\migrations_workspace >   eb migrate cleanup
Are you sure you would like to cleanup older artifacts within `./migrations/`? (y/N):
```

`--force`À utiliser pour ignorer les instructions de confirmation pendant le nettoyage.

```
PS C:\migrations_workspace >   eb migrate cleanup --force
```

Options

Aucune de ces options n'est obligatoire. Si vous exécutez eb migrate sans aucune option, l'interface de ligne de commande EB s'exécutera en mode non interactif. Aveceb migrate --interactive, l'EB CLI vous invite à saisir ou à sélectionner une valeur pour les paramètres requis.

Nom	Description
`-a` *`application-name`* or `--application-name` *`application-name`*	Nom de la nouvelle application Elastic Beanstalk. Type : Chaîne Par défaut : EBMigrated App
`--archive` *`directory-or-zip`*	Le répertoire ou le fichier ZIP contenant le code source précédemment généré pareb migrate --archive-only. Utilisez cette option pour déployer un package de migration créé précédemment. Exemple : `--archive .\migrations\latest\upload_target` ou `--archive .\migrations\latest\upload_target.zip`

`-ao`
 	
or
 	
`--archive-only`
	Créez uniquement le répertoire d'archives de destination sans déploiement.
	
	Le répertoire obtenu peut être déployé manuellement à l'eb migrateaide de l'`archive`option, oueb deploy.

`-c` *`subdomain-name`* or `--cname` *`subdomain-name`*	Le nom du sous-domaine devant préfixer l'entrée DNS CNAME de votre application migrée. Type : Chaîne Par défaut : le nom de l'environnement

Nom	Description

`-cf`
 	
or
 	
`--copy-firewall-config`
	Copiez la configuration du pare-feu du serveur source vers la destination pour tous les ports HTTP dotés de liaisons actives.
	
	Crée les règles de groupe de sécurité correspondantes dans AWS.

`-es` *`snapshot-id`* [*`snapshot-id`* ...] or `--ebs-snapshots` *`snapshot-id`* [*`snapshot-id`* ...]	Liste séparée par des virgules des instantanés Amazon EBS IDs à associer à l'environnement. Exemple : `--ebs-snapshots snap-1234567890abcdef0, snap-0987654321fedcba1`
`--encrypt-ebs-volumes`	Appliquez le chiffrement à tous les nouveaux volumes Amazon EBS. **Important** Il s'agit d'un paramètre à l'échelle du compte qui affecte toutes les futures créations de volumes Amazon EBS.
`-e` *`environment-name`* or `--environment-name` *`environment-name`*	Nom du nouvel environnement Elastic Beanstalk. Type : Chaîne Par défaut : EBMigrated Env Contraintes : Doit comporter entre 4 et 40 caractères. Ne peut contenir que des lettres, des chiffres et des traits d'union. Il ne peut pas commencer ni se terminer par un trait d'union.

Nom	Description
`--force`	Ignorez les instructions de confirmation pendant les opérations. Lorsqu'il est utilisé avec une cleanup sous-commande, supprime les artefacts de migration sans confirmation.
`-ip` *`profile-name`* or `--instance-profile` *`profile-name`*	Profil d'instance à associer aux EC2 instances Amazon de l'environnement. S'il n'est pas spécifié, crée un profil d'instance par défaut avec les autorisations d'accès aux ressources Elastic Beanstalk. Pour de plus amples informations, veuillez consulter the section called “Profil d’instance”.
`-i` *`instance-type`* or `--instance-type` *`instance-type`*	Type d' EC2 instance Amazon pour votre environnement Elastic Beanstalk. Type : Chaîne Par défaut : c5.2xlarge Pour connaître les types d'instances disponibles, consultez les types d' EC2 instances Amazon dans le guide de EC2 l'utilisateur Amazon.

`-in`
 	
or
 	
`--interactive`
	Forcer le mode interactif pour le processus de migration.
	
	Vous invite à saisir les valeurs de configuration même lorsque des valeurs par défaut sont disponibles.

Nom	Description

`-k key-name`
 	
or
 	
`--keyname key-name`
	Paire de EC2 clés Amazon pour permettre l'accès RDP aux instances de l'environnement.
	
	Utile pour étudier les problèmes au niveau de l'instance qui ne sont pas visibles dans les journaux.
	
	Valeurs valides : nom de paire de clés existant enregistré auprès d'Amazon EC2

`-p platform-version`
 	
or
 	
`--platform platform-version`
	Runtime de la plateforme Elastic Beanstalk pour l'environnement. Si ce n'est pas spécifié, détecté automatiquement à partir de la version hôte de Windows Server.
	
	Exemple : `"64bit Windows Server 2016 v2.16.2 running IIS 10.0"`
	
	Pour obtenir la liste des versions de plate-forme disponibles, utilisez la liste des plateformes Web.

Nom	Description
`--remote`	Indique d'exécuter la migration en mode distant. Cette option permet l'exécution à partir d'un hôte bastion, qui se connecte au serveur cible qui contient l'application et les configurations à migrer vers Elastic Beanstalk. Exécuté depuis le serveur Bastion, `eb migrate` découvre les configurations, met en place la logique de migration sur l'hôte du bastion, puis déploie votre application dans un nouvel environnement Elastic Beanstalk. Cette option élimine le besoin d'installer l'interface de ligne de commande EB et Python sur le serveur Windows que vous devez migrer. Vous installez plutôt Python et l'EB CLI sur un hôte bastion, où vous exécutez la eb migrate commande avec l'`--remote`option. Utilisez l'`--target-ip` option pour spécifier l'hôte avec les configurations IIS à migrer. Doit être utilisé avec `--target-ip --username` , et`--password` .
`--target-ip` *`ip-address`*	Adresse IP publique de la machine Windows distante qui contient les serveurs IIS à migrer. Nécessaire lors de l'utilisation`--remote`. Ne peut être spécifié que lors de l'utilisation`--remote`.
`--username` *`username`*	Nom d'utilisateur du profil utilisateur permettant d'accéder à la machine Windows distante qui contient les serveurs IIS à migrer. Nécessaire lors de l'utilisation`--remote`. Ne peut être spécifié que lors de l'utilisation`--remote`.

Nom	Description
`--password` *password*	Mot de passe du profil utilisateur permettant d'accéder à la machine Windows distante qui contient les serveurs IIS à migrer. Nécessaire lors de l'utilisation`--remote`. Ne peut être spécifié que lors de l'utilisation`--remote`.
`-sr` *role-name* or `--service-role` *role-name*	Rôle de service IAM pour Elastic Beanstalk afin de gérer les services associés. AWS S'il n'est pas spécifié, crée un rôle de service par défaut avec les autorisations nécessaires. Pour de plus amples informations, veuillez consulter the section called “Rôle de service ”. **Note** Spécifiez uniquement le nom du rôle, pas l'ARN complet. Elastic Beanstalk crée automatiquement l'ARN complet.
`-s` *site-names* or `--sites` *site-names*	Liste des sites IIS à migrer séparés par des virgules. Si ce n'est pas spécifié, migre tous les sites disponibles sur le serveur. Exemple : `--sites "Default Web Site,Intranet,API"`

Nom	Description
`--ssl-certificates` *certificate-arn* [,*certificate-arn* ...]	Liste séparée par des virgules des certificats SSL ACM ARNs à associer à l'Application Load Balancer. Nécessaire lors de la migration de sites avec des liaisons HTTPS. Exemple : `--ssl-certificates arn:aws:acm:region:account:certificate/certificate-id`
`-t` *key1*=*value1*[,*key2*=*value2* .. or `--tags` *key1*=*value1*[,*key2*=*va*	Liste de `key=value` paires séparées par des virgules pour baliser les nouvelles ressources de votre environnement : environnement, application Elastic Beanstalk, version de l'application. Pour de plus amples informations, veuillez consulter Environnements de balisage.
`--verbose`	Afficher des informations détaillées pendant le processus de migration. Lorsqu'il est utilisé avec une explore sous-commande, affiche les détails complets de la configuration du site.

Nom	Description

`-vpc` *`config-file-or-string`*
 	
or
 	
`--vpc-config` *`config-file-or-string`*
	Configuration VPC pour l'environnement, spécifiée sous forme de chemin de fichier JSON ou de chaîne JSON.
	
	La configuration doit inclure :
	
	`{`
	`  "id": "vpc-1234567890abcdef0",`
	`  "publicip": "true|false",`
	`  "elbscheme": "public|private",`
	`  "ec2subnets": ["subnet-a1b2c3d4",`
	`"subnet-e5f6g7h8"],`
	`  "securitygroups": "sg-123456,sg-789012",`
	`  "elbsubnets": ["subnet-a1b2c3d4",`
	`"subnet-e5f6g7h8"]`
	`}`
	
	• `id`: (Obligatoire) Identifiant VPC
	• `publicip`: s'il faut attribuer du public IPs aux instances
	• `elbscheme` : schéma d'équilibrage de charge (public ou privé)
	• `ec2subnets` : liste des sous-réseaux IDs pour les instances EC2
	• `securitygroups` : groupe de sécurité séparé par des virgules IDs
	• `elbsubnets` : liste des sous-réseaux IDs pour l'équilibreur de charge
	
	Important
	La migration ignorera tous les paramètres VPC existants de l'environnement source lorsque vous spécifiez le *`--vpc-config`* paramètre. Lorsque vous utilisez ce paramètre, la migration utilise uniquement les paramètres VPC spécifiés dans

Nom	Description
	le fichier de configuration que vous transmettez. L'utilisation de ce paramètre remplace le comportement par défaut qui consiste à découvrir la configuration VPC de l'instance source ou à utiliser le VPC par défaut.
Options courantes	

Output

La commande fournit des mises à jour de statut tout au long du processus de migration :

1. Détection de configuration VPC (lors de l'exécution sur une EC2 instance)
2. Progression de la génération du bundle source pour chaque site
3. État de création de l'environnement
4. Progression du déploiement

En cas de succès, affiche les détails du nouvel environnement, notamment :

- Nom et ID de l'environnement
- Application name (Nom de l'application)
- Région
- Version de la plateforme
- Environnement CNAME

En cas de problème lors de la migration, utilisez les eb healthcommandes eb eventset pour obtenir des informations détaillées.

Exemples

Utilisation de base

Migration de base en mode interactif :

```
PS C:\migrations_workspace >  eb migrate
Identifying VPC configuration of this EC2 instance (i-0123456789abcdef0):
  id: vpc-1234567890abcdef0
  publicip: true
  elbscheme: public
  ec2subnets: subnet-123,subnet-456,subnet-789
  securitygroups: sg-123,sg-456
  elbsubnets: subnet-123,subnet-456,subnet-789

Using .\migrations\latest to contain artifacts for this migration run.
Generating source bundle for sites, applications, and virtual directories...
  Default Web Site/ -> .\migrations\latest\upload_target\DefaultWebSite.zip

Creating application version
Creating environment

Environment details for: EBMigratedEnv
  Application name: EBMigratedApp
  Region: us-west-2
  Deployed Version: app-230320_153045
  Environment ID: e-abcdef1234
  Platform: 64bit Windows Server 2019 v2.7.0 running IIS 10.0
  Tier: WebServer-Standard-1.0
  CNAME: ebmigratedenv.us-west-2.elasticbeanstalk.com
  Updated: 2023-03-20 15:30:45
```

Migration de sites spécifiques avec une configuration personnalisée :

```
PS C:\migrations_workspace >  eb migrate `
    --sites "Default Web Site,InternalAPI" `
    --application-name "CorporateApp" `
    --environment-name "Production" `
    --instance-type "c5.xlarge" `
    --tags "Environment=Production,Team=WebOps" `
    --copy-firewall-config
```

Création d'une archive de migration sans déploiement :

```
PS C:\migrations_workspace >  eb migrate --archive-only
Using .\migrations\latest to contain artifacts for this migration run.
Generating source bundle for sites, applications, and virtual directories...
  Default Web Site/ -> .\migrations\latest\upload_target\DefaultWebSite.zip
```

```
Generated destination archive directory at .\migrations\latest\upload_target
You can execute `eb init` and `eb create` from this directory to deploy to EB.
```

Exemples de configuration avancée

Migration avec configuration VPC personnalisée à l'aide d'un fichier JSON :

```
PS C:\migrations_workspace >  cat vpc-config.json
{
    "id": "vpc-1234567890abcdef0",
    "publicip": "false",
    "elbscheme": "internal",
    "ec2subnets": [
        "subnet-private1",
        "subnet-private2"
    ],
    "securitygroups": [
        "sg-app",
        "sg-database",
        "sg-monitoring"
    ],
    "elbsubnets": [
        "subnet-private1",
        "subnet-private2"
    ]
}

PS C:\migrations_workspace eb migrate `
    --sites "InternalAPI" `
    --vpc-config vpc-config.json `
    --instance-type "r5.xlarge" `
    --tags "Environment=Internal,Security=High"
```

Migration de sites avec des certificats SSL et des en-têtes d'hôte :

```
PS C:\migrations_workspace >  eb migrate `
    --sites "SecurePortal" `
    --ssl-certificates "arn:aws:acm:region:account:certificate/
cert1,arn:aws:acm:region:account:certificate/cert2" `
    --verbose
INFO: Detected HTTPS bindings:
  - www.example.com:443
```

```
  - api.example.com:443

INFO: Configuring Application Load Balancer with SSL certificates
INFO: Creating host-based routing rules:
  - www.example.com -> target group 1
  - api.example.com -> target group 2
```

Migration avec configuration instantanée EBS:fo

```
PS C:\migrations_workspace >  eb migrate `
    --sites "Default Web Site" `
    --ebs-snapshots "snap-1234567890abcdef0" "snap-0987654321fedcba1" `
    --encrypt-ebs-volumes
Using .\migrations\latest to contain artifacts for this migration run.
INFO: Enabling EBS encryption for all new volumes in us-west-2
INFO: Configuring environment with specified EBS snapshots
```

Exemples de configuration de sécurité

Gestion des sites soumis à des règles de pare-feu complexes :

```
PS C:\migrations_workspace >  eb migrate `
    --sites "Default Web Site,ReportingService" `
    --copy-firewall-config `
    --verbose
INFO: Detected the following Windows Firewall rules:
  - Allow Web Traffic (TCP 80, 443)
  - Allow Reporting Traffic (TCP 8081)
INFO: Creating corresponding security group rules
```

Migration avec des rôles IAM personnalisés :

```
PS C:\migrations_workspace >  eb migrate `
    --sites "SecureApp" `
    --instance-profile "CustomInstanceProfile" `
    --service-role "CustomServiceRole"
```

Exemples d'exécution à distance

Migration des applications IIS depuis un serveur Windows distant :

```
PS C:\migrations_workspace >  eb migrate `
    --remote `
    --target-ip "192.0.2.10" `
    --username "administrator" `
    --password "YourPassword123" `
    --application-name "RemoteApp" `
    --environment-name "RemoteEnv"
INFO: Establishing SSH connection to remote host 192.0.2.10...
INFO: Connection established
INFO: Discovering IIS sites on remote host...
INFO: Found 2 sites: Default Web Site, API
INFO: Extracting site configurations...
INFO: Generating source bundle for sites, applications, and virtual directories...
  Default Web Site/ -> .\migrations\latest\upload_target\DefaultWebSite.zip
  API/ -> .\migrations\latest\upload_target\API.zip

Creating application version
Creating environment

Environment details for: RemoteEnv
  Application name: RemoteAppstage mi
  Region: us-west-2
  Deployed Version: app-230320_153045
  Environment ID: e-abcdef1234
  Platform: 64bit Windows Server 2019 v2.7.0 running IIS 10.0
  Tier: WebServer-Standard-1.0
  CNAME: remoteenv.us-west-2.elasticbeanstalk.com
  Updated: 2023-03-20 15:30:45
```

Migration à distance avec sélection de sites spécifiques :

```
PS C:\migrations_workspace >  eb migrate `
    --remote `
    --target-ip "192.0.2.10" `
    --username "administrator" `
    --password "YourPassword123" `
    --sites "API" `
    --instance-type "c5.large"
```

eb open

Description

Ouvre l'URL publique de votre site web dans le navigateur par défaut.

Syntaxe

eb open

eb open ***environment-name***

Options

Nom	Description
Options courantes	

Output

La commande eb open n'a pas de sortie. Au lieu de cela, elle ouvre l'application dans une fenêtre de navigateur.

eb platform

Description

Cette commande prend en charge deux espaces de travail différents :

Plateforme

Utilisez cet espace de travail pour gérer les plateformes personnalisées.

Environnement

Utilisez cet espace de travail pour sélectionner une plateforme par défaut ou afficher des informations sur la plateforme actuelle.

Elastic Beanstalk fournit le raccourci ebp pour eb platform.

> **Note**
>
> Windows PowerShell l'utilise ebp comme alias de commande. Si vous exécutez l'interface de ligne de commande EB sous Windows PowerShell, utilisez la forme longue de cette commande —eb platform.

Utilisation de la plateforme eb pour les plateformes personnalisées

Répertorie les versions de la plateforme actuelle et vous permet de gérer des plateformes personnalisées.

Syntaxe

eb platform create [***version***] [***options***]

eb platform delete [***version***] [***options***]

eb platform events [***version***] [***options***]

eb platform init [***platform***] [***options***]

eb platform list [***options***]

eb platform logs [***version***] [***options***]

eb platform status [***version***] [***options***]

eb platform use [***platform***] [***options***]

Options

Nom	Description
`create [version] [options]`	Créer une nouvelle version de la plateforme. En savoir plus.
`delete version [options]`	Supprimer une version de plateforme. En savoir plus.

Nom	Description
`events [version] [options]`	Afficher les événements issus de la version de plateforme. En savoir plus.
`init [platform] [options]`	Initialiser un référentiel de plateforme. En savoir plus.
`list [options]`	Répertorier les versions de la plateforme actuelle. En savoir plus.
`logs [version] [options]`	Afficher les journaux de l'environnement de génération pour une version de plateforme. En savoir plus.
`status [version] [options]`	Afficher le statut de la version de plateforme. En savoir plus.
`use [platform] [options]`	Choisir une autre plateforme à partir de laquelle créer de nouvelles versions. En savoir plus.
Options courantes	

Options courantes

Toutes les commandes eb platform contiennent les options courantes suivantes.

Nom	Description

`-h`
OR
`--help`
	Affiche un message d'aide, puis se ferme.

`--debug`	Affiche une sortie de débogage supplémentaire.
`--quiet`	Supprime toutes les sorties.
`-v`	Affiche une sortie supplémentaire.

Nom	Description
OR `--verbose`	
`--profile` *`PROFILE`*	Utilise les informations spécifiées *`PROFILE`* dans vos informati ons d'identification.
`-r` *`REGION`* OU `--region` *`REGION`*	Utilisez la région*`REGION`*.
`--no-verify-ssl`	Ne vérifiez pas les certificats AWS SSL.

Eb platform create

Crée une nouvelle version de la plateforme et renvoie l'ARN de la nouvelle version. Si aucun environnement de génération n'est en cours d'exécution dans la région actuelle, cette commande en lance un. Les options *`version`* et incrément (`-M`-`m`, et`-p`) s'excluent mutuellement.

Options

Nom	Description
`version`	Si *`version`* ce n'est pas spécifié, crée une nouvelle version basée sur la plateforme la plus récente avec la version du correctif (N dans n.n.N) incrémentée.

`-M`
 	
OU
 	
`--major-increment`
	Incrémente le numéro de version majeure (N dans N.n.n).

`-m` OR	Incrémente le numéro de version mineure (N dans n.N.n).

Nom	Description
`--minor-increment`	

`-p`
 	
OR
 	
`--patch-increment`
	Incrémente le numéro de version du correctif (N dans n.n.N).

`-i` *`INSTANCE_TYPE`* OU --type d'instance *`INSTANCE_TYPE`*	À utiliser *`INSTANCE_TYPE`* comme type d'instance, par exemple**`t1.micro`**.
`-ip` *`INSTANCE_PROFILE`* OU `--instance-profile` *`INSTANCE_PROFILE`*	*`INSTANCE_PROFILE`* À utiliser comme profil d'instance lors de la création AMIs d'une plateforme personnalisée. Si l'option `-ip` n'est pas spécifiée, crée le profil d'instance `aws-elasticbeanstalk-custom-platforme-ec2-role` et l'utilise pour la plateforme personnalisée.
`--tags` *`key1`*`=`*`value1`*`[,`*`ke`	Les balises sont spécifiées sous la forme d'une liste séparée par des virgules et composée de paires `key=value` .
`--timeout` *`minutes`*	Définissez le nombre de minutes avant que la commande expire.
`--vpc.id` *`VPC_ID`*	ID du VPC dans lequel Packer effectue la création.
`--vpc.subnets` *`VPC_SUBNETS`*	Sous-réseaux VPC dans lesquels Packer effectue la création.
`--vpc.publicip`	Associe IPs le public aux EC2 instances lancées.

Eb platform delete

Supprimer une version de plateforme. La version n'est pas supprimée si un environnement l'utilise.

Options

Nom	Description
version	La version à supprimer. Cette valeur est obligatoire.
`--cleanup`	Supprimer toutes les versions de plateforme à l'état `Failed`.
`--all-platforms`	Si `--cleanup` est spécifié, supprimer toutes les versions de plateforme à l'état `Failed` pour toutes les plateformes.
`--force`	Ne pas demander de confirmation lors de la suppression d'une version.

Eb platform events

Afficher les événements issus de la version de plateforme. Si *version* tel est le cas, affichez les événements de cette version, sinon affichez les événements de la version actuelle.

Options

Nom	Description
version	La version pour laquelle des événements sont affichés. Cette valeur est obligatoire.

`-f`
OR
`--follow`
	Continuer à afficher les événements au fur et à mesure qu'ils se produisent.

Eb platform init

Initialiser un référentiel de plateforme.

Options

Nom	Description
platform	Le nom de la plateforme à initialiser. Cette valeur est obligatoire, sauf si le mode interactif `-i` est activé.

`-i`
OR
`--interactive`
	Utiliser le mode interactif.

`-k` *KEYNAME* OU `--keyname` *KEYNAME*	Le nom de EC2 clé par défaut.

Vous pouvez exécuter cette commande dans un répertoire qui a été initialisé auparavant, mais dans ce cas, vous ne pouvez pas modifier le type de l'espace de travail.

Pour relancer l'initialisation aves d'autres options, utilisez l'option `-i`.

Eb platform list

Répertorier les versions de la plateforme associées à un espace de travail (répertoire) ou à une région.

La commande renvoie des résultats différents en fonction du type d'espace de travail dans lequel vous l'exécutez, comme suit :

- Dans un espace de travail de plateforme (un répertoire initialisé par `eb platform init`), la commande renvoie une liste de toutes les versions de plateforme de la plateforme personnalisée définie dans l'espace de travail. Ajoutez l'option `--verbose` ou `--all-platforms` pour obtenir une liste de toutes les versions de plateforme de toutes les plateformes personnalisées que votre compte possède dans la région associée à l'espace de travail.
- Dans un espace de travail d'application (un répertoire initialisé par `eb init`), la commande renvoie une liste de toutes les versions de plateforme, à la fois pour les plateformes prises en

charge par Elastic Beanstalk et pour les plateformes personnalisées de votre compte. La liste utilise des noms de version de plateforme courts, et certaines variantes de version de plateforme peuvent être associées. Ajoutez l'option `--verbose` pour obtenir une liste détaillée avec les noms complets et toutes les variantes répertoriées séparément.

- Dans un répertoire non initialisé, la commande fonctionne uniquement avec l'option `--region`. Elle renvoie une liste de toutes les versions de plateforme gérées par Elastic Beanstalk prises en charge dans la région. La liste utilise des noms de version de plateforme courts, et certaines variantes de version de plateforme peuvent être associées. Ajoutez l'option `--verbose` pour obtenir une liste détaillée avec les noms complets et toutes les variantes répertoriées séparément.

Options

Nom	Description

`-a`
 	
OR
 	
`--all-platforms`
	Valide uniquement dans un espace de travail initialisé (un répertoire initialisé par `eb platform init` ou `eb init`). Répertorie les versions de plateforme de toutes les plateformes personnalisées associées à votre compte.

`-s` *`STATUS`* OU `--status` *`STATUS`*	Répertoriez uniquement les plateformes correspondantes *`STATUS`* : • Prêt • Échec • Suppression en cours • Création

Eb platform logs

Afficher les journaux de l'environnement de génération pour une version de plateforme.

Options

Nom	Description
version	La version de la plateforme pour laquelle des journaux sont affichés. Si cette option est omise, afficher les journaux de la version actuelle.
--stream	Diffusez les journaux de déploiement configurés avec CloudWatch.

Eb platform status

Afficher le statut de la version de plateforme.

Options

Nom	Description
version	La version de la plateforme pour laquelle le statut est extrait. Si cette option est omise, afficher le statut de la version actuelle.

Eb platform use

Choisir une autre plateforme à partir de laquelle créer de nouvelles versions.

Options

Nom	Description
platform	Spécifie *platform* comme version active pour cet espace de travail. Cette valeur est obligatoire.

Utilisation de la plateforme eb pour les environnements

Répertorie les plateformes prises en charge et vous permet de définir la plate-forme par défaut et la version de la plateforme à utiliser quand vous lancez un environnement. Utilisez eb platform list pour

afficher une liste de toutes les plateformes prises en charge. Utilisez eb platform select pour modifier la plateforme pour votre projet. Utilisez eb platform show pour afficher la plateforme sélectionnée de votre projet.

Syntaxe

eb platform list

eb platform select

eb platform show

Options

Nom	Description
`list`	Répertorier la version de la plateforme actuelle.
`select`	Sélectionner la plateforme par défaut.
`show`	Afficher des informations sur la plateforme actuelle.

Exemple 1

L'exemple suivant répertorie les noms de toutes les configurations pour toutes les plateformes qu'Elastic Beanstalk prend en charge.

```
$ eb platform list
docker-1.5.0
glassfish-4.0-java-7-(preconfigured-docker)
glassfish-4.1-java-8-(preconfigured-docker)
go-1.3-(preconfigured-docker)
go-1.4-(preconfigured-docker)
iis-7.5
iis-8
iis-8.5
multi-container-docker-1.3.3-(generic)
node.js
php-5.3
php-5.4
```

```
php-5.5
python
python-2.7
python-3.4
python-3.4-(preconfigured-docker)
ruby-1.9.3
ruby-2.0-(passenger-standalone)
ruby-2.0-(puma)
ruby-2.1-(passenger-standalone)
ruby-2.1-(puma)
ruby-2.2-(passenger-standalone)
ruby-2.2-(puma)
tomcat-6
tomcat-7
tomcat-7-java-6
tomcat-7-java-7
tomcat-8-java-8
```

Exemple 2

L'exemple suivant vous invite à choisir parmi une liste de plateformes, ainsi que la version que vous souhaitez déployer pour la plateforme spécifiée.

```
$ eb platform select
Select a platform.
1) PHP
2) Node.js
3) IIS
4) Tomcat
5) Python
6) Ruby
7) Docker
8) Multi-container Docker
9) GlassFish
10) Go
(default is 1): 5

Select a platform version.
1) Python 2.7
2) Python
3) Python 3.4 (Preconfigured - Docker)
```

Exemple 3

L'exemple suivant présente des informations sur la plateforme par défaut actuelle.

```
$ eb platform show
Current default platform: Python 2.7
New environments will be running:  64bit Amazon Linux 2014.09 v1.2.0 running Python 2.7

Platform info for environment "tmp-dev":
Current: 64bit Amazon Linux 2014.09 v1.2.0 running Python
Latest:  64bit Amazon Linux 2014.09 v1.2.0 running Python
```

eb printenv

Description

Affiche toutes les propriétés de l'environnement dans la fenêtre de commande.

Syntaxe

eb printenv

eb printenv ***environment-name***

Options

Nom	Description
Options courantes	

Output

En cas de réussite, la commande renvoie l'état de l'opération `printenv`.

Exemple

L'exemple suivant affiche les propriétés de l'environnement pour l'environnement spécifié.

```
$ eb printenv
Environment Variables:
     PARAM1 = Value1
```

eb restore

Description

Reconstruit un environnement suspendu, en créant un nouvel environnement avec le même nom, le même ID et la même configuration. Le nom de l'environnement, le nom de domaine et la version de l'application doivent être disponibles pour que la reconstruction réussisse.

Syntaxe

eb restore

eb restore ***environment_id***

Options

Nom	Description
Options courantes	

Output

L'interface de ligne de commande (CLI) EB affiche une liste d'environnements suspendus qui sont disponibles pour être restaurés.

Exemple

```
$ eb restore
Select a terminated environment to restore

  #   Name          ID             Application Version      Date Terminated        Ago

  3   gamma         e-s7mimej8e9   app-77e3-161213_211138   2016/12/14 20:32 PST   13
 mins
```

```
  2   beta          e-sj28uu2wia   app-77e3-161213_211125   2016/12/14 20:32 PST   13
 mins
  1   alpha         e-gia8mphu6q   app-77e3-161213_211109   2016/12/14 16:21 PST   4
 hours

 (Commands: Quit, Restore, # #)

Selected environment alpha
Application:    scorekeep
Description:    Environment created from the EB CLI using "eb create"
CNAME:          alpha.h23tbtbm92.us-east-2.elasticbeanstalk.com
Version:        app-77e3-161213_211109
Platform:       64bit Amazon Linux 2016.03 v2.1.6 running Java 8
Terminated:     2016/12/14 16:21 PST
Restore this environment? [y/n]: y

2018-07-11 21:04:20    INFO: restoreEnvironment is starting.
2018-07-11 21:04:39    INFO: Created security group named: sg-e2443f72
...
```

eb scale

Description

Adapte l'environnement pour qu'il s'exécute toujours sur un certain nombre d'instances, en définissant à la fois les nombres minimal et maximal d'instances sur le nombre spécifié.

Syntaxe

eb scale ***number-of-instances***

eb scale ***number-of-instances environment-name***

Options

Nom	Description
--timeout	Le nombre de minutes avant que la commande expire.
Options courantes	

Output

En cas de réussite, la commande met à jour les nombres minimal et maximal d'instances à exécuter sur le nombre spécifié.

Exemple

L'exemple suivant définit le nombre d'instances sur 2.

```
$ eb scale 2
2018-07-11 21:05:22    INFO: Environment update is starting.
2018-07-11 21:05:27    INFO: Updating environment tmp-dev's configuration settings.
2018-07-11 21:08:53    INFO: Added EC2 instance 'i-5fce3d53' to Auto Scaling Group
 'awseb-e-2cpfjbra9a-stack-AWSEBAutoScalingGroup-7AXY7U13ZQ6E'.
2018-07-11 21:08:58    INFO: Successfully deployed new configuration to environment.
2018-07-11 21:08:59    INFO: Environment update completed successfully.
```

eb setenv

Description

Définit les propriétés d'environnement pour l'environnement par défaut.

Syntaxe

eb setenv ***key*=*value***

Vous pouvez inclure autant de propriétés que vous voulez, mais la taille totale de toutes les propriétés ne peut pas dépasser 4 096 octets. Vous pouvez supprimer une variable en laissant la valeur vierge. Pour connaître les limites, consultez Configuration des propriétés de l'environnement (variables d'environnement).

> **Note**
>
> Si `value` contient un caractère spécial\, vous devez l'échapper en le faisant précéder d'un caractère .

Options

Nom	Description
--timeout	Le nombre de minutes avant que la commande expire.
Options courantes	

Output

En cas de réussite, la commande affiche que la mise à jour de l'environnement a abouti.

Exemple

L'exemple suivant définit la variable d'environnement ExampleVar.

```
$ eb setenv ExampleVar=ExampleValue
2018-07-11 21:05:25    INFO: Environment update is starting.
2018-07-11 21:05:29    INFO: Updating environment tmp-dev's configuration settings.
2018-07-11 21:06:50    INFO: Successfully deployed new configuration to environment.
2018-07-11 21:06:51    INFO: Environment update completed successfully.
```

La commande suivante définit plusieurs propriétés d'environnement. Elle ajoute la propriété de l'environnement nommée `foo` et définit sa valeur sur `bar`, modifie la valeur de la propriété `JDBC_CONNECTION_STRING` et supprime les propriétés `PARAM4` et `PARAM5`.

```
$ eb setenv foo=bar JDBC_CONNECTION_STRING=hello PARAM4= PARAM5=
```

eb ssh

Description

> **Note**
>
> Cette commande ne fonctionne pas avec des environnements exécutant des instances Windows Server.

Connectez-vous à une EC2 instance Amazon Linux dans votre environnement à l'aide de Secure Shell (SSH). Si un environnement comporte plusieurs instances en cours d'exécution, l'interface de ligne de commande (CLI) EB vous invite à spécifier à quelle instance vous souhaitez vous connecter. Pour utiliser cette commande, SSH doit être installé sur votre ordinateur local et disponible à partir de la ligne de commande. Les fichiers de clé privée doivent se trouver dans un dossier nommé `.ssh` sous votre répertoire utilisateur, et les EC2 instances de votre environnement doivent avoir des adresses IP publiques.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande se connecte également à des instances dans l'environnement personnalisé.

> **Clés SSH**
>
> Si vous n'avez pas déjà configuré SSH, vous pouvez utiliser l'interface de ligne de commande (CLI) EB pour créer une clé lorsque vous exécutez eb init. Si vous avez déjà exécuté eb init, exécutez-le à nouveau avec l'option `--interactive` et sélectionnez Yes (Oui) et Create New Keypair (Créer une nouvelle paire de clés) lorsque vous êtes invité à configurer SSH. Les clés créées au cours de ce processus sont stockées dans le dossier approprié par l'interface de ligne de commande (CLI) EB.

Cette commande ouvre temporairement le port 22 dans le groupe de sécurité de votre environnement pour le trafic entrant à partir de 0.0.0.0/0 (toutes les adresses IP) si aucune règle pour le port 22 n'est déjà en place. Si vous avez configuré le groupe de sécurité de votre environnement pour ouvrir le port 22 sur une plage CIDR limitée pour une sécurité accrue, l'interface de ligne de commande (CLI) EB respectera ce paramètre et renoncez à toutes modifications apportées au groupe de sécurité. Pour substituer ce comportement et forcer l'interface de ligne de commande (CLI) EB à ouvrir le port 22 à tout trafic entrant, utilisez l'option `--force`.

Consultez EC2 groupes de sécurité pour obtenir des informations sur la configuration du groupe de sécurité de votre environnement.

Syntaxe

eb ssh

eb ssh ***environment-name***

Options

Nom	Description

`-i`
 	
ou
 	
`--instance`
	Spécifie l'ID d'instance de l'instance à laquelle vous vous connectez. Nous vous recommandons d'utiliser cette option.

`-n`
 	
ou
 	
`--number`
	Spécifiez l'instance à laquelle se connecter à l'aide d'un chiffre.

`-o`
 	
ou
 	
`--keep_open`
	Laissez le port 22 ouvert sur le groupe de sécurité à la fin de la session SSH.

`--command`	Exécutez une commande shell sur l'instance spécifiée au lieu de démarrer une session SSH.
`--custom`	Spécifiez une commande SSH à utiliser au lieu de « ssh -i keyfile ». N'incluez pas d'utilisateur distant ni de nom d'hôte.
`--setup`	Modifiez la paire de clés attribuée aux instances de l'environ nement (requiert le remplacement des instances).
`--force`	Ouvrez le port 22 au trafic entrant à partir de 0.0.0.0/0 dans le groupe de sécurité de l'environnement, même si le groupe de sécurité est déjà configuré pour SSH. Utilisez cette option si le groupe de sécurité de votre environne ment est configuré pour ouvrir le port 22 à une plage CIDR limitée qui n'inclut pas l'adresse IP à partir de laquelle vous essayez de vous connecter.

Nom	Description
`--timeout` *minutes*	Définissez le nombre de minutes avant que la commande expire. Peut être utilisé uniquement avec l'argument `--setup`.
Options courantes	

Output

En cas de réussite, la commande ouvre une connexion SSH sur l'instance.

Exemple

L'exemple suivant vous connecte à l'environnement spécifié.

```
$ eb ssh
Select an instance to ssh into
1) i-96133799
2) i-5931e053
(default is 1): 1
INFO: Attempting to open port 22.
INFO: SSH port 22 open.
The authenticity of host '54.191.45.125 (54.191.45.125)' can't be established.
RSA key fingerprint is ee:69:62:df:90:f7:63:af:52:7c:80:60:1b:3b:51:a9.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '54.191.45.125' (RSA) to the list of known hosts.

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2014.09-release-notes/
No packages needed for security; 1 packages available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-8-185 ~]$ ls
[ec2-user@ip-172-31-8-185 ~]$ exit
logout
Connection to 54.191.45.125 closed.
INFO: Closed port 22 on ec2 instance security group
```

eb status

Description

Fournit des informations sur l'état de l'environnement.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande fournit également des informations sur l'environnement de génération.

Syntaxe

eb status

eb status ***environment-name***

Options

Nom	Description

`-v`
ou
`--verbose`
	Fournit plus d'informations sur les instances individuelles, telles que leur statut avec l'équilibreur de charge Elastic Load Balancing.

Options courantes	

Output

En cas de réussite, la commande renvoie les informations suivantes sur l'environnement :

- Nom de l'environnement
- Nom de l'application
- Version de l'application déployée
- ID de l'environnement
- Plateforme

- Niveau de l'environnement
- CNAME
- Heure de dernière mise à jour de l'environnement
- État
- Santé

Si vous utilisez le mode détaillé, l'interface de ligne de commande EB vous indique également le nombre d'instances Amazon EC2 en cours d'exécution.

Exemple

L'exemple suivant affiche l'état du tmp-dev de l'environnement.

```
$ eb status
Environment details for: tmp-dev
  Application name: tmp
  Region: us-west-2
  Deployed Version: None
  Environment ID: e-2cpfjbra9a
  Platform: 64bit Amazon Linux 2014.09 v1.0.9 running PHP 5.5
  Tier: WebServer-Standard-1.0
  CNAME: tmp-dev.elasticbeanstalk.com
  Updated: 2014-10-29 21:37:19.050000+00:00
  Status: Launching
  Health: Grey
```

eb swap

Description

Echange le CNAME de l'environnement avec le CNAME d'un autre environnement (par exemple, pour éviter des temps d'arrêt lorsque vous mettez à jour la version de votre application).

> **Note**
>
> Si vous avez plus de deux environnements, vous êtes invité à sélectionner dans une liste d'environnements le nom de l'environnement qui utilise actuellement le CNAME que vous

souhaitez. Pour supprimer cela, vous pouvez spécifier le nom de l'environnement à utiliser en incluant l'option `-n` lorsque vous exécutez la commande.

Syntaxe

eb swap

eb swap ***environment-name***

> **Note**
>
> *`environment-name`*Il s'agit de l'environnement pour lequel vous souhaitez un CNAME différent. Si vous ne le spécifiez pas *`environment-name`* en tant que paramètre de ligne de commande lors de l'exécutioneb swap, l'interface de ligne de commande EB met à jour le CNAME de l'environnement par défaut.

Options

Nom	Description

`-n`
 	
or
 	
`--destination_name`
	Spécifie le nom de l'environnement avec lequel vous souhaitez échanger CNAMEs. Si vous exécutez eb swap sans cette option, alors l'interface de ligne de commande (CLI) EB vous invite à choisir dans une liste de vos environnements.

Options courantes	

Output

En cas de réussite, la commande renvoie l'état de l'opération `swap`.

Exemples

L'exemple suivant échange le tmp-dev de l'environnement avec live-env.

```
$ eb swap
Select an environment to swap with.
1) staging-dev
2) live-env
(default is 1): 2
2018-07-11 21:05:25    INFO: swapEnvironmentCNAMEs is starting.
2018-07-11 21:05:26    INFO: Swapping CNAMEs for environments 'tmp-dev' and 'live-env'.
2018-07-11 21:05:30    INFO: 'tmp-dev.elasticbeanstalk.com' now points to 'awseb-e-j-
AWSEBLoa-M7U21VXNLWHN-487871449.us-west-2.elb.amazonaws.com'.
2018-07-11 21:05:30    INFO: Completed swapping CNAMEs for environments 'tmp-dev' and
 'live-env'.
```

L'exemple suivant échange le tmp-dev de l'environnement avec le live-env de l'environnement mais ne vous invite pas à entrer ou à sélectionner une valeur pour aucun paramètre.

```
$ eb swap tmp-dev --destination_name live-env
2018-07-11 21:18:12    INFO: swapEnvironmentCNAMEs is starting.
2018-07-11 21:18:13    INFO: Swapping CNAMEs for environments 'tmp-dev' and 'live-env'.
2018-07-11 21:18:17    INFO: 'tmp-dev.elasticbeanstalk.com' now points to 'awseb-e-j-
AWSEBLoa-M7U21VXNLWHN-487871449.us-west-2.elb.amazonaws.com'.
2018-07-11 21:18:17    INFO: Completed swapping CNAMEs for environments 'tmp-dev' and
 'live-env'.
```

eb tags

Description

Ajoutez, supprimez, mettez à jour et listez des balises d'une ressource Elastic Beanstalk.

Pour de plus amples informations sur le balisage des ressources dans Elastic Beanstalk, veuillez consulter Étiquette des ressources d'application Elastic Beanstalk.

Syntaxe

eb tags [***environment-name***] [--resource ***ARN***] -l | --list

eb tags [***environment-name***] [--resource ***ARN***] -a | --add ***key1***=***value1***[,***key2***=***value2*** ...]

eb tags [***environment-name***] [--resource ***ARN***] -u | --update ***key1***=***value1***[,***key2***=***value2*** ...]

eb tags [*environment-name*] [--resource *ARN*] -d | --delete *key1*[,*key2* ...]

Vous pouvez combiner les options de sous-commande `--add`, `--update` et `--delete` en une seule commande. Au moins l'une d'elles est obligatoire. Vous ne pouvez pas combiner n'importe laquelle des trois options de sous-commande avec `--list`.

Sans tous les arguments supplémentaires, tous ces commandes répertorient ou modifient les balises de l'environnement par défaut dans le répertoire actuel de l'application. Avec un *environment-name* argument, les commandes répertorient ou modifient les balises de cet environnement. Avec l'option `--resource`, les commandes listent ou modifient les balises de n'importe quelle ressource Elastic Beanstalk – une application, un environnement, une version d'application, une configuration enregistrée, ou une version de plateforme personnalisée. Spécifiez la ressource par son Amazon Resource Name (ARN).

Options

Aucune de ces options n'est obligatoire. Si vous exécutez eb create sans aucune option, vous êtes invité à saisir ou à sélectionner une valeur pour chaque paramètre.

Nom	Description

`-l`
ou
`--list`
	Liste tous les balises qui sont actuellement appliquées à la ressource.

`-a` *key1*`=`*value1*`[,`*key2*`=`*value2* ou `--add` *key1*`=`*value1*`[,`*key2*`=`*val*	Appliquez de nouvelles balises à la ressource. Spécifiez les balises sous la forme d'une liste séparée par des virgules et composée de paires `key=value` . Vous ne pouvez pas spécifier les clés de balises existantes. Valeurs valides : consultez Balisage de ressources .
`-u` *key1*`=`*value1*`[,`*key2*`=`*value2* ou `--updat e` *key1*`=`*value1*`[,`*key2*`=`*value2* .	Mettez à jour les valeurs des balises de ressources existantes. Spécifiez les balises sous la forme d'une liste séparée par des virgules et composée de paires `key=value` . Vous devez spécifier les clés de balises existantes.

Nom	Description
	Valeurs valides : consultez Balisage de ressources .

`-d key1[,key2 ...]`
 	
ou
 	
`--delete key1[,key2 ...]`
	Supprimez des balises de ressources existantes. Spécifiez les balises sous la forme d'une liste de clés séparées par des virgules. Vous devez spécifier les clés de balises existantes.
	
	Valeurs valides : consultez Balisage de ressources .

`-r region`
 	
or
 	
`--region region`
	L' Région AWS endroit dans lequel se trouve votre ressource.
	
	Par défaut : la région configuré par défaut.
	
	Pour obtenir la liste des valeurs que vous pouvez spécifier pour cette option, consultez Points de terminaison et quotas AWS Elastic Beanstalk dans Références générales AWS.

`--resource ARN`	L'ARN de la ressource que la commande modifie ou pour laquelle elle répertorie les balises. Si elle n'est pas spécifiée, la commande fait référence à l'environnement spécifié (par défaut) de l'application du répertoire actuel. Valeurs valides : voir l'une des sous-rubriques Balisage de ressources qui est spécifique à la ressource qui vous intéresse. Ces rubriques montrent comment l'ARN de la ressource est construit et expliquent comment obtenir une liste de ces ressources ARNs qui existent pour votre application ou votre compte.

Output

L'option de sous-commande `--list` affiche la liste des balises de la ressource. La sortie montre à la fois les balises qu'Elastic Beanstalk applique par défaut et vos balises personnalisées.

```
$ eb tags --list
```

```
Showing tags for environment 'MyApp-env':

Key                                 Value

Name                                MyApp-env
elasticbeanstalk:environment-id     e-63cmxwjaut
elasticbeanstalk:environment-name   MyApp-env
mytag                               tagvalue
tag2                                2nd value
```

Les options de sous-commande `--add`, `--update` et `--delete`, lorsqu'elles aboutissent, n'ont pas de sortie. Vous pouvez ajouter l'option `--verbose` pour voir la sortie détaillée de l'activité de la commande.

```
$ eb tags --verbose --update "mytag=tag value"
Updated Tags:

Key                                 Value

mytag                               tag value
```

Exemples

La commande suivante ajoute une balise avec succès avec la clé `tag1` et la valeur `value1` à l'environnement par défaut de l'application, et, dans le même temps, supprime la balise `tag2`.

```
$ eb tags --add tag1=value1 --delete tag2
```

La commande suivante ajoute une balise avec succès à une configuration enregistrée au sein d'une application.

```
$ eb tags --add tag1=value1 \
      --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-
id:configurationtemplate/my-app/my-template"
```

La commande suivante échoue, car elle essaie de mettre à jour une balise inexistante.

```
$ eb tags --update tag3=newval
ERROR: Tags with the following keys can't be updated because they don't exist:
```

```
  tag3
```

La commande suivante échoue, car elle essaie de mettre à jour et de supprimer la même clé.

```
$ eb tags --update mytag=newval --delete mytag
ERROR: A tag with the key 'mytag' is specified for both '--delete' and '--update'. Each
 tag can be either deleted or updated in a single operation.
```

eb terminate

Description

Met fin à l'environnement en cours d'exécution afin que vous n'ayez pas à payer de frais pour les ressources non utilisées AWS .

Via l'option `--all`, supprime l'application sur laquelle le répertoire actuel a été initialisé à l'aide de eb init. La commande met fin à tous les environnements de l'application. Elle résilie également les versions d'application et les configurations enregistrées de l'application, puis supprime l'application.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande met hors service l'environnement personnalisé en cours d'exécution.

> **Note**
>
> Vous pouvez toujours lancer un nouvel environnement en utilisant la même version ultérieurement.

Si vous possédez des données d'un environnement que vous souhaitez conserver, définissez la stratégie de suppression de base de données sur `Retain` avant de résilier l'environnement. Cela permet de maintenir la base de données opérationnelle en dehors d'Elastic Beanstalk. Tous les environnements Elastic Beanstalk doivent ensuite s'y connecter en tant que base de données externe. Si vous souhaitez sauvegarder les données sans maintenir la base de données opérationnelle, définissez la stratégie de suppression pour qu'elle prenne un instantané de la base de données avant de résilier l'environnement. Pour de plus amples informations, consultez Cycle de vie de base de données dans le chapitre Configuration des environnements de ce guide.

> **Important**
>
> Si vous résiliez un environnement, vous devez également supprimer tous les mappages CNAME que vous avez créés, car d'autres clients peuvent réutiliser un nom d'hôte disponible. Veillez à supprimer les enregistrements DNS qui pointent vers votre environnement résilié, afin d'éviter toute entrée DNS en suspens. Une entrée DNS bloquée peut exposer le trafic Internet destiné à votre domaine à des failles de sécurité. Elle peut également présenter d'autres risques.
> Pour de plus amples informations, veuillez consulter Protection contre les enregistrements de délégation suspendus dans Route 53 dans le manuel du développeur Amazon Route 53. Vous pouvez également en savoir plus sur le blocage des entrées DNS dans Enhanced Domain Protections for Amazon CloudFront Requests sur le blog sur la AWS sécurité.

Syntaxe

eb terminate

eb terminate ***environment-name***

Options

Nom	Description
`--all`	Met fin à tous les environnements dans l'application, suspend les versions d'application et les configurations enregistrées de l'application, puis supprime l'application.
`--force`	Résilie l'environnement sans demande de confirmation.
`--ignore-links`	Résilie l'environnement même s'il existe des environnements dépendants qui y sont liés. Consultez Compose Environments.
`--timeout`	Le nombre de minutes avant que la commande expire.

Output

En cas de réussite, la commande renvoie l'état de l'opération `terminate`.

Exemple

L'exemple de demande suivant résilie le tmp-dev DE l'environnement.

```
$ eb terminate
The environment "tmp-dev" and all associated instances will be terminated.
To confirm, type the environment name: tmp-dev
2018-07-11 21:05:25    INFO: terminateEnvironment is starting.
2018-07-11 21:05:40    INFO: Deleted CloudWatch alarm named: awseb-e-2cpfjbra9a-stack-
AWSEBCloudwatchAlarmHigh-16V08YOF2KQ7U
2018-07-11 21:05:41    INFO: Deleted CloudWatch alarm named: awseb-e-2cpfjbra9a-stack-
AWSEBCloudwatchAlarmLow-6ZAWH9F20P7C
2018-07-11 21:06:42    INFO: Deleted Auto Scaling group policy named:
 arn:aws:autoscaling:us-east-2:11122223333:scalingPolicy:5d7d3e6b-
d59b-47c5-b102-3e11fe3047be:autoScalingGroupName/awseb-e-2cpfjbra9a-stack-
AWSEBAutoScalingGroup-7AXY7U13ZQ6E:policyName/awseb-e-2cpfjbra9a-stack-AWSEBAutoSca
lingScaleUpPolicy-1876U27JEC34J
2018-07-11 21:06:43    INFO: Deleted Auto Scaling group policy named:
 arn:aws:autoscaling:us-east-2:11122223333:scalingPolicy:29c6e7c7-7ac8-46fc-91f5-
cfabb65b985b:autoScalingGroupName/awseb-e-2cpfjbra9a-stack-
AWSEBAutoScalingGroup-7AXY7U13ZQ6E:policyName/awseb-e-2cpfjbra9a-stack-AWSEBAutoSca
lingScaleDownPolicy-SL4LHODMOMU
2018-07-11 21:06:48    INFO: Waiting for EC2 instances to terminate. This may take a
 few minutes.
2018-07-11 21:08:55    INFO: Deleted Auto Scaling group named: awseb-e-2cpfjbra9a-
stack-AWSEBAutoScalingGroup-7AXY7U13ZQ6E
2018-07-11 21:09:10    INFO: Deleted security group named: awseb-e-2cpfjbra9a-stack-
AWSEBSecurityGroup-XT4YYGFL7I99
2018-07-11 21:09:40    INFO: Deleted load balancer named: awseb-e-2-AWSEBLoa-
AK6RRYFQVV3S
2018-07-11 21:09:42    INFO: Deleting SNS topic for environment tmp-dev.
2018-07-11 21:09:52    INFO: terminateEnvironment completed successfully.
```

eb upgrade

Description

Met à niveau la plateforme de votre environnement vers la version la plus récente de la plateforme sur laquelle il est actuellement en cours d'exécution.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande met à niveau l'environnement sur la version la plus récente de la plateforme personnalisée sur laquelle il est actuellement en cours d'exécution.

Syntaxe

eb upgrade

eb upgrade ***environment-name***

Options

Nom	Description
`--force`	Met à niveau sans requérir que vous confirmiez le nom de l'environnement avant de commencer le processus de mise à niveau.
`--noroll`	Met à jour toutes les instances sans utiliser de mises à jour propagées pour maintenir certaines instances en service lors de la mise à niveau.
Options courantes	

Output

La commande offre une présentation de la modification et vous invite à confirmer la mise à niveau en tapant le nom de l'environnement. En cas de réussite, votre environnement est mis à jour, puis lancé avec la version la plus récente de la plateforme.

Exemple

L'exemple suivant met à niveau la version actuelle de la plateforme de l'environnement spécifié vers la version de la plateforme la plus récente.

```
$ eb upgrade
Current platform: 64bit Amazon Linux 2014.09 v1.0.9 running Python 2.7
Latest platform:  64bit Amazon Linux 2014.09 v1.2.0 running Python 2.7
```

```
WARNING: This operation replaces your instances with minimal or zero downtime. You may
 cancel the upgrade after it has started by typing "eb abort".
You can also change your platform version by typing "eb clone" and then "eb swap".

To continue, type the environment name:
```

eb use

Description

Définit l'environnement spécifié en tant qu'environnement par défaut.

Lors de l'utilisation de Git, eb use définit l'environnement par défaut pour la branche actuelle. Exécutez cette commande une fois dans chaque branche que vous souhaitez déployer sur Elastic Beanstalk.

Syntaxe

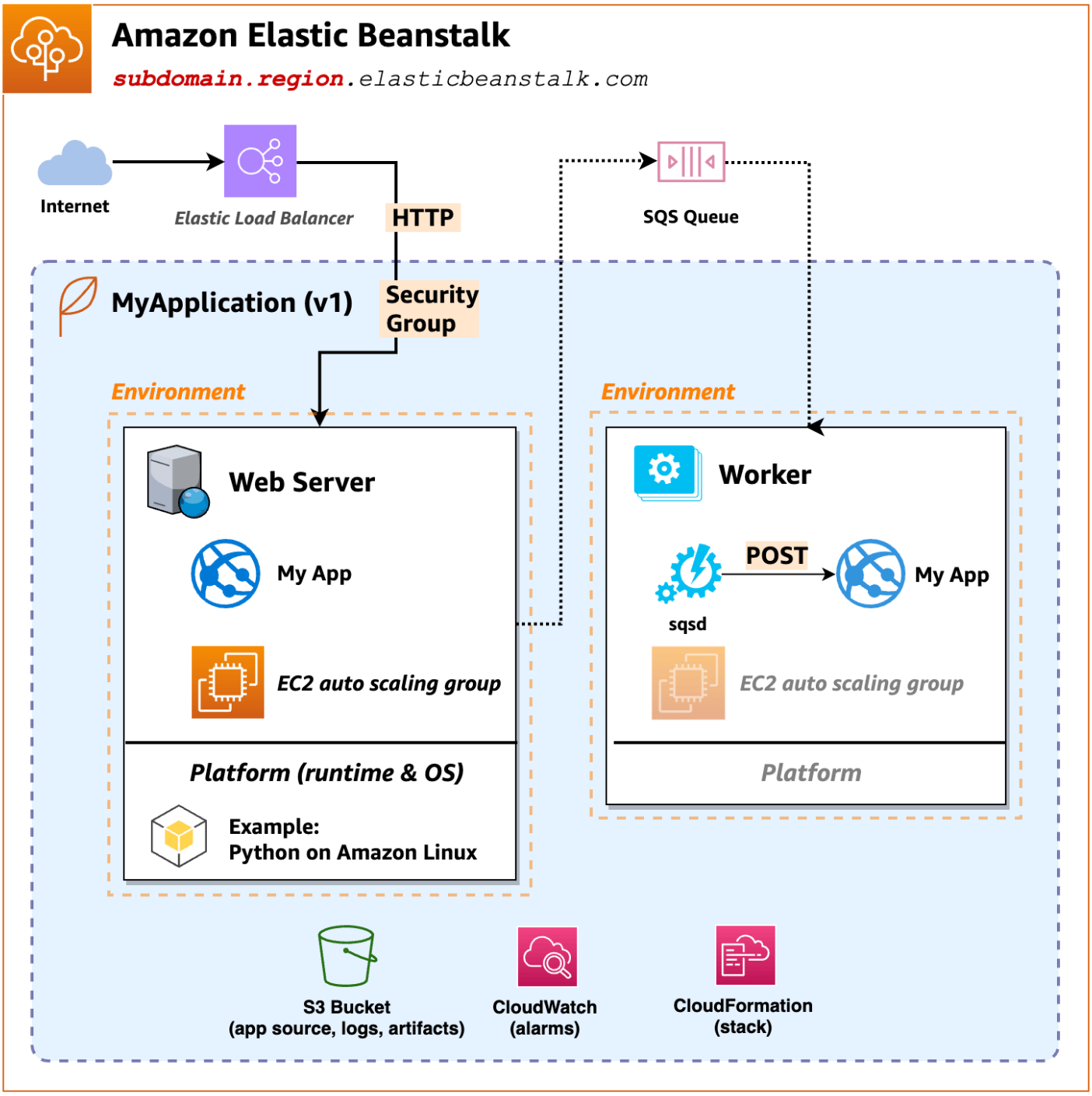
eb use ***environment-name***

Options

Nom	Description
`--source codecommit/` *repository-name*`/`*branch-name*	CodeCommit référentiel et branche.
`-r` *region* `--region` *region*	Modifiez la région dans laquelle vous créez les environnements.
Options courantes	

Comprendre les concepts d'Elastic Beanstalk

En vous familiarisant avec les concepts et les termes, vous pourrez mieux comprendre les éléments nécessaires au déploiement de vos applications avec Elastic Beanstalk.



Application

Une application Elastic Beanstalk est un conteneur pour les composants Elastic Beanstalk, notamment les environnements, les versions et les configurations d'environnement. Dans une application Elastic Beanstalk, vous gérez toutes les ressources nécessaires à l'exécution de votre code.

Version de l'application

Dans Elastic Beanstalk, une version de l'application fait référence à une itération étiquetée et spécifique du code à déployer pour une application web. Une version de l'application pointe vers un objet Amazon Simple Storage Service (Amazon S3) qui contient le code déployable tel qu'un fichier WAR Java.

Une version de l'application fait partie d'une application. Les applications peuvent avoir de nombreuses versions et chaque version de l'application est unique. Dans un environnement d'exploitation, vous pouvez déployer toute version de l'application que vous avez déjà chargée dans l'application ou vous pouvez charger et déployer immédiatement une nouvelle version de l'application. Par exemple, vous pouvez télécharger plusieurs versions d'applications pour tester les différences entre elles.

Environnement

Un environnement est un ensemble de AWS ressources exécutant une version d'application. Chaque environnement exécute une seule version d'application à la fois, cependant, vous pouvez exécuter la même version d'application ou différentes versions d'application dans de nombreux environnements simultanément. Lorsque vous créez un environnement, Elastic Beanstalk fournit les ressources nécessaires à votre compte pour exécuter la version de AWS l'application que vous avez spécifiée.

Niveau environnemental

Lorsque vous lancez un environnement Elastic Beanstalk, vous commencez par choisir un niveau d'environnement. Le niveau d'environnement désigne le type d'application que l'environnement exécute et détermine les ressources qu'Elastic Beanstalk met à disposition pour le soutenir. Une application qui traite des demandes HTTP s'exécute dans une couche d’environnement de serveur Web. Un environnement backend qui extrait des tâches à partir d'une file d'attente Amazon Simple Queue Service (Amazon SQS) s'exécute dans une couche d'environnement de travail.

Configuration de l'environnement

Une configuration de l'environnement identifie un ensemble de paramètres et des réglages qui définissent le comportement d'un environnement et de ses ressources associées. Lorsque vous mettez à jour les paramètres de configuration d'un environnement, Elastic Beanstalk applique automatiquement les modifications aux ressources existantes ou supprime et déploie de nouvelles ressources (en fonction du type de modification).

Configuration enregistrée

Une configuration enregistrée est un modèle que vous pouvez utiliser comme point de départ pour créer des configurations d'environnement uniques. Vous pouvez créer et modifier des configurations enregistrées et les appliquer à des environnements à l'aide de la console Elastic Beanstalk, de l'EB AWS CLI CLI ou de l'API. L'API et le AWS CLI font référence aux configurations enregistrées en tant que modèles de configuration.

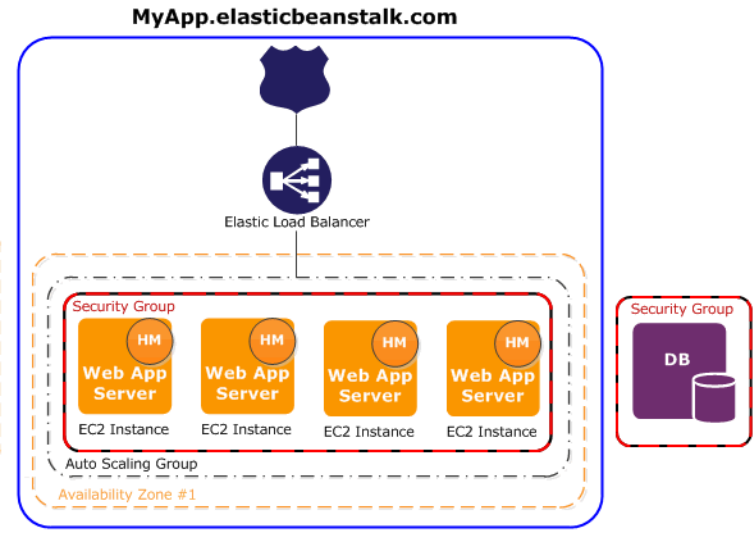
Plateforme

Une plateforme combine un système d'exploitation, un environnement d'exécution de langage de programmation, un serveur web, un serveur d'applications et des composants Elastic Beanstalk. Vous concevez et ciblez votre application web sur une plateforme. Elastic Beanstalk fournit une grande variété de plateformes sur lesquelles vous pouvez créer vos applications.

Pour en savoir plus, consultez Plateformes Elastic Beanstalk.

Environnements de serveurs Web Elastic Beanstalk

Le schéma suivant illustre un exemple d'architecture Elastic Beanstalk pour un niveau d'environnement de serveur web et montre comment les composants de ce type de niveau d'environnement travaillent ensemble.



L'environnement est le cœur de l'application. Dans le schéma, l'environnement est affiché dans la ligne unie de niveau supérieur. Lorsque vous créez un environnement, Elastic Beanstalk fournit les ressources nécessaires pour exécuter votre application. AWS les ressources créées pour un environnement incluent un équilibreur de charge élastique (ELB dans le schéma), un groupe Auto Scaling et une ou plusieurs instances Amazon Elastic Compute Cloud EC2 (Amazon).

Chaque environnement dispose d'un CNAME (URL) qui pointe vers un équilibreur de charge. L'environnement dispose d'une URL telle que `myapp.us-west-2.elasticbeanstalk.com`. Cette URL a un alias dans Amazon Route 53 vers une URL Elastic Load Balancing (similaire à `abcdef-123456.us-west-2.elb.amazonaws.com`) en utilisant un enregistrement CNAME. Amazon Route 53 est un service web de système de noms de domaine (DNS) hautement disponible et évolutif. Cela fournit un acheminement sûr et fiable vers votre infrastructure. Votre nom de domaine que vous avez enregistré avec votre fournisseur DNS réachemine les requêtes vers le CNAME.

L'équilibreur de charge se trouve devant les EC2 instances Amazon, qui font partie d'un groupe Auto Scaling. Amazon EC2 Auto Scaling démarre automatiquement des EC2 instances Amazon supplémentaires pour s'adapter à la charge croissante de votre application. Si la charge de votre application diminue, Amazon EC2 Auto Scaling arrête les instances, mais laisse toujours au moins une instance en cours d'exécution.

La pile logicielle exécutée sur les EC2 instances Amazon dépend du type de conteneur. Un type de conteneur définit la topologie de l'infrastructure et la pile logicielle à utiliser pour cet environnement. Par exemple, un environnement Elastic Beanstalk avec un conteneur Apache Tomcat utilise le système d'exploitation Amazon Linux, le serveur web Apache et le logiciel Apache Tomcat. Pour

afficher la liste des types de conteneurs pris en charge, consultez Plateformes prises en charge par Elastic Beanstalk. Chaque EC2 instance Amazon qui exécute votre application utilise l'un de ces types de conteneurs. En outre, un composant logiciel appelé gestionnaire d'hôtes (HM) s'exécute sur chaque EC2 instance Amazon. Le gestionnaire hôte est chargé des opérations suivantes :

- Déploiement de l'application
- Regroupement des événements et des métriques pour une récupération via la console, l'API ou la ligne de commande
- Génération d'événements niveau instance
- Surveillance des fichiers journaux de l'application pour les erreurs critiques
- Surveillance du serveur d'applications
- Mise à jour corrective de composants de l'instance
- Rotation des fichiers journaux de votre application et publication de ces derniers sur Amazon S3

Le responsable de l'hôte rapporte les métriques, les erreurs et les événements, ainsi que l'état de l'instance du serveur, qui sont disponibles via la console Elastic APIs Beanstalk, et. CLIs

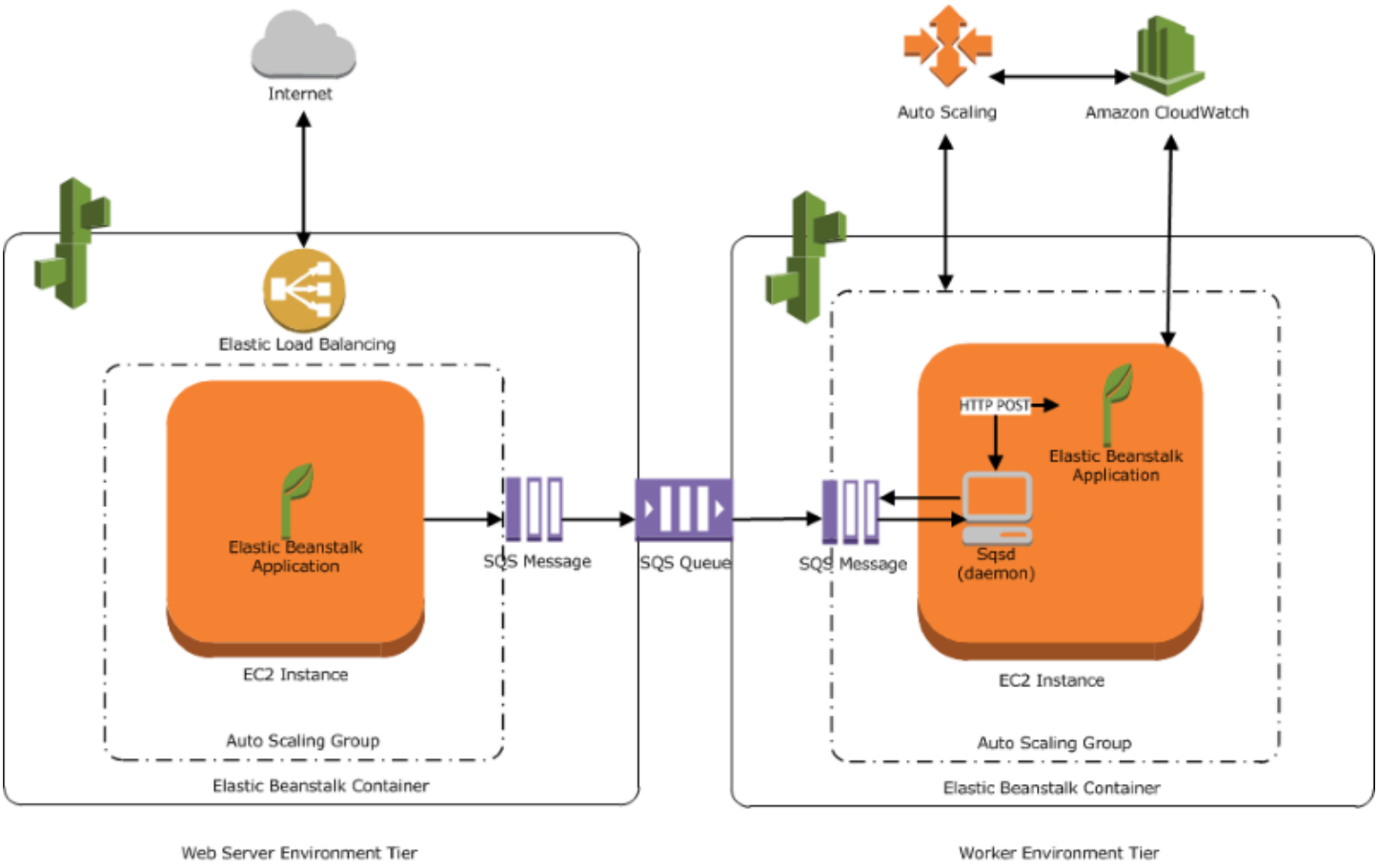
Les EC2 instances Amazon présentées dans le schéma font partie d'un groupe de sécurité. Un groupe de sécurité définit les règles de pare-feu pour vos instances. Par défaut, Elastic Beanstalk définit un groupe de sécurité, ce qui permet à tout le monde de se connecter via le port 80 (HTTP). Vous pouvez définir plus d'un groupe de sécurité. Par exemple, vous pouvez définir un groupe de sécurité pour votre serveur de base de données. Pour plus d'informations sur les groupes EC2 de sécurité Amazon et sur la façon de les configurer pour votre application Elastic Beanstalk, consultez. EC2 groupes de sécurité

Environnements de travail Elastic Beanstalk

AWS les ressources créées pour un niveau d'environnement de travail incluent un groupe Auto Scaling, une ou plusieurs EC2 instances Amazon et un rôle IAM. Pour le niveau d'environnement de travail, Elastic Beanstalk crée et met en service une file d'attente Amazon SQS si vous n'en avez pas déjà. Lorsque vous lancez un environnement de travail, Elastic Beanstalk installe les fichiers de support nécessaires au langage de programmation de votre choix et un daemon sur chaque EC2 instance du groupe Auto Scaling. Le démon lit les messages d'une file d'attente Amazon SQS. Le démon envoie les données à partir de chaque message lu à l'application web exécutée dans l'environnement de travail en vue de leur traitement. Si votre environnement de travail contient

plusieurs instances, chacune d'elles a son propre démon, mais toutes sont lues depuis la même file d'attente Amazon SQS.

Le schéma suivant montre les différents composants et leurs interactions entre les environnements et les AWS services.



Amazon CloudWatch est utilisé pour les alarmes et la surveillance de l'état de santé. Pour plus d'informations, consultez la section concernant Création de rapports d'intégrité de base.

Pour de plus amples informations sur le fonctionnement du niveau d'environnement de travail, veuillez consulter Environnements de travail Elastic Beanstalk.

Considérations relatives à la conception de vos applications Elastic Beanstalk

Étant donné que les applications déployées à l'aide de AWS Cloud ressources AWS Elastic Beanstalk fonctionnent sur les ressources, vous devez tenir compte de plusieurs facteurs de configuration pour optimiser vos applications : évolutivité, sécurité, stockage persistant, tolérance

aux pannes, diffusion de contenu, mises à jour logicielles et correctifs, connectivité. Chacun de ces éléments est abordé séparément dans cette rubrique. Pour une liste complète des AWS livres blancs techniques, couvrant des sujets tels que l'architecture, la sécurité et l'économie, consultez les livres blancs sur AWS le cloud computing.

Capacité de mise à l’échelle

Lorsque vous travaillez dans un environnement matériel physique, contrairement à un environnement cloud, vous pouvez aborder la capacité de mise à l’échelle de l'une des deux façons suivantes. Vous pouvez soit augmenter l'échelle par une mise à l'échelle verticale, soit la réduire par une mise à l'échelle horizontale. L'approche de mise à l'échelle nécessite que vous investissiez dans du matériel puissant, capable de répondre aux exigences croissantes de votre entreprise. L'approche évolutive nécessite que vous suiviez un modèle d'investissement distribué. Ainsi, vos acquisitions de matériel et d'applications peuvent être plus ciblées, vos ensembles de données sont fédérés et votre conception est orientée service. L'approche de mise à l'échelle peut être très coûteuse, et il y a toujours le risque que la demande vienne à dépasser la capacité. À cet égard, l'approche évolutive est généralement plus efficace. Toutefois, lorsque vous l'utilisez, vous devez être en mesure de prédire la demande à intervalles réguliers et de déployer l'infrastructure en morceaux pour répondre à cette demande. Par conséquent, cette approche peut souvent entraîner une capacité inutilisée et nécessiter une surveillance attentive.

En migrant vers le cloud, vous pouvez faire en sorte que votre infrastructure s'aligne correctement sur la demande en tirant parti de l'élasticité du cloud. L'élasticité aide à rationaliser l'acquisition et la libération des ressources. Grâce à elle, votre infrastructure peut rapidement se mettre à l'échelle horizontale et monter en puissance en fonction des fluctuations de la demande. Pour l'utiliser, configurez vos paramètres de scalabilité automatique pour augmenter ou réduire l'échelle en fonction des métriques des ressources de votre environnement. Par exemple, vous pouvez définir des métriques (telles que l'utilisation du serveur ou du réseauI/O. You can use Auto Scaling for compute capacity to be added automatically whenever usage rises and for it to be removed whenever usage drops. You can publish system metrics (for example, CPU, memory, disk I/O, and network I/O) pour Amazon CloudWatch. Vous pouvez ensuite configurer des alarmes CloudWatch afin de déclencher des actions Auto Scaling ou d'envoyer des notifications en fonction de ces métriques. Pour obtenir des instructions sur la configuration de scalabilité automatique, consultez Auto Scaling de vos instances d'environnement Elastic Beanstalk.

Nous vous recommandons également de concevoir toutes vos applications Elastic Beanstalk aussi sans état que possible, utilisant des composants à tolérance de panne et à couplage faible

pouvant monter en puissance en fonction des besoins. Pour plus d'informations sur la conception d'architectures d'applications évolutives pour AWS, consultez AWS Well-Architected Framework.

Sécurité

La sécurité AWS est une responsabilité partagée. Amazon Web Services protège les ressources physiques dans votre environnement et assure que le cloud est un endroit sûr où vous pouvez exécuter des applications. Vous êtes responsable de la sécurité des données entrantes et sortantes de votre environnement Elastic Beanstalk et de la sécurité de votre application.

Configurez SSL pour protéger les informations qui circulent entre votre application et les clients. Pour configurer SSL, vous avez besoin d'un certificat gratuit fourni par AWS Certificate Manager (ACM). Si vous possédez déjà un certificat d'une autorité de certification externe (CA), vous pouvez utiliser ACM pour importer votre certificat. Sinon, vous pouvez l'importer à l'aide du AWS CLI.

Si ACM n'est pas disponible dans votre Région AWS, vous pouvez acheter un certificat auprès d'une autorité de certification externe, telle que VeriSign Entrust. Utilisez ensuite le AWS Command Line Interface (AWS CLI) pour télécharger un certificat tiers ou auto-signé et une clé privée vers Gestion des identités et des accès AWS (IAM). La clé publique du certificat authentifie votre serveur dans le navigateur. Elle sert également de base pour créer la clé de séance partagée qui chiffre les données dans les deux directions. Pour obtenir les instructions sur la création, le chargement et l'attribution d'un certificat à votre environnement, consultez Configuration de HTTPS pour votre environnement Elastic Beanstalk.

Lorsque vous configurez un certificat SSL pour votre environnement, les données sont chiffrées entre le client et l'équilibreur de charge Elastic Load Balancing de votre environnement. Par défaut, le chiffrement est arrêté au niveau de l'équilibreur de charge et le trafic entre l'équilibreur de charge et les EC2 instances Amazon n'est pas chiffré.

Stockage permanent

Les applications Elastic Beanstalk s'exécutent EC2 sur des instances Amazon qui ne disposent pas d'un stockage local persistant. Lorsque les EC2 instances Amazon se terminent, le système de fichiers local n'est pas enregistré. Les nouvelles EC2 instances Amazon commencent par un système de fichiers par défaut. Nous vous recommandons de configurer votre application pour stocker des données dans une source de données permanente. AWS offre un certain nombre de services de stockage permanent que vous pouvez utiliser pour votre application. Le tableau suivant les répertorie.

Service de stockage	Documentation du service	Intégration d'Elastic Beanstalk
Amazon S3	Documentation Amazon Simple Storage Service	Utilisation d'Elastic Beanstalk avec Amazon S3
Amazon Elastic File System	Documentation Amazon Elastic File System	Utilisation d'Elastic Beanstalk avec Amazon Elastic File System
Amazon Elastic Block Store	Amazon Elastic Block Store Feature Guide: Elastic Block Store	
Amazon DynamoDB	Documentation Amazon DynamoDB	Utilisation d'Elastic Beanstalk avec Amazon DynamoDB
Amazon Relational Database Service (RDS)	Documentation Amazon Relational Database Service	Utilisation d'Elastic Beanstalk avec Amazon RDS

Note

Elastic Beanstalk crée un utilisateur d'application Web que vous pouvez configurer en tant que propriétaire des répertoires d'applications sur les instances. EC2 Pour les versions de plateforme Amazon Linux 2 publiées le 3 février 2022 ou après, Elastic Beanstalk attribue à l'utilisateur de l'application web un id utilisateur (identifiant d'utilisateur) et un id groupe (identifiant de groupe) de 900 pour les nouveaux environnements. Il en va de même pour les environnements existants après une mise à jour de la version de la plateforme. Cette approche maintient une autorisation d'accès cohérente pour l'utilisateur de l'application web au stockage permanent du système de fichiers.
Dans le cas peu probable où un autre utilisateur ou processus utilise déjà le 900, le système d'exploitation attribue par défaut une autre valeur à l'idu et à l’idg de l'utilisateur de l'application web. Exécutez la commande Linux id webapp sur vos EC2 instances pour vérifier les valeurs uid et gid attribuées à l'utilisateur de l'application Web.

Tolérance aux pannes

En règle générale, vous devez être pessimiste lors de la conception d'une architecture pour le cloud. Tirez parti de l'élasticité qu'elle offre. Vous devez toujours concevoir, mettre en œuvre et déployer en prévoyant une reprise automatique après une défaillance. Utilisez plusieurs zones de disponibilité pour vos EC2 instances Amazon et pour Amazon RDS. Les zones de disponibilité sont sur le plan conceptuel comme des centres de données logiques. Utilisez Amazon CloudWatch pour avoir une meilleure visibilité sur l'état de santé de votre application Elastic Beanstalk et prendre les mesures appropriées en cas de panne matérielle ou de dégradation des performances. Configurez vos paramètres Auto Scaling pour maintenir votre flotte d' EC2 instances Amazon à une taille fixe afin que les EC2 instances Amazon défaillantes soient remplacées par de nouvelles. Si vous utilisez Amazon RDS, définissez alors la période de conservation des sauvegardes, afin qu'Amazon RDS puisse effectuer des sauvegardes automatiques.

Diffusion de contenu

Lorsque des utilisateurs se connectent à votre site web, leurs demandes peuvent être acheminées via un certain nombre de réseaux individuels. Ainsi, les utilisateurs peuvent subir des performances médiocres en raison d'une latence élevée. Amazon CloudFront peut vous aider à résoudre les problèmes de latence en distribuant votre contenu Web, tel que des images et des vidéos, sur un réseau de sites périphériques dans le monde entier. Les demandes des utilisateurs sont acheminées vers l'emplacement périphérique le plus proche, afin que le contenu soit diffusé avec les meilleures performances possibles. CloudFront fonctionne parfaitement avec Amazon S3, qui stocke de manière durable les versions originales et définitives de vos fichiers. Pour plus d'informations sur Amazon CloudFront, consultez le manuel du CloudFront développeur Amazon.

Correctifs et mises à jour de logiciels

AWS Elastic Beanstalk publie régulièrement des mises à jour de plate-forme pour fournir des correctifs, des mises à jour logicielles et de nouvelles fonctionnalités. Elastic Beanstalk offre plusieurs options pour gérer les mises à jour de la plateforme. Grâce aux mises à jour gérées de la plateforme, votre environnement passe automatiquement à la dernière version d'une plateforme pendant une fenêtre de maintenance programmée, alors que votre application reste en service. Pour les environnements créés le 25 novembre 2019 ou ultérieurement à l'aide de la console Elastic Beanstalk, les mises à jour gérées sont activées par défaut dans la mesure du possible. Vous pouvez également lancer des mises à jour manuellement à l'aide de la console Elastic Beanstalk ou de l'interface de ligne de commande EB.

Connectivité

Elastic Beanstalk doit être capable de se connecter aux instances de votre environnement pour compléter les déploiements. Lorsque vous déployez une application Elastic Beanstalk à l'intérieur d'un Amazon VPC, la configuration requise pour activer la connectivité dépend du type d'environnement Amazon VPC que vous créez :

- Pour les environnements à une seule instance, aucune configuration supplémentaire n'est requise. En effet, dans ces environnements, Elastic Beanstalk attribue à EC2 chaque instance Amazon une adresse IP Elastic publique qui permet à l'instance de communiquer directement avec Internet.
- Pour les environnements évolutifs et à équilibrage de charge dans un Amazon VPC avec des sous-réseaux publics et privés, vous devez procéder comme suit :
  - Créez un équilibreur de charge dans le sous-réseau public pour acheminer le trafic entrant depuis Internet vers les instances Amazon. EC2
  - Créez un appareil de traduction d'adresses réseau (NAT) pour acheminer le trafic sortant des EC2 instances Amazon dans des sous-réseaux privés vers Internet.
  - Créez des règles de routage entrantes et sortantes pour les EC2 instances Amazon au sein du sous-réseau privé.
  - Si vous utilisez une instance NAT, configurez les groupes de sécurité pour l'instance NAT et les EC2 instances Amazon afin de permettre la communication Internet.
- Pour un environnement évolutif et équilibré en charge dans un Amazon VPC qui dispose d'un sous-réseau public, aucune configuration supplémentaire n'est requise. En effet, dans cet environnement, vos EC2 instances Amazon sont configurées avec une adresse IP publique qui leur permet de communiquer avec Internet.

Pour plus d'informations sur l'utilisation d'Elastic Beanstalk avec Amazon VPC, consultez Utilisation d'Elastic Beanstalk avec Amazon VPC.

Gestion des applications Elastic Beanstalk

Ce chapitre explique comment gérer et configurer vos applications Elastic Beanstalk. La première étape de l'utilisation AWS Elastic Beanstalk consiste à créer une application qui représente votre application Web dans AWS. Dans Elastic Beanstalk, une application sert de conteneur pour les environnements qui exécutent votre application web, et les versions du code source de votre application web, les configurations enregistrées, les journaux et autres artefacts que vous créez lorsque vous utilisez Elastic Beanstalk.

Pour créer une application

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le volet de navigation, choisissez Applications (Applications), puis Create a new application (Créer une nouvelle application).
3. Utilisez le formulaire à l'écran pour fournir un nom d'application.
4. (Facultatif) Fournissez une description et ajoutez des clés et des valeurs de balise.
5. Choisissez Créer.

Une fois que vous avez créé une application, la console vous invite à créer un environnement pour cette application. Pour obtenir des informations détaillées sur toutes les options disponibles, veuillez consulter Création d'un environnement Elastic Beanstalk.

Si vous n'avez plus besoin d'une application, vous pouvez la supprimer.

> **Warning**
>
> La suppression d'une application entraîne la mise hors service de tous les environnements associés, ainsi que la suppression de toutes les versions d'application et des configurations enregistrées appartenant à l'application.

Pour supprimer une application

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS

2. Dans le volet de navigation, choisissez Applications, puis sélectionnez votre application dans la liste.
3. Choisissez Actions, puis Delete application (Supprimer l'application).

Rubriques

- Console de gestion d'application Elastic Beanstalk
- Gestion des versions d'application
- Création d'un bundle de sources d'applications Elastic Beanstalk
- Utilisation de l'interface de ligne de commande EB avec AWS CodeBuild
- L'étiquette des applications
- Étiquette des ressources d'application Elastic Beanstalk

Console de gestion d'application Elastic Beanstalk

Cette rubrique explique comment utiliser la AWS Elastic Beanstalk console pour gérer les applications, les versions des applications et les configurations enregistrées.

Pour accéder à la console de gestion des applications

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.

   La page de présentation de l'application affiche une liste avec une présentation de tous les environnements associés à l'application.
3. Vous disposez de plusieurs façons de continuer :

   a. Dans le menu déroulant Actions, vous pouvez choisir l'une des actions de gestion des applications : créer un environnement, supprimer l'application, afficher les versions de l'application, afficher les configurations enregistrées, restaurer l'environnement arrêté.

      Pour lancer un environnement dans cette application, vous pouvez directement choisir Créer un environnement. Pour en savoir plus, consultez the section called "Création d'environnements".

b. La page répertorie le nom de l'environnement à côté des applications déployées dans un environnement. Choisissez un nom d'environnement pour accéder à la console de gestion des environnements pour cet environnement, où vous pouvez configurer, surveiller ou gérer ce dernier.

c. Lorsque vous sélectionnez une application dans la liste, le volet de navigation de gauche répertorie l'application.

   - Choisissez Application versions (Versions d'application) en suivant le nom de l'application dans le volet de navigation pour afficher et gérer les versions d'application de votre application.

     Une version d'application est une version téléchargée de votre code d'application. Vous pouvez télécharger de nouvelles versions, déployer une version existante sur n'importe quel environnement de l'application, ou supprimer d'anciennes versions. Pour de plus amples informations, veuillez consulter Gestion des versions d'application.

   - Choisissez Saved configurations (Configurations enregistrées) en suivant le nom de l'application dans le volet de navigation pour afficher et gérer les configurations enregistrées à partir d'environnements en cours d'exécution.

     Une configuration enregistrée est un ensemble de paramètres que vous pouvez utiliser pour restaurer les paramètres d'un environnement à un état antérieur ou créer un environnement avec les mêmes paramètres. Pour de plus amples informations, veuillez consulter Utilisation des configurations enregistrées par Elastic Beanstalk.

Gestion des versions d'application

Cette rubrique explique les versions des applications et explique comment les créer et les gérer.

Elastic Beanstalk crée une version d'application chaque fois que vous chargez le code source. Généralement, cette situation se produit lorsque vous créez un environnement, ou lorsque vous chargez et déployez le code via la console de gestion de l'environnement ou l'interface de ligne de commande EB. Elastic Beanstalk supprime ces versions de l'application en fonction de la stratégie de cycle de vie de l'application et lorsque vous supprimez l'application. Pour plus d'informations sur la stratégie de cycle de vie de l'application, consultez Configuration des paramètres du cycle de vie des versions d'application.

Vous pouvez également télécharger un bundle de fichiers source sans le déployer depuis la console de gestion des applications ou avec la commande de l'interface de ligne de commande EB eb

appversion. Elastic Beanstalk stocke les lots source dans Amazon Simple Storage Service (Amazon S3) et ne les supprime pas automatiquement.

Vous pouvez appliquer des balises à une version d'application lorsque vous la créez, et modifier les balises de versions d'applications existantes. Pour en savoir plus, consultez Étiquette des versions d'application.

Création de versions d'applications

Vous pouvez également créer une nouvelle version d'application à l'aide de l'interface de ligne de commande EB. Pour plus d'informations, veuillez consultereb appversion au chapitre Commandes de l'interface de ligne de commande EB.

> **Note**
>
> Au fil du temps, votre application peut accumuler un grand nombre de versions de l'application. Pour économiser de l'espace de stockage et éviter d'atteindre le quota de versions de l'application, il est conseillé de supprimer les versions dont vous n'avez plus besoin.

Le fichier que vous spécifiez dans la procédure suivante est associé à votre candidature. Vous pouvez déployer la version d'application dans un environnement nouveau ou existant.

Pour créer une version d’application

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.
3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Application versions (Versions d'application).
4. Choisissez Charger. Utilisez le formulaire à l'écran pour télécharger le bundle source de votre application.

   > **Note**
   >
   > La taille de fichier du bundle de fichiers source ne doit pas dépasser 62,5 Mo.

5. Le cas échéant, fournissez une brève description, et ajouter les clés et valeurs de balise.
6. Choisissez Charger.

Supprimer des versions de l'application

Vous pouvez également supprimer une version d'application à l'aide de l'interface de ligne de commande EB. Pour plus d'informations, veuillez consultereb appversion au chapitre Commandes de l'interface de ligne de commande EB.

> **Note**
>
> La suppression d'une version d'application n'a aucune incidence sur les environnements qui exécutent actuellement cette version.

Vous pouvez également configurer Elastic Beanstalk pour supprimer automatiquement les anciennes versions en configurant les paramètres du cycle de vie de la version d'application. Si vous configurez ces paramètres de cycle de vie, ils sont appliqués lorsque vous créez de nouvelles versions de l'application. Par exemple, si vous configurez un maximum de 25 versions d'application, Elastic Beanstalk supprime la version la plus ancienne lorsque vous chargez une 26e version. Si vous définissez une ancienneté maximale de 90 jours, toute version dont l'ancienneté dépasse 90 jours est supprimée lorsque vous chargez une nouvelle version. Pour en savoir plus, consultez the section called “Cycle de vie des versions”.

Pour supprimer une version d’application

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.
3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Application versions (Versions d'application).
4. Sélectionnez une ou plusieurs versions de l’application à supprimer.
5. Choisissez Actions, puis Delete (Supprimer).

6. (Facultatif) Pour laisser le bundle des fichiers source d'application pour ces versions d'application dans votre compartiment Amazon Simple Storage Service (Amazon S3), désactivez la case Delete versions from Amazon S3 (Supprimer les versions d'Amazon S3).
7. Sélectionnez Delete (Supprimer).

Si vous ne choisissez pas de supprimer le lot source d'Amazon S3, Elastic Beanstalk supprime toujours la version de ses enregistrements. Toutefois, le bundle source est laissé dans votre compartiment de stockage Elastic Beanstalk. Le quota de versions d'application s'applique uniquement aux versions des pistes Elastic Beanstalk. Vous pouvez donc supprimer des versions pour respecter le quota, mais conserver tous les bundles de fichiers source dans Amazon S3.

> **Note**
>
> Le quota de versions d'application ne s'applique pas aux bundles de fichiers source, mais vous pouvez cependant encourir des frais Amazon S3 et conserver des informations personnelles même si vous n'en avez plus besoin. Elastic Beanstalk ne supprime jamais les bundles de fichiers source automatiquement. Vous devez supprimer les bundles de fichiers source lorsque vous n'en avez plus besoin.

Configuration des paramètres du cycle de vie des versions d'application

Cette rubrique explique les politiques et les quotas qu'Elastic Beanstalk applique aux versions de votre application dans un environnement donné, y compris la durée pendant laquelle une version de l'application reste dans un environnement.

Chaque fois que vous téléchargez une nouvelle version de votre application avec la console Elastic Beanstalk ou l'interface de ligne de commande EB, Elastic Beanstalk crée une version d'application. Si vous ne supprimez pas les versions que vous n'utilisez plus, vous finirez par atteindre le quota des versions d'application et vous ne pourrez pas créer de nouvelles versions de cette application.

Vous pouvez éviter d'atteindre le quota en appliquant une stratégie de cycle de vie des versions d'application à vos applications. Une stratégie de cycle de vie demande à Elastic Beanstalk de supprimer les versions d'application qui sont obsolètes, ou de supprimer des versions d'application lorsque le nombre total de versions pour une application dépasse un nombre spécifié.

Elastic Beanstalk applique la stratégie de cycle de vie d'une application chaque fois que vous créez une nouvelle version d'application, et supprime jusqu'à 100 versions chaque fois que la stratégie de

cycle de vie est appliquée. Elastic Beanstalk supprime les anciennes versions après avoir créé la nouvelle version et ne comptabilise pas la nouvelle version dans le nombre de versions maximum défini dans la stratégie.

Elastic Beanstalk ne supprime pas les versions d'application en cours d'utilisation par un environnement, ou celles déployées dans des environnements qui ont été arrêtés moins de dix semaines avant le déclenchement de la stratégie.

Le quota de versions d'application s'applique à toutes les applications dans une région. Si vous avez plusieurs applications, configurez chacune d'entre elles avec une stratégie de cycle de vie appropriée pour éviter d'atteindre le quota. Par exemple, si vous avez 10 applications dans une région et que le quota est de 1 000 versions d'application, envisagez de définir une stratégie de cycle de vie avec un quota de 99 versions d'application pour toutes les applications, ou définissez d'autres valeurs dans chaque application tant que le total est inférieur à 1 000 versions d'application. Elastic Beanstalk applique uniquement la stratégie si la création de la version d'application réussit. Si vous avez déjà atteint le quota, vous devez donc supprimer certaines versions manuellement avant de créer une nouvelle version.

Par défaut, Elastic Beanstalk quitte le bundle de fichiers source de la version d'application dans Amazon S3 pour éviter la perte de données. Vous pouvez supprimer le bundle de fichiers source pour économiser de l'espace.

Vous pouvez définir les paramètres du cycle de vie via la CLI Elastic Beanstalk et. APIs Voireb appversion, CreateApplication(en utilisant le `ResourceLifecycleConfig` paramètre), et UpdateApplicationResourceLifecyclepour plus de détails.

Configuration des paramètres du cycle de vie d'une application dans la console

Vous pouvez spécifier les paramètres de cycle de vie dans la console Elastic Beanstalk.

Pour spécifier les paramètres de cycle de vie de votre application

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.
3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Application versions (Versions d'application).

4. Cliquez sur Paramètres.
5. Utilisez le formulaire à l'écran pour configurer les paramètres de cycle de vie de l'application.
6. Choisissez Enregistrer.

La page des paramètres vous permet d'effectuer les actions suivantes.

- Configurer les paramètres de cycle de vie en fonction du nombre total de versions d'application ou de l'âge des versions d'application.
- Spécifier si le bundle de fichiers source doit être supprimé dans S3 lorsque la version de l'application est supprimée.
- Spécifier le rôle sous lequel la version de l'application est supprimée. Pour inclure toutes les autorisations requises pour la suppression de la version, choisissez le rôle de service Elastic Beanstalk par défaut, nommé `aws-elasticbeanstalk-service-role`, ou un autre rôle de service à l'aide des stratégies de service gérées Elastic Beanstalk. Pour de plus amples informations, veuillez consulter Gestion des rôles de service Elastic Beanstalk.

Étiquette des versions d'application

Cette rubrique explique les avantages du balisage des versions de votre application Elastic Beanstalk et explique comment gérer les balises.

Vous pouvez appliquer des balises aux versions de vos AWS Elastic Beanstalk applications. Les balises sont des paires clé-valeur associées à des AWS ressources. Pour plus d'informations sur l'étiquette des ressources Elastic Beanstalk, les cas d'utilisation, les contraintes de clé et de valeur d'identification, et les types de ressources pris en charge, consultez Étiquette des ressources d'application Elastic Beanstalk.

Vous pouvez spécifier des identifications lorsque vous créez une version d'application. Dans une version d'application, vous pouvez ajouter ou supprimer des identifications, ainsi que mettre à jour les valeurs des identifications existantes. Vous pouvez ajouter jusqu'à 50 identifications par version de l'application.

Ajout d'identifications lors de la création de la version de l'application

Lorsque vous utilisez la console Elastic Beanstalk pour créer un environnement, et que vous choisissez de charger une version de votre code d'application, vous pouvez spécifier des clés et valeurs d'identification à associer à la nouvelle version d'application.

Vous pouvez également utiliser la console Elastic Beanstalk pour charger une version d'application sans immédiatement l'utiliser dans un environnement. Vous pouvez spécifier des clés et des valeurs d'identification lorsque vous chargez une version d'application.

Avec le AWS CLI ou d'autres clients basés sur une API, ajoutez des balises en utilisant le `--tags` paramètre de la commande. create-application-version

```
$ aws elasticbeanstalk create-application-version \
      --tags Key=mytag1,Value=value1 Key=mytag2,Value=value2 \
      --application-name my-app --version-label v1
```

Lorsque vous utilisez l'interface de ligne de commande EB pour créer ou mettre à jour un environnement, une version d'application est créée à partir du code que vous déployez. Il n'y a pas de moyen direct d'étiqueter une version d'application lors de sa création via l'interface de ligne de commande EB. Consultez la section suivante pour en savoir plus sur l'ajout d'identifications à une version d'application existante.

Gestion des identifications de la version d'une application existante

Vous pouvez ajouter, mettre à jour et supprimer des identifications dans une version d'application Elastic Beanstalk existante.

Pour gérer les identifications d'une version d'application à l'aide de la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.
3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Application versions (Versions d'application).
4. Sélectionnez la version d’application que vous souhaitez gérer.
5. Choisissez Actions, puis Manage Tags (Gérer les identifications).
6. Utilisez le formulaire à l'écran pour ajouter, mettre à jour ou supprimer des identifications.
7. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Si vous utilisez l'interface de ligne de commande EB pour mettre à jour la version de votre application, utilisez eb tags pour ajouter, mettre à jour, supprimer ou répertorier des identifications.

Par exemple, la commande suivante répertorie les identifications d'une version d'application.

```
~/workspace/my-app$ eb tags --list --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:applicationversion/my-app/my-version"
```

La commande suivante met à jour l'identification `mytag1` et supprime la l'identification `mytag2`.

```
~/workspace/my-app$ eb tags --update mytag1=newvalue --delete mytag2 \
      --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:applicationversion/my-app/my-version"
```

Pour obtenir une liste complète des options et d'autres exemples, consultez `eb tags`.

Avec le AWS CLI ou d'autres clients basés sur une API, utilisez la list-tags-for-resource commande pour répertorier les balises d'une version d'application.

```
$ aws elasticbeanstalk list-tags-for-resource --resource-arn
 "arn:aws:elasticbeanstalk:us-east-2:my-account-id:applicationversion/my-app/my-version"
```

Utilisez la commande update-tags-for-resource pour ajouter, mettre à jour ou supprimer des identifications dans une version d'application.

```
$ aws elasticbeanstalk update-tags-for-resource \
      --tags-to-add Key=mytag1,Value=newvalue --tags-to-remove mytag2 \
      --resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:applicationversion/my-app/my-version"
```

Spécifiez les balises à ajouter et les identifications à mettre à jour dans le paramètre `--tags-to-add` de update-tags-for-resource. Une identification inexistante est ajoutée et la valeur d'une identification existante est mise à jour.

> **Note**
>
> Pour utiliser certaines AWS CLI commandes et certaines de l'interface de ligne de commande EB avec une version d'application Elastic Beanstalk, vous avez besoin de l'ARN de la version de l'application. Vous pouvez extraire l'ARN à l'aide de la commande suivante.

```
$ aws elasticbeanstalk describe-application-versions --application-name my-app
 --version-label my-version
```

Création d'un bundle de sources d'applications Elastic Beanstalk

Cette rubrique explique comment télécharger les fichiers source de votre application vers Elastic Beanstalk dans un bundle de sources. Il explique les exigences d'un ensemble de sources, sa structure et les approches pour en créer un.

Lorsque vous utilisez la AWS Elastic Beanstalk console pour déployer une nouvelle application ou une version d'application, vous devez télécharger les fichiers de l'application dans un bundle source. Votre groupe source doit répondre aux critères suivants :

- Se compose d'un seul fichier `ZIP` ou fichier `WAR` (vous pouvez inclure plusieurs fichiers `WAR` à l'intérieur de votre fichier `ZIP`)
- Ne dépasse pas 62,5 Mo
- N'inclut pas un dossier parent ou un répertoire de niveau supérieur (sous-répertoires acceptés)

Si vous souhaitez déployer une application de travail qui traite des tâches en arrière-plan de façon périodique, votre groupe source d'application doit également inclure un fichier `cron.yaml`. Pour de plus amples informations, veuillez consulter Tâches périodiques.

Si vous déployez votre application à l'aide de l'interface de ligne de commande Elastic Beanstalk (EB CLI), du Toolkit AWS for Eclipse ou du Toolkit for AWS Visual Studio, le fichier ZIP ou WAR sera automatiquement structuré correctement. Pour plus d'informations, consultez Configuration de l'interface de ligne de commande EB (EB CLI) pour gérer Elastic Beanstalk, Déploiement d'applications Java avec Elastic Beanstalk et Le AWS Toolkit for Visual Studio.

Sections

- Création d'une solution groupée source à partir de la ligne de commande
- Création d'une solution groupée source avec Git
- Compression de fichiers dans le Finder de Mac OS X ou l'Explorateur Windows
- Création d'une solution groupée source pour une application .NET
- Test de votre solution groupée source

Création d'une solution groupée source à partir de la ligne de commande

Créer un bundle de fichiers source à l'aide de la commande `zip`. Pour inclure les fichiers et les dossiers cachés, utilisez un schéma semblable au suivant.

```
~/myapp$ zip ../myapp.zip -r * .[^.]*
  adding: app.js (deflated 63%)
  adding: index.js (deflated 44%)
  adding: manual.js (deflated 64%)
  adding: package.json (deflated 40%)
  adding: restify.js (deflated 85%)
  adding: .ebextensions/ (stored 0%)
  adding: .ebextensions/xray.config (stored 0%)
```

Cela permet de garantir que les fichiers de configuration Elastic Beanstalk et d'autres fichiers et dossiers qui commencent par un point sont inclus dans l'archive.

Pour les applications web Tomcat, utilisez `jar` pour créer une archive web.

```
~/myapp$ jar -cvf myapp.war .
```

Les commandes ci-dessus incluent des fichiers masqués qui peuvent augmenter la taille du bundle de fichiers source inutilement. Pour plus de contrôle, utilisez un modèle de fichier plus détaillé ou créez votre bundle source avec Git.

Création d'une solution groupée source avec Git

Si vous utilisez Git pour gérer le code source de votre application, utilisez la commande `git archive` pour créer votre groupe source.

```
$ git archive -v -o myapp.zip --format=zip HEAD
```

`git archive` comprend uniquement les fichiers qui sont stockés dans Git, et exclut les fichiers ignorés et Git. Cela vous aide à conserver un bundle de fichiers source le plus petit possible. Pour plus d'informations, consultez la page de manuel git-archive.

Compression de fichiers dans le Finder de Mac OS X ou l'Explorateur Windows

Lorsque vous créez un fichier `ZIP` dans le Finder de Mac OS X ou l'Explorateur Windows, assurez-vous de compresser les fichiers et les sous-dossiers eux-mêmes, plutôt que de compresser le dossier parent.

> **Note**
>
> L'interface utilisateur graphique (GUI) sur les systèmes d'exploitation basés sur Mac OS X et sur Linux n'affiche pas les fichiers et les dossiers dont les noms commencent par un point (.). Utilisez la ligne de commande au lieu de l'interface utilisateur graphique pour compresser votre application si le fichier `ZIP` doit inclure un dossier masqué, comme `.ebextensions`. Pour que les procédures de ligne de commande créent un fichier `ZIP` sur un système d'exploitation basé sur Linux ou sur Mac OS X, consultez Création d'une solution groupée source à partir de la ligne de commande.

Example

Supposons que vous avez un dossier de projet Python intitulé `myapp` qui inclut les fichiers et les sous-dossiers suivants :

```
myapplication.py
README.md
static/
static/css
static/css/styles.css
static/img
static/img/favicon.ico
static/img/logo.png
templates/
templates/base.html
templates/index.html
```

Comme indiqué dans la liste des exigences ci-dessus, votre groupe source doit être compressé sans un dossier parent, afin que sa structure décompressée n'inclue pas un répertoire supplémentaire de niveau supérieur. Dans cet exemple, aucun dossier `myapp` ne doit être créé lorsque les fichiers

sont décompressés (ou, dans la ligne de commande, aucun segment `myapp` ne doit être ajouté aux chemins d'accès aux fichiers).

Cet exemple de structure de fichiers est utilisé dans cette rubrique pour illustrer comment compresser des fichiers.

Création d'une solution groupée source pour une application .NET

Si vous utilisez Visual Studio, vous pouvez utiliser l'outil de déploiement inclus dans le AWS Toolkit for Visual Studio pour déployer votre application .NET sur Elastic Beanstalk. Pour de plus amples informations, veuillez consulter Déploiement d'applications Elastic Beanstalk dans .NET à l'aide d'outils de déploiement AWS.

Si vous avez besoin de créer manuellement un groupe source pour votre application .NET, vous ne pouvez pas créer simplement un fichier `ZIP` contenant le répertoire du projet. Vous devez créer un package de déploiement web pour votre projet convenant pour le déploiement sur Elastic Beanstalk. Il existe plusieurs méthodes que vous pouvez utiliser pour créer un package de déploiement :

- Créez le package de déploiement à l'aide de l'assistant Publier le site Web dans Visual Studio. Pour de plus amples informations, veuillez consulter Comment : créer un Package de déploiement Web dans Visual Studio.

  > **Important**
  >
  > Lorsque vous créez le package de déploiement web, le nom du site doit commencer par `Default Web Site`.

- Si vous avez un projet .NET, vous pouvez créer le package de déploiement à l'aide de la commande msbuild, comme illustré dans l'exemple suivant.

  > **Important**
  >
  > Le paramètre `DeployIisAppPath` doit commencer par `Default Web Site`.

  ```
  C:/> msbuild <web_app>.csproj /t:Package /p:DeployIisAppPath="Default Web Site"
  ```

- Si vous avez un projet de site web, vous pouvez utiliser l'outil Web Deploy IIS pour créer le package de déploiement. Pour plus d'informations, consultez Packaging and Restoring a Web site.

> **Important**
>
> Le paramètre `apphostconfig` doit commencer par `Default Web Site`.

Si vous déployez plusieurs applications ou une application ASP.NET Core, placez votre dossier `.ebextensions` à la racine de votre bundle de fichiers source, côte à côte avec les ensembles d'applications et le fichier manifeste :

```
~/workspace/source-bundle/
|-- .ebextensions
|   |-- environmentvariables.config
|   `-- healthcheckurl.config
|-- AspNetCore101HelloWorld.zip
|-- AspNetCoreHelloWorld.zip
|-- aws-windows-deployment-manifest.json
`-- VS2015AspNetWebApiApp.zip
```

Test de votre solution groupée source

Vous pouvez souhaiter tester votre groupe source localement avant de le télécharger dans Elastic Beanstalk. Étant donné qu'Elastic Beanstalk utilise essentiellement la ligne de commande pour extraire les fichiers, il est préférable de réaliser vos tests à partir de la ligne de commande, plutôt qu'avec un outil GUI.

Assurez-vous que les fichiers décompressés s'affichent dans le même dossier que l'archive elle-même, plutôt que dans un nouveau répertoire ou dossier de niveau supérieur.

Utilisation de l'interface de ligne de commande EB avec AWS CodeBuild

AWS CodeBuild compile votre code source, exécute des tests unitaires et produit des artefacts prêts à être déployés. Vous pouvez utiliser CodeBuild avec l'interface de ligne de commande EB pour automatiser la création de votre application à partir de son code source. La création de l'environnement et chaque déploiement ultérieur commencent par une étape de développement, suivie du déploiement de l'application qui en résulte.

> **Note**
>
> Certaines régions n'en proposent pas CodeBuild. L'intégration entre Elastic CodeBuild Beanstalk et Elastic ne fonctionne pas dans ces régions.
> Pour plus d'informations sur les AWS services offerts dans chaque région, consultez le tableau des régions.

Création d’une application

Pour créer une application Elastic Beanstalk qui utilise CodeBuild

1. Incluez un fichier de spécification de CodeBuild construction dans le dossier de votre application. `buildspec.yml`
2. Ajoutez au fichier une entrée `eb_codebuild_settings` avec des options spécifiques à Elastic Beanstalk.
3. Exécutez eb init dans le dossier.

> **Note**
>
> N'utilisez pas le point (`.`) ou l'espace ( ) dans le nom de l'application lorsque vous utilisez l'EB CLI avec CodeBuild.

Elastic Beanstalk CodeBuild étend le format du fichier de spécification de build pour inclure les paramètres supplémentaires suivants :

```
eb_codebuild_settings:
  CodeBuildServiceRole: role-name
  ComputeType: size
  Image: image
  Timeout: minutes
```

`CodeBuildServiceRole`

L'ARN ou le nom du rôle de service Gestion des identités et des accès AWS (IAM) qui CodeBuild peut être utilisé pour interagir avec les AWS services dépendants en votre nom. Cette valeur est obligatoire. Si vous l'omettez, toute commande eb deploy ou eb create suivante échoue.

Pour en savoir plus sur la création d'un rôle de service pour CodeBuild, voir Créer un rôle CodeBuild de service dans le guide de AWS CodeBuild l'utilisateur.

> **Note**
>
> Vous avez également besoin d'autorisations pour effectuer des actions en tant CodeBuild que telles. La politique relative aux utilisateurs gérés par AdministratorAccessElastic AWSElastic Beanstalk - Beanstalk inclut toutes les autorisations d'action requises. CodeBuild Si vous n'utilisez pas la stratégie gérée, veillez à autoriser les autorisations suivantes dans votre stratégie d'utilisateur.
>
> ```
> "codebuild:CreateProject",
> "codebuild:DeleteProject",
> "codebuild:BatchGetBuilds",
> "codebuild:StartBuild"
> ```
>
> Pour en savoir plus, consultez Gestion des stratégies utilisateur Elastic Beanstalk.

`ComputeType`

La quantité de ressources utilisée par le conteneur Docker dans l'environnement de CodeBuild construction. Les valeurs valides sont BUILD_ GENERAL1 _SMALL, BUILD_ _MEDIUM et BUILD_ GENERAL1 _LARGE. GENERAL1

`Image`

Nom de l'image Docker Hub ou Amazon ECR CodeBuild utilisée pour l'environnement de génération. Cette image Docker doit contenir tous les outils et bibliothèques d'exécution nécessaires à la création de votre code et doit correspondre à la plate-forme cible de votre application. CodeBuild gère et gère un ensemble d'images spécifiquement destinées à être utilisées avec Elastic Beanstalk. Il est recommandé d'utiliser l'une d'elles. Pour plus de détails, consultez la section Images Docker fournies par CodeBuild dans le guide de l'AWS CodeBuild utilisateur.

La valeur `Image` est facultative. Si vous l'omettez, la commande eb init tente de choisir une image qui correspondent le mieux à votre plateforme cible. De plus, si vous exécutez eb init en mode interactif et si le choix automatique d'image échoue, vous êtes invité à en choisir une. Au terme de l'initialisation, eb init écrit l'image choisie dans le fichier `buildspec.yml`.

`Timeout`

Durée, en minutes, pendant laquelle le CodeBuild build s'exécute avant l'expiration du délai imparti. Cette valeur est facultative. Pour plus de détails sur les valeurs valides et par défaut, voir Créer un projet de construction dans CodeBuild.

> **Note**
>
> Ce délai contrôle la durée maximale d'une CodeBuild exécution, et l'EB CLI le respecte également dans le cadre de la première étape de création d'une version d'application. Il est différent de la valeur que vous pouvez spécifier avec l'option `--timeout` des commandes eb create ou eb deploy. Cette valeur contrôle la durée maximale pendant laquelle l'interface de ligne de commande EB attend la création ou la mise à jour de l'environnement.

Génération et déploiement du code de votre application

Chaque fois que le code de votre application doit être déployé, l'interface de ligne de commande EB exécute une compilation, puis déploie les artefacts de génération qui en résultent dans votre environnement. CodeBuild Cela se produit lorsque vous créez un environnement Elastic Beanstalk pour votre application à l'aide de la commande eb create et chaque fois que vous déployez par la suite des modifications du code dans l'environnement à l'aide de la commande eb deploy.

Si l' CodeBuild étape échoue, la création ou le déploiement de l'environnement ne démarre pas.

L'étiquette des applications

Cette rubrique explique les avantages du balisage de vos applications Elastic Beanstalk. Il fournit également des instructions pour créer et gérer les balises d'application. Les balises sont des paires clé-valeur associées à des AWS ressources. Pour plus d'informations sur l'étiquette des ressources Elastic Beanstalk, les cas d'utilisation, les contraintes de clé et de valeur d'identification, et les types de ressources pris en charge, consultez Étiquette des ressources d'application Elastic Beanstalk.

Vous pouvez spécifier des identifications lorsque vous créez une application. Dans une application existante, vous pouvez ajouter ou supprimer des identifications, ainsi que mettre à jour les valeurs des identifications existantes. Vous pouvez ajouter jusqu'à 50 identifications à chaque application.

Ajout d'identifications lors de la création d'application

Lorsque vous utilisez la console Elastic Beanstalk pour créer une application, vous pouvez spécifier des clés et valeurs d'identification dans la boîte de dialogue Create New Application (Créer une nouvelle application).

Si vous utilisez l'interface de ligne de commande EB pour créer une application, utilisez l'option `--tags` avec eb init pour ajouter des identifications.

```
~/workspace/my-app$ eb init --tags mytag1=value1,mytag2=value2
```

Avec le AWS CLI ou d'autres clients basés sur une API, ajoutez des balises en utilisant le `--tags` paramètre de la commande. create-application

```
$ aws elasticbeanstalk create-application \
      --tags Key=mytag1,Value=value1 Key=mytag2,Value=value2 \
      --application-name my-app --version-label v1
```

Gestion des identifications d'une application existante

Vous pouvez ajouter, mettre à jour et supprimer des identifications dans une application Elastic Beanstalk existante.

Pour gérer les identifications d'une application dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.
3. Choisissez Actions, puis Manage Tags (Gérer les identifications).
4. Utilisez le formulaire à l'écran pour ajouter, mettre à jour ou supprimer des identifications.
5. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Si vous utilisez l'interface de ligne de commande EB pour mettre à jour votre application, utilisez eb tags pour ajouter, mettre à jour, supprimer ou répertorier des identifications.

Par exemple, la commande suivante répertorie les identifications dans une application.

```
~/workspace/my-app$ eb tags --list --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:application/my-app"
```

La commande suivante met à jour l'identification `mytag1` et supprime l'identification `mytag2`.

```
~/workspace/my-app$ eb tags --update mytag1=newvalue --delete mytag2 \
      --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:application/my-app"
```

Pour obtenir une liste complète des options et d'autres exemples, consultez `eb tags`.

Avec le AWS CLI ou d'autres clients basés sur une API, utilisez la list-tags-for-resource commande pour répertorier les balises d'une application.

```
$ aws elasticbeanstalk list-tags-for-resource --resource-arn
 "arn:aws:elasticbeanstalk:us-east-2:my-account-id:application/my-app"
```

Utilisez la commande update-tags-for-resource pour ajouter, mettre à jour ou supprimer des identifications dans une application.

```
$ aws elasticbeanstalk update-tags-for-resource \
      --tags-to-add Key=mytag1,Value=newvalue --tags-to-remove mytag2 \
      --resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:application/my-app"
```

Spécifiez les identifications à ajouter et les identifications à mettre à jour dans le paramètre `--tags-to-add` de update-tags-for-resource. Une identification inexistante est ajoutée et la valeur d'une identification existante est mise à jour.

> **Note**
>
> Pour utiliser certaines AWS CLI commandes et certaines de l'interface de ligne de commande EB avec une application Elastic Beanstalk, vous avez besoin de l'ARN de l'application. Vous pouvez extraire l'ARN à l'aide de la commande suivante.
>
> ```
> $ aws elasticbeanstalk describe-applications --application-names my-app
> ```

Étiquette des ressources d'application Elastic Beanstalk

Cette rubrique explique les avantages de l'utilisation de balises avec les ressources de votre application Elastic Beanstalk, ainsi que les contraintes associées à cette utilisation. Il explique également comment créer et gérer des balises pour les ressources de l'application.

Vous pouvez appliquer des balises aux ressources de vos AWS Elastic Beanstalk applications. Les balises sont des paires clé-valeur associées à des AWS ressources. Les identifications peuvent vous aider à classer les ressources. Ils sont particulièrement utiles si vous gérez de nombreuses ressources dans le cadre de plusieurs AWS applications.

Voici plusieurs méthodes d'utilisation de l'étiquette avec les ressources Elastic Beanstalk :

- Étapes du déploiement : identifiez les ressources associés aux différentes étapes de votre application, comme développement, bêta et production.
- Répartition des coûts — Utilisez les rapports de répartition des coûts pour suivre votre utilisation des AWS ressources associées aux différents comptes de dépenses. Ces rapports incluent les ressources labellisées et non labellisées, et regroupent les coûts en fonction des identifications. Pour savoir comment les rapports d'allocation des coûts utilisent les identifications, consultez Utilisation des identifications d'allocation des coûts pour les rapports de facturation personnalisés dans le Guide de l'utilisateur pour la gestion de la facturation et des coûts d'AWS .
- Contrôle d'accès : utilisez les identifications pour gérer les autorisations concernant les demandes et les ressources. Par exemple, un utilisateur qui ne peut créer et gérer que des environnements de test bêta ne doit avoir accès qu'aux ressources de l'étape bêta. Pour en savoir plus, consultez Utilisation de balises pour contrôler l'accès aux ressources Elastic Beanstalk.

Vous pouvez ajouter jusqu'à 50 identifications à chaque ressource. Les environnements sont légèrement différents : Elastic Beanstalk ajoute par défaut trois identifications système aux environnements, et vous ne pouvez pas modifier ou supprimer ces identifications. Outre les identifications par défaut, vous pouvez ajouter jusqu'à 47 identifications supplémentaires à chaque environnement.

Les contraintes suivantes s'appliquent aux clés et valeurs d'identification :

- Les clés et les valeurs peuvent contenir uniquement des lettres, des chiffres, des espaces et les symboles suivants : `_ . : / = + - @`
- Les clés peuvent contenir jusqu'à 127 caractères. Les valeurs peuvent contenir jusqu'à 255 caractères.

> **Note**
>
> Ces limites s'appliquent aux caractères Unicode en UTF-8. Pour les autres encodages multioctets, la limite peut être inférieure.

- Les clés sont sensibles à la casse.
- Les clés ne peuvent pas commencer par `aws:` ou `elasticbeanstalk:`.

Ressources que vous pouvez étiqueter

Vous trouverez ci-dessous les types de ressources Elastic Beanstalk que vous pouvez étiqueter, ainsi que des liens vers des rubriques relatives à la gestion des identifications pour chacune d'entre elles :

- Applications
- Environnements
- Versions de l'application
- Configurations enregistrées

Propagation des balises vers les modèles de lancement

Elastic Beanstalk propose la possibilité d'activer la propagation de balises d'environnement aux modèles de lancement. Cette option fournit une prise en charge continue du contrôle d'accès basé sur des balises (TBAC) avec les modèles de lancement.

> **Note**
>
> Les configurations de lancement sont progressivement supprimées et remplacées par des modèles de lancement. Pour plus d'informations, consultez la section Configurations de lancement dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling.

Pour éviter les interruptions de fonctionnement, les EC2 instances CloudFormation ne propagent pas les balises aux modèles de lancement existants. Si un cas d'utilisation nécessite des balises pour les ressources de votre environnement, vous pouvez activer Elastic Beanstalk

pour créer des modèles de lancement avec des balises pour ces ressources. Pour ce faire, définissez l'option `LaunchTemplateTagPropagationEnabled` dans l'espace de noms aws:autoscaling:launchconfiguration sur `true`. La valeur par défaut est `false`.

L'exemple de fichier de configuration suivant permet la propagation de balises pour lancer des modèles.

```
option_settings:
  aws:autoscaling:launchconfiguration:
    LaunchTemplateTagPropagationEnabled: true
```

Elastic Beanstalk peut uniquement propager des balises pour lancer des modèles pour les ressources suivantes :

- Volumes EBS
- EC2 instances
- EC2 interfaces réseau
- CloudFormation modèles de lancement qui définissent une ressource

Cette contrainte existe car les balises CloudFormation ne sont autorisées que pour des ressources spécifiques lors de la création de modèles. Pour plus d'informations, consultez TagSpecificationle guide de AWS CloudFormation l'utilisateur.

> **Important**
>
> - La modification de la valeur de cette option de `false` à `true` pour un environnement existant peut constituer un changement critique pour les balises existantes.
> - Lorsque cette fonctionnalité est activée, la propagation des balises doit être EC2 remplacée, ce qui peut entraîner des temps d'arrêt. Vous pouvez activer les mises à jour dynamiques pour appliquer les modifications de configuration par lots et éviter les temps d'arrêt de service pendant le processus de mise à jour. Pour de plus amples informations, veuillez consulter Configuration changes.

Pour en savoir plus sur l'utilisation des modèles de lancement, consultez ce qui suit :

- Lancez des modèles dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling

- Utilisation de modèles dans le Guide de l'utilisateur AWS CloudFormation
- Extraits de modèle Elastic Beanstalk dans le Guide de l'utilisateur AWS CloudFormation

Gestion des environnements Elastic Beanstalk

Ce chapitre explique comment créer et gérer vos environnements Elastic Beanstalk. Cette page d'introduction fournit un aperçu des mises à jour, de la maintenance et des configurations que vous appliquerez au fil du temps au fur et à mesure de l'évolution de votre application et de votre environnement.

Fonctions de l'environnement

Vous pouvez créer et gérer des environnements distincts pour le développement, les tests et l'utilisation en production, et vous pouvez déployer n'importe quelle version de votre application dans n'importe quel environnement. Les environnements peuvent être de longue durée ou temporaires. Lorsque vous mettez un environnement hors service, vous pouvez sauvegarder sa configuration pour le recréer ultérieurement.

Déploiements d'applications

Au cours du développement de votre application, vous allez la déployer souvent, probablement dans plusieurs environnements différents pour répondre à différents besoins. Elastic Beanstalk vous permet de configurer la façon dont les déploiements sont effectués. Vous pouvez effectuer le déploiement dans toutes les instances de votre environnement simultanément, ou fractionner le déploiement en lots via la propagation des déploiements.

Configuration changes

Les changements de configuration sont traités séparément des déploiements et ont leur propre champ d'application. Par exemple, si vous modifiez le type des instances EC2 qui exécutent votre application, toutes les instances doivent être remplacées. En revanche, si vous modifiez la configuration de l'équilibreur de charge de l'environnement, ce changement peut être effectué sur place, sans interrompre le service ni réduire la capacité. Vous pouvez également appliquer des changements de configuration qui modifient les instances de votre environnement par lots, via les mises à jour de configuration par propagation.

> **Note**
>
> Modifiez les ressources de votre environnement en utilisant seulement Elastic Beanstalk. Si vous modifiez des ressources à l'aide de la console, des commandes CLI ou d'Elastic Beanstalk d'un autre service SDKs, vous ne serez pas en mesure de surveiller avec précision

l'état de ces ressources, et vous ne pourrez pas enregistrer la configuration ou recréer l'environnement de manière fiable. De plus, les modifications hors-bande peuvent provoquer des problèmes lors de la mise à jour ou de la mise hors service d'un environnement.

Mises à jour de plateforme

Lorsque vous lancez un environnement, vous sélectionnez une version de plateforme. Nous mettons régulièrement à jour les plateformes vers de nouvelles versions de plateforme dans le but d'améliorer les performances et de lancer de nouvelles fonctionnalités. Vous pouvez mettre à jour votre environnement afin d'utiliser la dernière version de plateforme à tout moment. Consultez le guide des AWS Elastic Beanstalk plateformes pour obtenir une liste des plateformes prises en charge et un historique des versions de la plateforme qui inclut les plages de dates auxquelles elles étaient actuelles.

Options d'architecture

Lorsque votre application devient plus complexe, vous pouvez la diviser en plusieurs composants, chacun d'entre eux s'exécutant dans un environnement distinct. Pour les charges de travail longues, vous pouvez lancer des environnements de travail qui traitent des tâches à partir d'une file d'attente Amazon Simple Queue Service (Amazon SQS).

Rubriques

- Utilisation de la console de gestion de l'environnement Elastic Beanstalk
- Création d'un environnement Elastic Beanstalk
- Gestion de plusieurs environnements Elastic Beanstalk en tant que groupe avec l'interface de ligne de commande EB
- Déploiement d'applications dans des environnements Elastic Beanstalk
- Configuration changes
- Mise à jour de la version de la plateforme de votre environnement Elastic Beanstalk
- Annulation de mises à jour de configuration d'environnement et de déploiements d'application
- Reconstruction d'environnements Elastic Beanstalk
- Types d'environnement
- Environnements de travail Elastic Beanstalk
- Création de liens entre les environnements Elastic Beanstalk

- Restaurer votre environnement Elastic Beanstalk à partir d'un état non valide

Utilisation de la console de gestion de l'environnement Elastic Beanstalk

Cette section décrit comment gérer votre environnement Elastic Beanstalk à l'aide de la console de gestion de l'environnement. La console permet de gérer la configuration de votre environnement et d'effectuer des actions courantes. Celles-ci incluent le redémarrage des serveurs Web exécutés dans votre environnement, le clonage de votre environnement et la reconstruction de votre environnement à partir de zéro.

Rubriques

- Accès à la console de gestion de l'environnement
- Volet de présentation de l'environnement
- Détail de l'environnement
- Actions dans l'environnement

Accès à la console de gestion de l'environnement

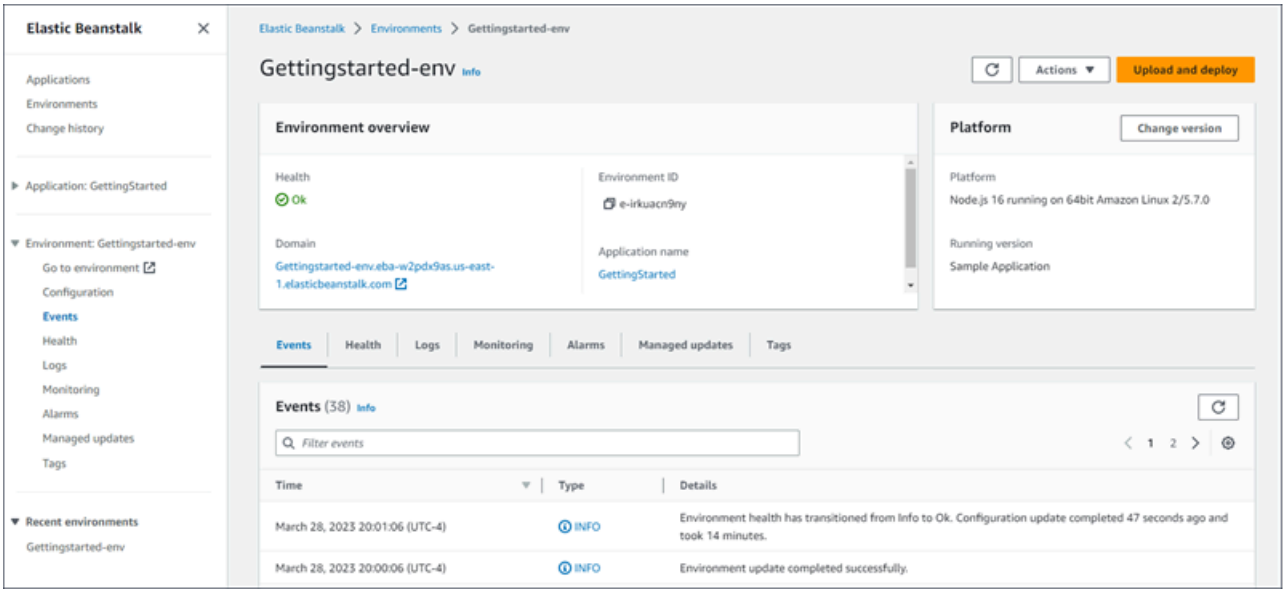
La procédure suivante décrit les étapes à suivre pour lancer la console de gestion de l'environnement.

Si vous êtes déjà connecté à la console Elastic Beanstalk, vous pouvez également lancer la page de gestion de l'environnement depuis le. Console de gestion d'application Sélectionnez un environnement dans la liste pour afficher les détails de la console de gestion de l'environnement sélectionné.

Accéder à la console de gestion de l'environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

L'image suivante illustre la console de gestion de l'environnement.



Le volet supérieur est la page de présentation de l'environnement. Il affiche des informations de haut niveau sur votre environnement.

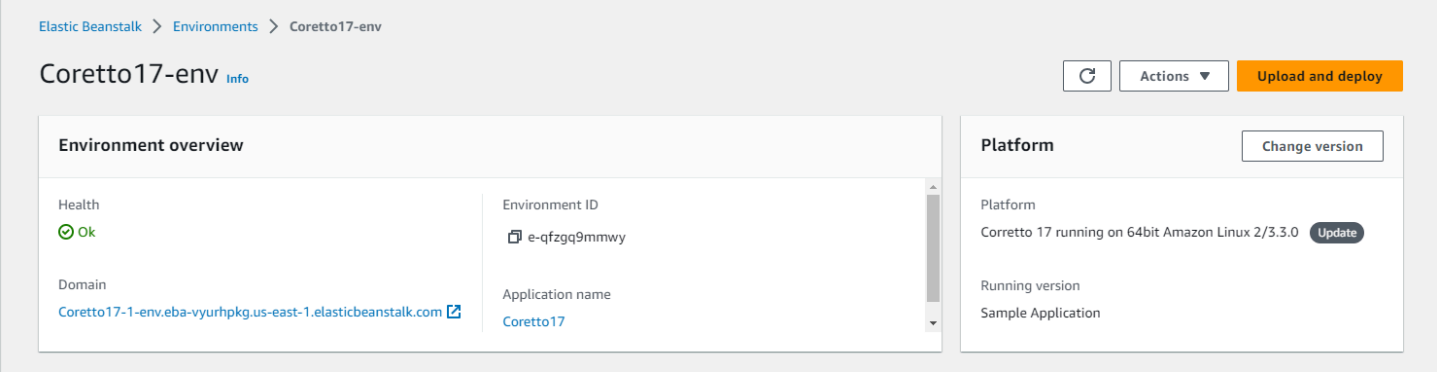
La moitié inférieure de la page affiche des onglets qui fournissent des informations plus détaillées. L'onglet Événements s'affiche par défaut. Les pages liées aux onglets sont également répertoriées dans le volet de navigation de gauche sous l'environnement.

Le volet de navigation de la console affiche le nom de l'application déployée dans l'environnement, ainsi que les pages de gestion des applications associées. Le nom de l'environnement est également affiché sur la page de navigation, suivi des pages de gestion de l'environnement. Les liens répertoriés sous le nom de l'environnement incluent également Accéder à l'environnement et Configuration, en plus des pages à onglets mentionnées précédemment.

Volet de présentation de l'environnement

Cette rubrique décrit les informations fournies par le volet de présentation de l'environnement. Il affiche des informations de haut niveau sur votre environnement et se trouve dans la moitié supérieure de la console de gestion de l'environnement.

L'image suivante affiche le volet de présentation de l'environnement.



Santé

État général de l'environnement. Si l'état de votre environnement se dégrade, le lien Afficher les causes s'affiche à côté de l'état de l'environnement. Sélectionnez ce lien pour afficher l'onglet État avec plus de détails.

Domain

Le domaine ou URL de l'environnement se trouve dans la partie supérieure de la page Environment overview (Présentation de l'environnement), sous Health (État) de l'environnement. Il s'agit de l'URL de l'application web que l'environnement exécute. Vous pouvez lancer l'application en sélectionnant l'URL.

ID de l'environnement

ID de l'environnement. Il s'agit d'un identifiant interne généré lors de la création de l'environnement.

Application name (Nom de l'application)

Nom de la version de l'application déployée et exécutée sur votre environnement.

Version en cours d'exécution

Nom de la version de l'application déployée et exécutée sur votre environnement. Sélectionnez Upload and deploy (Charger et déployer) pour charger un bundle source et le déployer dans votre environnement. Cette option permet de créer une nouvelle version d'application.

Plateforme

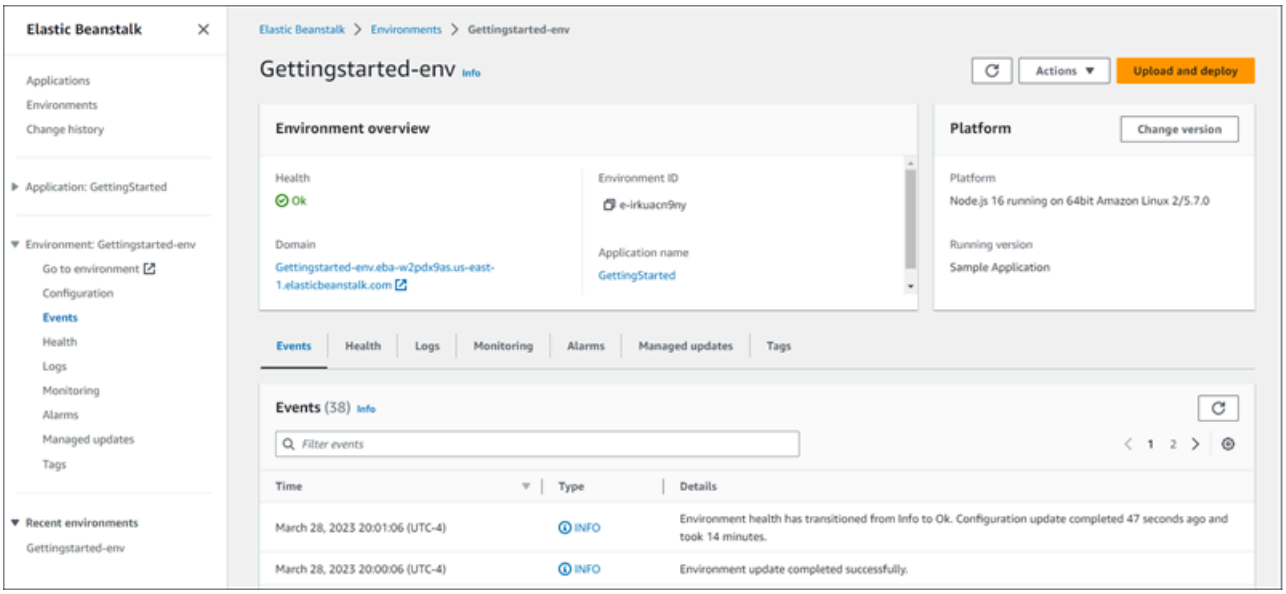
Nom de la version de plateforme en cours d'exécution sur votre environnement. En général, elle comprend l'architecture, le système d'exploitation (OS), le langage et le serveur d'applications (collectivement, la branche de la plateforme), avec un numéro de version de plateforme spécifique.

Si la version de votre plateforme n'est pas la plus récente, une étiquette d'état s'affiche à côté d'elle dans la section Plateforme. L'étiquette Mettre à jour indique que, bien que la version de plateforme soit prise en charge, une version plus récente est disponible. La version de plateforme peut également être étiquetée comme Obsolète ou Mise hors service. Sélectionnez Modifier la version pour mettre à jour la branche de votre plateforme vers une version plus récente. Pour plus d'informations sur les états d'une version de plateforme, consultez la section Branche de plateforme dans le Glossaire des plateformes Elastic Beanstalk. L'image précédente de cette page illustre l'étiquette d'état de mise à jour pour la plate-forme donnée.

Détail de l'environnement

Cette rubrique décrit les informations supplémentaires fournies par la console de gestion de l'environnement à partir du volet de navigation de gauche et des pages à onglets.

L'image suivante illustre la console de gestion de l'environnement.



La partie inférieure de la console de gestion de l'environnement répertorie les onglets qui fournissent des informations plus détaillées et variées sur l'environnement. Vous pouvez sélectionner la page à onglets ou l'étiquette de page dans le volet de navigation de gauche.

Dans le volet de navigation gauche de la console, sous le nom de l'environnement, deux choix ne figurent pas dans les pages à onglets. Il s'agit d'accéder à l'environnement et à la configuration.

> **Note**
>
> Sélectionnez Accéder à l'environnement pour lancer votre application.

Configuration

Utilisez la page Configuration dans le volet de navigation de gauche pour afficher et mettre à jour les paramètres de configuration actuels de votre environnement et de ses ressources. Cela inclut la configuration réseau, la configuration de la base de données, l'équilibrage de charge, les notifications, les paramètres de surveillance de l'état, la configuration des mises à jour de la plate-forme gérée, la configuration du déploiement, le streaming des journaux d'instance, CloudWatch l'intégration AWS X-Ray, les paramètres du serveur proxy, les propriétés de l'environnement et les options spécifiques à la plate-forme. Utilisez les paramètres figurant sur cette page pour personnaliser le comportement de votre environnement au cours des déploiements, activer des fonctionnalités supplémentaires, et modifier le type d'instance et d'autres paramètres que vous avez choisis lors de la création de l'environnement.

Pour de plus amples informations, veuillez consulter Configuration d'environnements Elastic Beanstalk.

Événements

La page Events (Événements) affiche le flux d'événements relatif à votre environnement. Elastic Beanstalk émet des messages d'événement chaque fois que vous interagissez avec l'environnement et lorsque l'une des ressources de votre environnement est créée ou modifiée en conséquence.

Pour de plus amples informations, veuillez consulter Affichage du flux d'événements d'un environnement Elastic Beanstalk.

Santé

Si la surveillance améliorée de l'état est activée, cette page répertorie les EC2 instances de votre environnement et les informations de santé en temps réel pour chaque instance.

La page Santé globale affiche les données de santé sous forme de moyenne pour toutes les instances de votre environnement combinées.

Le volet Enhanced instance health affiche des informations de santé en temps réel pour chaque EC2 instance individuelle de votre environnement. Cette fonctionnalité permet à Elastic Beanstalk de surveiller attentivement les ressources de votre environnement, afin d'évaluer plus précisément l'état de votre application.

Lorsque la surveillance améliorée de l'état est activée, cette page affiche des informations sur les demandes envoyées par les instances de votre environnement et sur les métriques provenant du système d'exploitation, dont la latence, le chargement et l'utilisation de l'UC.

Pour de plus amples informations, veuillez consulter Rapports et surveillance de l'état de santé améliorés dans Elastic Beanstalk.

Journaux

La page Logs vous permet de récupérer les journaux EC2 des instances de votre environnement. Lorsque vous demandez des journaux, Elastic Beanstalk envoie une commande aux instances, qui les chargent ensuite dans votre compartiment de stockage Elastic Beanstalk dans Amazon S3. Lorsque vous demandez des journaux sur cette page, Elastic Beanstalk les supprime automatiquement d'Amazon S3 après 15 minutes.

Vous pouvez également configurer les instances de votre environnement afin de charger les journaux dans Amazon S3 pour un stockage permanent une fois qu'ils ont fait l'objet d'une rotation au niveau local.

Pour de plus amples informations, veuillez consulter Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk.

Contrôle

La page Surveillance présente l'ensemble des informations relatives à l'état de votre environnement. Cela inclut l'ensemble de mesures par défaut fourni par Elastic Load Balancing et Amazon EC2, ainsi que des graphiques illustrant l'évolution de l'état de l'environnement au fil du temps.

Pour de plus amples informations, veuillez consulter Surveillance de l'état de l'environnement dans la console AWS de gestion.

alertes

La page Existing alarms (Alarmes existantes) affiche des informations relatives aux alarmes que vous avez configurées pour votre environnement. Vous pouvez utiliser les options figurant sur cette page pour créer ou supprimer des alarmes.

Pour de plus amples informations, veuillez consulter Gestion des alarmes.

Mises à jour gérées

La page Mises à jour gérées affiche des informations sur les mises à jour de plateforme gérées à venir et achevées et sur le remplacement des instances.

Ces fonctionnalités de mis à jour gérée vous permettent de configurer votre environnement afin de le mettre à jour automatiquement vers la dernière version de plateforme, au cours de la fenêtre

de maintenance hebdomadaire de votre choix. Entre les versions de la plateforme, vous pouvez choisir de demander à votre environnement de remplacer toutes ses EC2 instances Amazon pendant la période de maintenance. Cela peut remédier aux problèmes qui se produisent lorsque votre application est exécutée pendant de longues périodes.

Pour de plus amples informations, veuillez consulter Mises à jour gérées de la plateforme.

Étiquettes

La page Tags (Balises) affiche les balises appliquées par Elastic Beanstalk à l'environnement lors de sa création, ainsi que toutes les balises ajoutées. Vous pouvez ajouter, modifier et supprimer des balises personnalisées. Vous ne pouvez pas modifier ou supprimer les balises appliquées par Elastic Beanstalk.

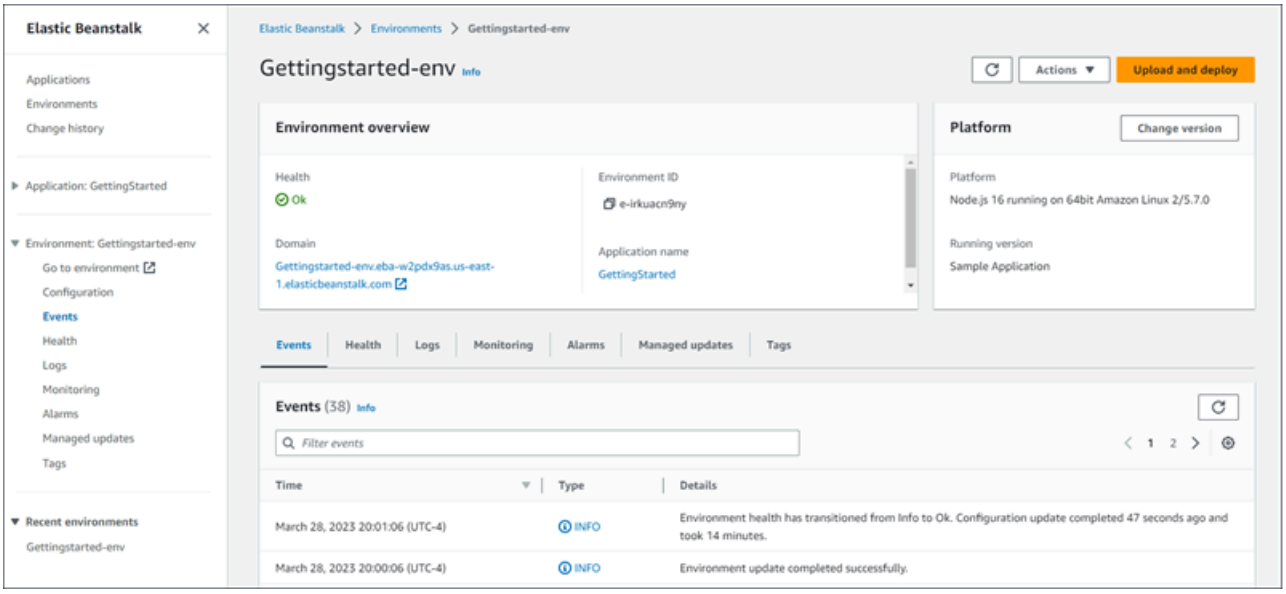
Des balises d'environnement sont appliquées à toutes les ressources créées par Elastic Beanstalk pour prendre en charge votre application.

Pour de plus amples informations, veuillez consulter Balisage des ressources dans vos environnements Elastic Beanstalk.

Actions dans l'environnement

Cette rubrique décrit les opérations courantes que vous pouvez sélectionner pour effectuer sur votre environnement dans le menu déroulant Actions de la console de gestion de l'environnement.

L'image suivante illustre la console de gestion de l'environnement. Le menu déroulant Actions se trouve sur le côté droit de l'en-tête qui affiche le nom de l'environnement, à côté du bouton Actualiser.



> **Note**
>
> Certaines actions ne sont disponibles que dans certaines conditions et restent désactivées jusqu'à ce que ces conditions soient réunies.

Charger la configuration

Chargez une configuration précédemment enregistrée. Les configurations sont enregistrées dans votre application et peuvent être chargées par tout environnement associé. Si vous avez modifié la configuration de votre environnement, vous pouvez charger une configuration enregistrée afin d'annuler ces modifications. Vous pouvez également charger une configuration que vous avez enregistrée à partir d'un autre environnement exécutant la même application afin de propager les modifications de configuration dans ces environnements.

Enregistrer la configuration

Enregistrez la configuration actuelle de votre environnement dans votre application. Avant de modifier la configuration de votre environnement, enregistrez la configuration actuelle de façon à pouvoir la restaurer ultérieurement, si nécessaire. Vous pouvez également appliquer une configuration enregistrée lorsque vous lancez un nouvel environnement.

Domaines de l'environnement d'échange (URLs)

Echangez le CNAME de l'environnement actuel avec celui d'un nouvel environnement. Après un échange CNAME, l'ensemble du trafic vers l'application utilisant l'URL de l'environnement est acheminé vers le nouvel environnement. Lorsque vous êtes prêt à déployer une nouvelle version de votre application, vous pouvez lancer un environnement distinct dans le cadre de la nouvelle version. Lorsque le nouvel environnement est prêt à recevoir des demandes, effectuez un échange CNAME pour commencer à acheminer le trafic vers le nouvel environnement. Cette action n'interrompt pas vos services. Pour de plus amples informations, veuillez consulter Déploiements bleu/vert avec Elastic Beanstalk.

Environnement de clonage

Lancez un nouvel environnement avec la même configuration que celui en cours d'exécution.

Clone with latest platform (Cloner avec la plateforme la plus récente)

Clonez votre environnement actuel avec la dernière version de la plateforme Elastic Beanstalk en cours d'utilisation. Cette option n'est disponible que lorsqu'une version plus récente de la plateforme de l'environnement actuel est disponible.

Abort Current Operation (Annuler l'opération en cours)

Arrêtez une mise à jour d'environnement en cours. Suite à l'arrêt d'une opération, l'état de certaines instances de votre environnement peut être différent des autres, en fonction du niveau d'avancement de l'opération au moment où elle a été arrêtée. Cette option n'est disponible que lorsque votre environnement est en cours de mise à jour.

Restart App Servers (Redémarrer les serveurs d'applications)

Redémarrez le serveur Web qui est en cours d'exécution sur les instances de votre environnement. Cette option n'arrête ni ne redémarre aucune AWS ressource. Si votre environnement se comporte de façon étrange suite à l'envoi de demandes incorrectes, le redémarrage du serveur d'application peut restaurer temporairement la fonctionnalité, le temps d'identifier la cause première et de résoudre le problème.

Rebuild Environment (Reconstruire l'environnement)

Suspendez toutes les ressources de l'environnement en cours d'exécution et créez un nouvel environnement avec les mêmes paramètres. Cette opération prend plusieurs minutes, soit à peu près le temps nécessaire au déploiement d'un nouvel environnement à partir de zéro. Lors d'une reconstruction, toutes les instances Amazon RDS qui sont exécutées dans la couche Données de votre environnement sont supprimées. Si vous avez besoin des données, créez un instantané. Vous pouvez créer un instantané manuellement dans la console RDS ou configurer la stratégie de suppression de votre couche Données afin de créer un instantané automatiquement avant de supprimer l'instance. Il s'agit du paramètre par défaut lorsque vous créez une couche Données.

Terminate Environment (Supprimer l'environnement)

Résiliez toutes les ressources de l'environnement en cours d'exécution et supprimez l'environnement de l'application. Si une instance RDS est exécutée dans une couche Données et que vous devez conserver ses données, vérifiez que la stratégie de suppression de base de données est définie sur `Snapshot` ou sur `Retain`. Pour de plus amples informations, consultez Cycle de vie de base de données dans le chapitre Configuration des environnements de ce guide.

Création d'un environnement Elastic Beanstalk

La procédure suivante lance un nouvel environnement exécutant l'application par défaut. Ces étapes sont simplifiées afin que votre environnement soit rapidement opérationnel, à l'aide de valeurs d'option par défaut.

> **Remarque concernant les autorisations**
>
> La création d'un environnement nécessite les autorisations de la stratégie gérée d'accès complet d'Elastic Beanstalk. Consultez Stratégie utilisateur Elastic Beanstalk pour plus de détails.

Pour lancer un environnement avec une application (console)

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le volet de navigation, choisissez Applications. Sélectionnez une application existante dans la liste. Vous pouvez également choisir d'en créer un, en suivant les instructions deGestion des applications .
3. Sur la page de présentation de l'application, choisissez Create environment.

   Cela lance l'assistant de création d'environnement. L'assistant fournit un ensemble d'étapes vous permettant de créer un nouvel environnement.
4. Pour le niveau environnement, choisissez le niveau d'environnement de serveur Web ou le niveau d'environnement de travail. Vous ne pouvez pas modifier le niveau d'un environnement après sa création.

   > **Note**
   >
   > La plateforme .NET sur Windows Server ne prend pas en charge le niveau d'environnement worker.

   Les champs d'informations de l'application sont définis par défaut, en fonction de l'application que vous avez sélectionnée précédemment.

Dans le groupe Informations sur l'environnement, le nom de l'environnement est défini par défaut, en fonction du nom de l'application. Si vous préférez un autre nom d'environnement, vous pouvez saisir une autre valeur dans le champ. Vous pouvez éventuellement saisir un nom de domaine, sinon Elastic Beanstalk génère automatiquement une valeur. Vous pouvez également éventuellement saisir une description de l'environnement.

5. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.

> **Note**
>
> Elastic Beanstalk prend en charge plusieurs versions pour la plupart des plateformes répertoriées. Par défaut, la console sélectionne la version recommandée pour la plateforme et la branche de plateforme que vous choisissez. Si votre application nécessite une version différente, vous pouvez la sélectionner ici. Pour plus d'informations sur les versions de plateforme prises en charge, consultez the section called “Plateformes prises en charge”.

6. Pour le code d'application, vous avez plusieurs choix pour continuer.
   - Pour lancer l'exemple d'application par défaut sans fournir le code source, choisissez Exemple d'application. Cette action permet de sélectionner l'application d'une seule page fournie par Elastic Beanstalk pour la plate-forme que vous avez sélectionnée précédemment.
   - Si vous avez téléchargé un exemple d'application à partir de ce guide, ou si vous avez votre propre code source pour une application, procédez comme suit.
     a. Sélectionnez Téléverser votre code.
     b. Choisissez ensuite Fichier local, puis sous Télécharger l'application, sélectionnez Choisir un fichier.
     c. Le système d'exploitation de votre machine cliente vous présentera une interface permettant de sélectionner le fichier local que vous avez téléchargé. Sélectionnez le fichier du bundle source et continuez.
7. Le choix des préréglages dépend de votre objectif pour l'environnement.
   - Si vous créez un exemple d'environnement pour en savoir plus sur Elastic Beanstalk ou un environnement de développement, choisissez Instance unique (éligible au niveau gratuit).

- Si vous créez un environnement de production ou un environnement pour en savoir plus sur l'équilibrage de charge, choisissez l'une des options de haute disponibilité.

8. Choisissez Suivant.

Pour configurer l'accès au service

Ensuite, vous avez besoin de deux rôles. Un rôle de service permet à Elastic Beanstalk de surveiller vos instances EC2 et de mettre à niveau la plateforme de votre environnement. Un rôle de profil d'instance EC2 permet des tâches telles que la rédaction de journaux et l'interaction avec d'autres services.

Pour créer ou sélectionner le rôle de service

1. Si vous avez déjà créé un rôle de service et que vous souhaitez en choisir un existant, sélectionnez la valeur dans le menu déroulant Rôle de service et ignorez le reste de ces étapes pour créer un rôle de service.
2. Si aucune valeur n'est répertoriée pour le rôle de service, ou si vous souhaitez en créer un nouveau, passez aux étapes suivantes.
3. Pour Rôle de service, choisissez Créer un rôle.
4. Pour Type d’entité de confiance, choisissez Service AWS .
5. Dans le cas d'utilisation, choisissez Elastic Beanstalk — Environment.
6. Choisissez Suivant.
7. Vérifiez que les politiques d'autorisations incluent les éléments suivants, puis choisissez Next :

   - `AWSElasticBeanstalkEnhancedHealth`
   - `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy`

8. Choisissez Créer un rôle.
9. Retournez à l'onglet Configurer l'accès au service, actualisez la liste, puis sélectionnez le rôle de service nouvellement créé.

Pour créer ou sélectionner un profil d'instance EC2

1. Si vous avez déjà créé un profil d'instance EC2 et que vous souhaitez en choisir un existant, sélectionnez la valeur dans le menu déroulant du profil d'instance EC2 et ignorez le reste de ces étapes pour créer un profil d'instance EC2.

2. Si aucune valeur n'est répertoriée pour le profil d'instance EC2, ou si vous souhaitez en créer un nouveau, passez aux étapes suivantes.
3. Choisissez Créer un rôle.
4. Pour Type d’entité de confiance, choisissez Service AWS .
5. Dans le cas d'utilisation, choisissez Elastic Beanstalk — Compute.
6. Choisissez Suivant.
7. Vérifiez que les politiques d'autorisations incluent les éléments suivants, puis choisissez Next :
   - `AWSElasticBeanstalkWebTier`
   - `AWSElasticBeanstalkWorkerTier`
   - `AWSElasticBeanstalkMulticontainerDocker`
8. Choisissez Créer un rôle.
9. Retournez à l'onglet Configurer l'accès au service, actualisez la liste, puis sélectionnez le profil d'instance EC2 nouvellement créé.

Pour terminer la configuration et la création de votre application

1. (Facultatif) Si vous avez déjà créé une paire de clés EC2, vous pouvez la sélectionner dans le menu déroulant du champ de paire de clés EC2. Vous l'utiliseriez pour vous connecter en toute sécurité à l'instance Amazon EC2 fournie par Elastic Beanstalk pour votre application. Si vous ignorez cette étape, vous pouvez toujours créer et attribuer une paire de clés EC2 une fois l'environnement créé. Pour de plus amples informations, veuillez consulter EC2 paire de clés.
2. Choisissez Skip to Review (Passer à la révision) sur la page Configure service access (Configurer l’accès au service).
3. La page Review (Révision) affiche un résumé de tous vos choix.

   Pour personnaliser davantage votre environnement, choisissez Edit (Modifier) à côté de l'étape qui inclut les éléments que vous souhaitez configurer. Les options suivantes peuvent être définies uniquement lors de la création de l'environnement :

   - Nom de l'environnement
   - Nom de domaine
   - Version de la plateforme
   - Processeur

- Type d'équilibreur de charge
- Palier

Vous pouvez modifier les paramètres suivants après la création de l'environnement, mais ils requièrent la mise en œuvre de nouvelles instances ou d'autres ressources et leur application peut prendre du temps :

- Type d'instance, volume racine, paire de clés et rôle Gestion des identités et des accès AWS (IAM)
- Base de données interne Amazon RDS
- VPC

Pour de plus amples informations sur tous les paramètres disponibles, veuillez consulter Assistant de création d'un environnement.

4. Choisissez Submit (Soumettre) en bas de la page pour initialiser la création de votre nouvel environnement.

Alors qu'Elastic Beanstalk crée votre environnement, vous êtes redirigé vers la console Elastic Beanstalk. Une fois que l'intégrité de l'environnement devient verte, choisissez l'URL à côté du nom de l'environnement pour afficher l'application en cours d'exécution. Cette URL est généralement accessible depuis Internet, sauf si vous configurez votre environnement afin d'utiliser un VPC personnalisé avec un équilibreur de charge interne.

Rubriques

- Assistant de création d'un environnement
- Clonage d'un environnement Elastic Beanstalk
- Arrêt d'un environnement Elastic Beanstalk
- Création d'environnements Elastic Beanstalk avec la CLI AWS
- Création d'environnements Elastic Beanstalk avec l'API
- Construction d'une URL Launch Now
- Création et mise à jour de groupes d'environnements Elastic Beanstalk

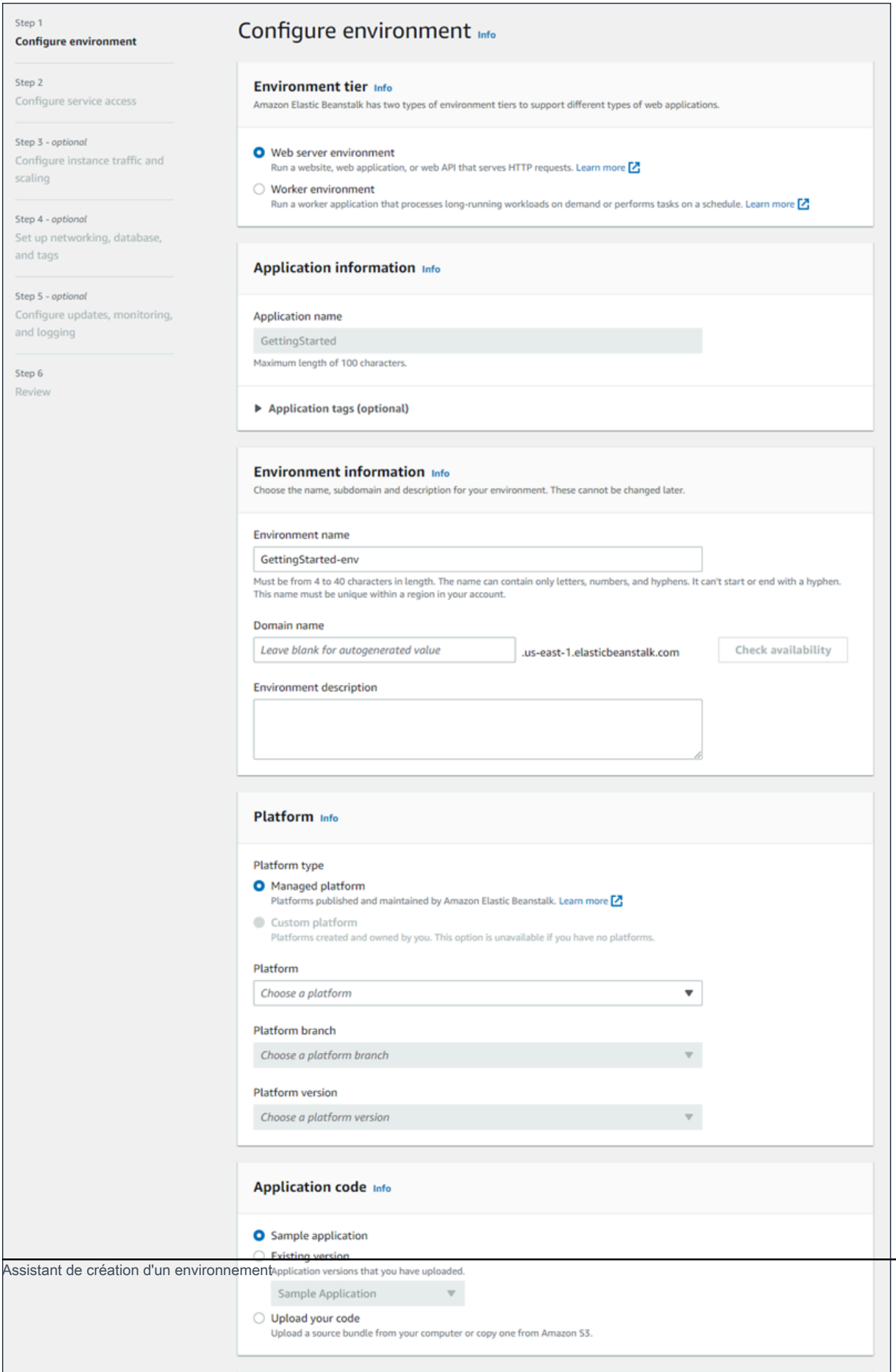
Assistant de création d'un environnement

Cette rubrique décrit l'assistant Create a new environment (Créer un nouvel environnement) ainsi que l'ensemble des méthodes auxquelles vous pouvez avoir recours pour configurer l'environnement que vous souhaitez créer.

> **Note**
>
> Création d'un environnement Elastic BeanstalkNous montrons ici comment lancer l'assistant de création d'environnement et créer rapidement un environnement avec des valeurs par défaut et des paramètres recommandés. Cette rubrique actuelle vous expliquera toutes les options.

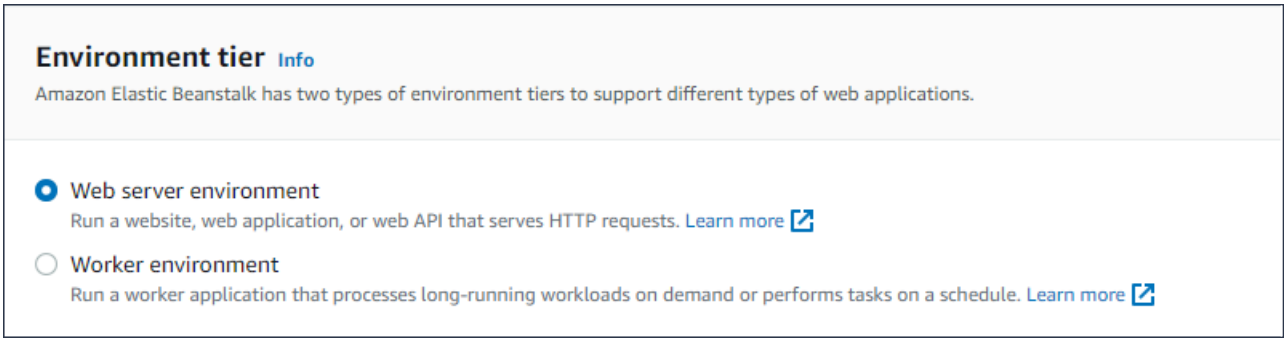
Page de l'assistant

L'assistant Create environment (Créer un environnement) fournit un ensemble d'étapes vous permettant de créer un nouvel environnement.


Step 1
Configure environment
Step 2
Configure service access
Step 3 - optional
Configure instance traffic and scaling
Step 4 - optional
Set up networking, database, and tags
Step 5 - optional
Configure updates, monitoring, and logging
Step 6
Review
Configure environment Info
Environment tier Info
Amazon Elastic Beanstalk has two types of environment tiers to support different types of web applications.
Web server environment
Run a website, web application, or web API that serves HTTP requests. Learn more
Worker environment
Run a worker application that processes long-running workloads on demand or performs tasks on a schedule. Learn more
Application information Info
Application name
GettingStarted
Maximum length of 100 characters.
Application tags (optional)
Environment information Info
Choose the name, subdomain and description for your environment. These cannot be changed later.
Environment name
GettingStarted-env
Must be from 4 to 40 characters in length. The name can contain only letters, numbers, and hyphens. It can't start or end with a hyphen. This name must be unique within a region in your account.
Domain name
Leave blank for autogenerated value
.us-east-1.elasticbeanstalk.com
Check availability
Environment description
Platform Info
Platform type
Managed platform
Platforms published and maintained by Amazon Elastic Beanstalk. Learn more
Custom platform
Platforms created and owned by you. This option is unavailable if you have no platforms.
Platform
Choose a platform
Platform branch
Choose a platform branch
Platform version
Choose a platform version
Application code Info
Sample application
Existing version
Application versions that you have uploaded.
Sample Application
Upload your code
Upload a source bundle from your computer or copy one from Amazon S3.

Niveau environnemental

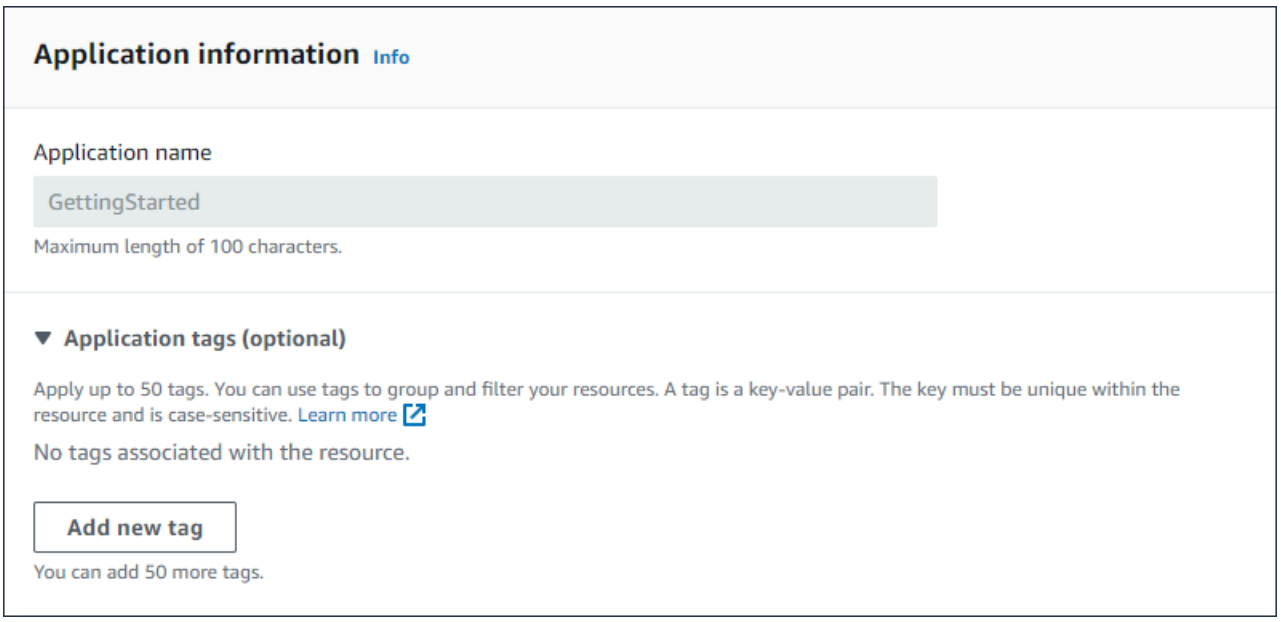
Ensuite, pour le niveau d'environnement, choisissez l'environnement de serveur web ou le niveau d'environnement de travail. Vous ne pouvez pas modifier le niveau d'un environnement après sa création.



> **Note**
>
> La plateforme .NET sur Windows Server ne prend pas en charge le niveau d'environnement worker.

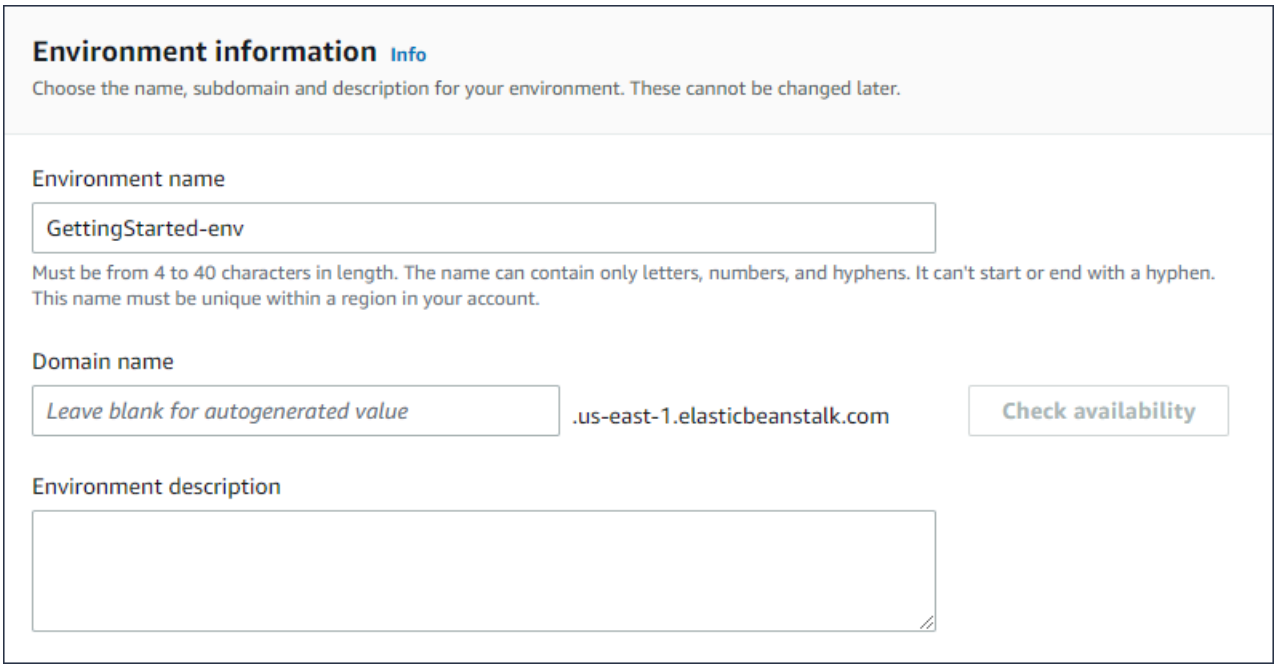
Informations sur l'application

Si vous avez lancé l'assistant en sélectionnant Create new environment (Créer un nouvel environnement) à partir de la page Application overview (Présentation de l'application), le nom de l'application est prérempli. Sinon, entrez un nom d'application. Ajoutez éventuellement des balises d'application.



Informations sur l'environnement

Définissez le nom et le sous-domaine de l'environnement et créez une description pour votre environnement. N'oubliez pas que les paramètres de cet environnement ne peuvent plus être modifiés une fois l'environnement créé.



- Name (Nom) : entrez le nom de l'environnement. Le formulaire fournit un nom généré.
- Domain (Domaine) : (environnements de serveur web) entrez un nom de domaine unique pour votre environnement. Le nom par défaut est le nom de l'environnement. Vous pouvez entrer un

nom de domaine différent. Elastic Beanstalk utilise ce nom pour créer un CNAME unique pour l'environnement. Pour vérifier si le nom de domaine souhaité est disponible, choisissez Vérifier la disponibilité.

- Description : entrez la description de cet environnement.

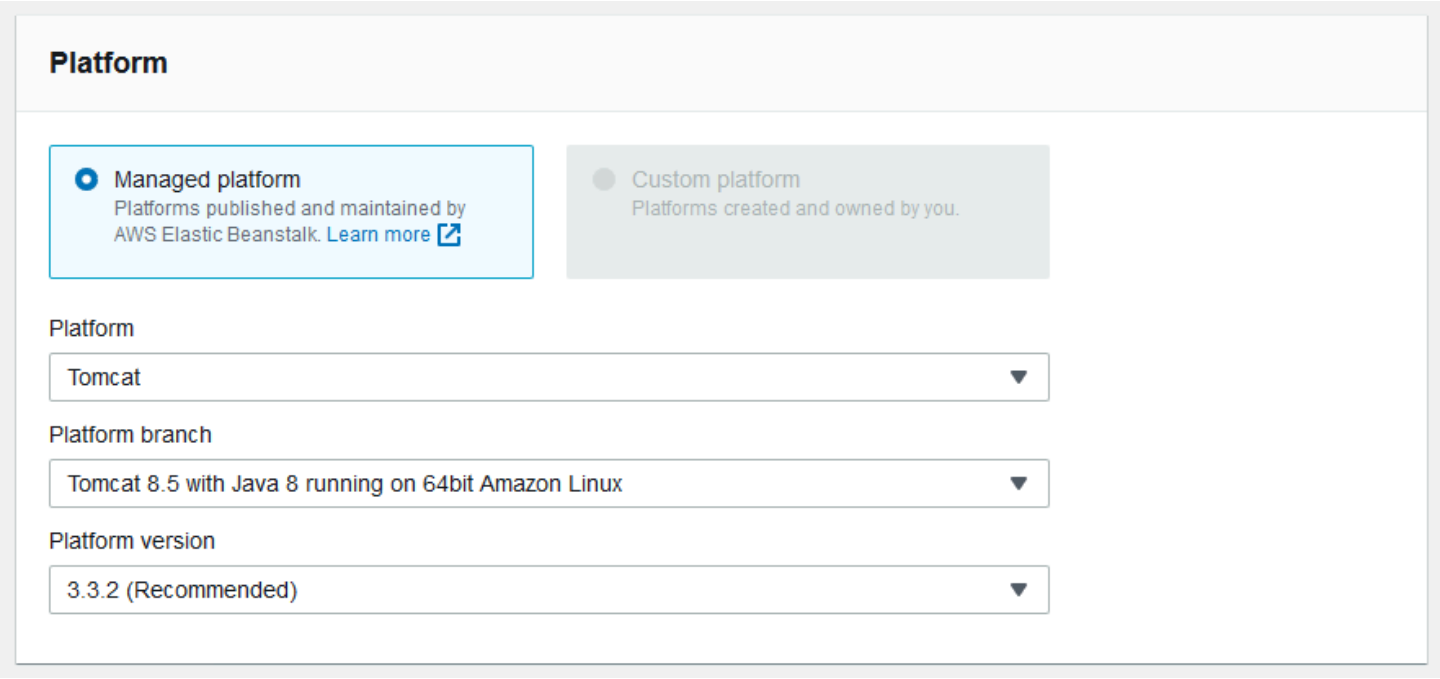
Sélection d'une plateforme pour le nouvel environnement

Vous pouvez créer un nouvel environnement à partir de deux types de plateformes :

- Plateforme gérée
- Plateforme personnalisée

Plateforme gérée

Dans la plupart des cas, vous utilisez une plateforme gérée Elastic Beanstalk pour votre nouvel environnement. Lorsque l'assistant de création d'un environnement démarre, il sélectionne par défaut l'option Managed platform (Plateforme gérée).



Sélectionnez une plateforme, une branche de plateforme au sein de cette plateforme et une version de plateforme spécifique dans la branche. Lorsque vous sélectionnez une branche de plateforme, la version recommandée dans la branche est sélectionnée par défaut. En outre, vous pouvez sélectionner n'importe quelle version de plateforme que vous avez utilisée auparavant.

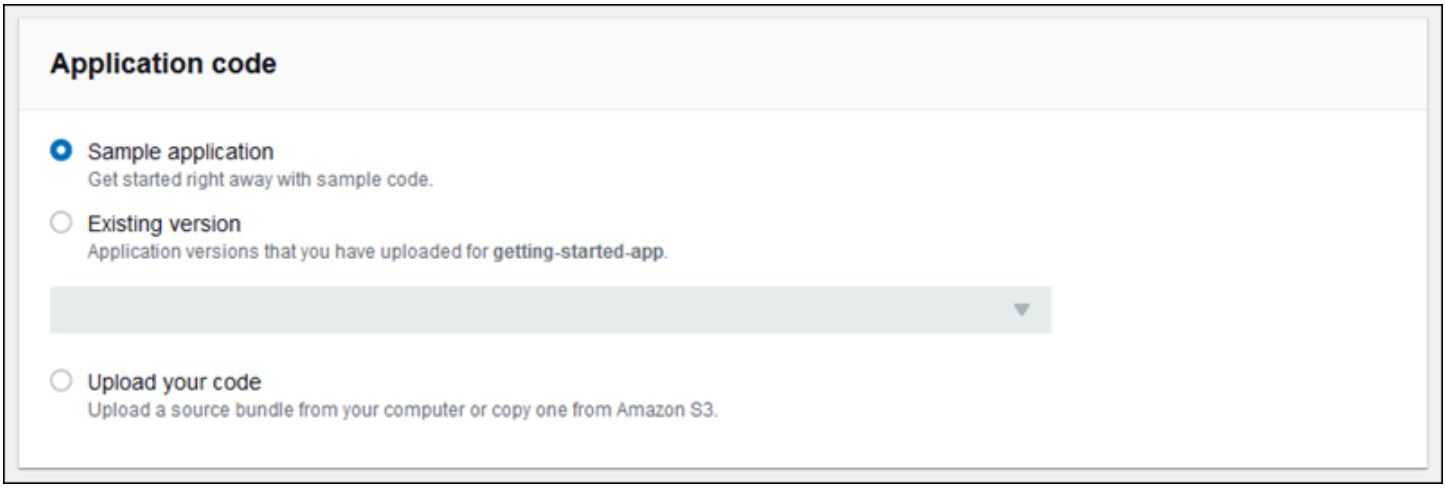
> **Note**
>
> Pour un environnement de production, nous vous recommandons de choisir une version de plateforme dans une branche de plateforme prise en charge. Pour plus d'informations sur les états de branche de plateforme, consultez la définition d'une branche de plate-forme dans le the section called “Glossaire des plateformes”.

Plateforme personnalisée

Si une off-the-shelf plateforme ne répond pas à vos besoins, vous pouvez créer un nouvel environnement à partir d'une plateforme personnalisée. Pour spécifier une plateforme personnalisée, choisissez l'option Custom platform (Plateforme personnalisée), puis sélectionnez l'une des plateformes personnalisées disponibles. Si aucune plateforme personnalisée n'est disponible, cette option est grisée.

Fourniture du code d'application

Maintenant que vous avez sélectionné la plateforme à utiliser, l'étape suivante consiste à fournir votre code d'application.



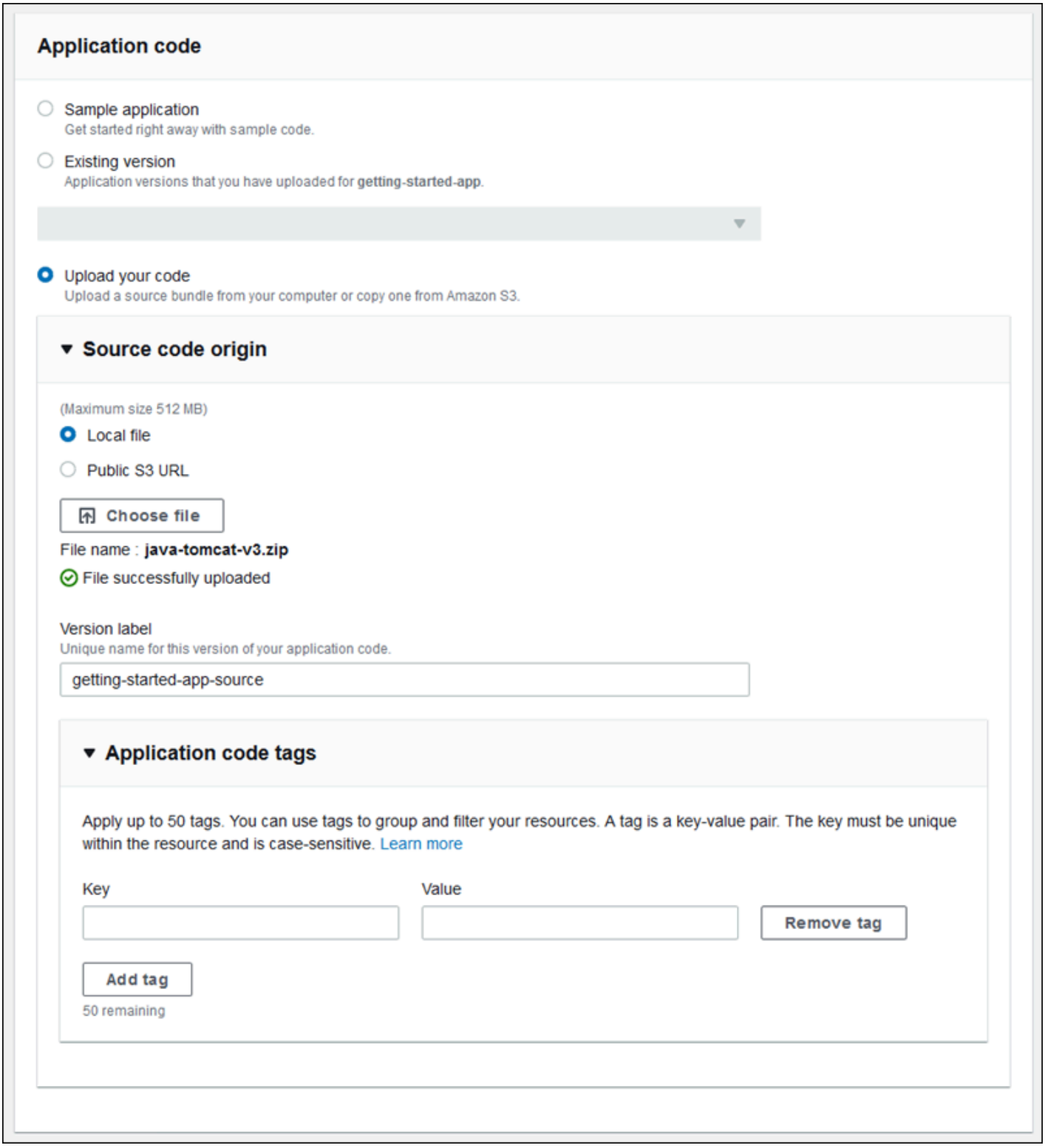
Vous avez plusieurs options:

- Vous pouvez utiliser l'exemple d'application fourni par Elastic Beanstalk pour chaque plateforme.
- Vous pouvez utiliser le code que vous avez déjà déployé dans Elastic Beanstalk. Choisissez Version existante et votre application dans la section Code de l'application.

- Vous pouvez charger un nouveau code. Sélectionnez Charger votre code, puis Charger. Vous pouvez charger un nouveau code d'application à partir d'un fichier local, ou spécifier l'URL du compartiment Amazon S3 contenant le code de votre application.

  > **Note**
  >
  > En fonction de la version de plateforme que vous avez sélectionnée, vous pouvez charger votre application dans un bundle source au format ZIP, un fichier WAR ou une configuration Docker en texte brut. La taille du fichier ne doit pas dépasser 500 Mo.

  Lorsque vous choisissez de charger un nouveau code, vous pouvez également fournir les balises à associer à votre nouveau code. Pour plus d'informations sur le balisage d'une version d'application, consultez the section called “Étiquette des versions d'application”.



Pour créer rapidement un environnement à l'aide des options de configuration par défaut, vous pouvez maintenant choisir Create environment (Créer un environnement). Choisissez Configurer

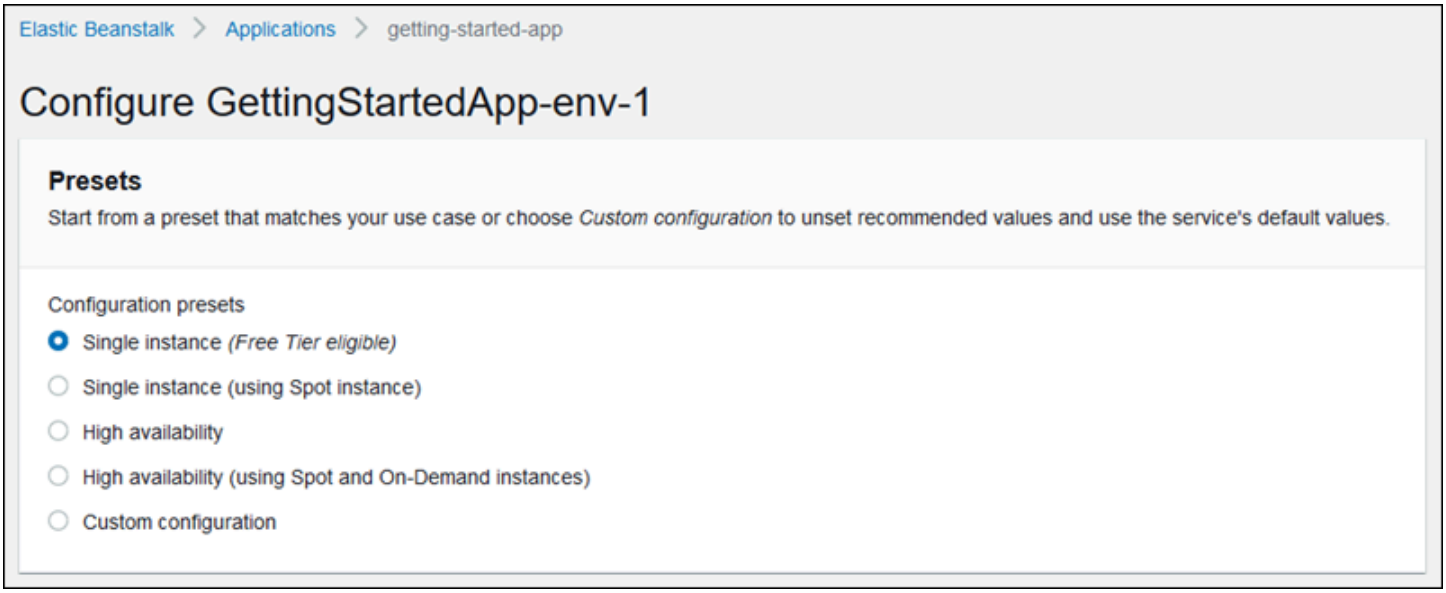
plus d'options pour apporter d'autres modifications de configuration, comme décrit dans les sections suivantes.

Page de configuration de l'Assistant

Lorsque vous choisissez Configure more options (Configurer d'autres options), l'Assistant affiche la page Configure (Configurer). Sur cette page, vous pouvez sélectionner une configuration prédéfinie, modifier la version de la plateforme que vous souhaitez utiliser pour votre environnement ou faire des choix de configuration spécifiques pour le nouvel environnement.

Choisir une configuration prédéfinie

Dans la section Presets (Préréglages) de la page, Elastic Beanstalk fournit plusieurs préréglages de configuration pour différents cas d'utilisation. Chaque préréglage inclut les valeurs recommandées pour plusieurs options de configuration.



Les préréglages haute disponibilité comprennent un équilibreur de charge et sont recommandés pour les environnements de production. Choisissez-les si vous souhaitez un environnement capable d'exécuter plusieurs instances pour une haute disponibilité et d'évoluer en fonction de la charge. Les préréglages pour instance unique sont principalement recommandés pour le développement. Deux des paramètres prédéfinis permettent d'activer les demandes d'instance Spot. Pour de plus amples informations sur la configuration de capacité Elastic Beanstalk, veuillez consulter Groupe Auto Scaling.

Le dernier préréglage, Custom configuration (Configuration personnalisée) supprime toutes les valeurs recommandées à l'exception des paramètres de rôle et utilise les valeurs par défaut de

l'API. Choisissez cette option si vous déployez un bundle source avec des fichiers de configuration qui définissent les options de configuration. L'option Configuration personnalisée est également sélectionnée automatiquement si vous modifiez les préréglages de configuration Faible coût ou Haute disponibilité.

Personnalisation de votre configuration

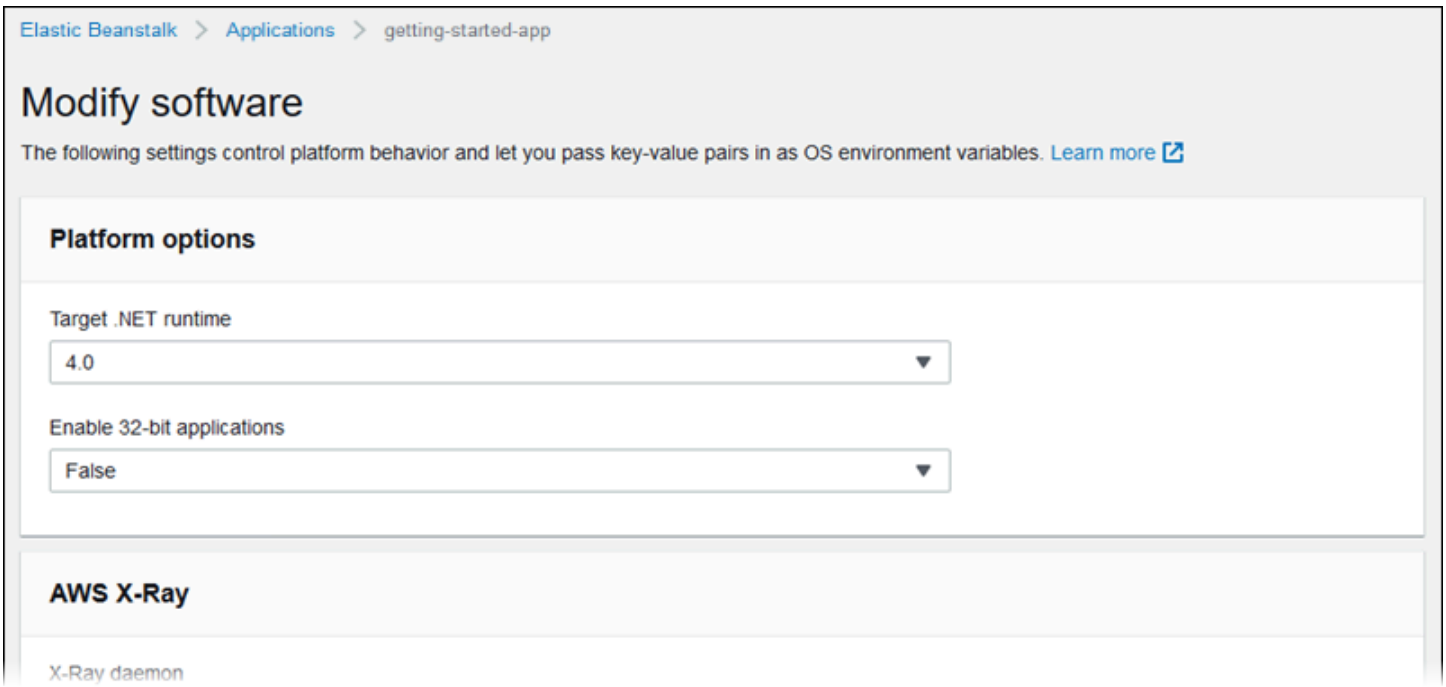
En plus (ou au lieu) de choisir un préréglage de configuration, vous pouvez ajuster les options de configuration de votre environnement. L'Assistant Configure (Configurer) affiche plusieurs catégories de configuration. Chaque catégorie de configuration affiche un résumé des valeurs pour un groupe de paramètres de configuration. Choisissez Edit (Modifier) pour modifier ce groupe de paramètres.

Catégories de configuration

- Paramètres du logiciel
- instances
- Capacity
- Equilibreur de charge
- Mises à jour et déploiements continus
- Sécurité
- Contrôle
- Mises à jour gérées
- Notifications
- Réseau
- Base de données
- Étiquettes
- Environnement de travail

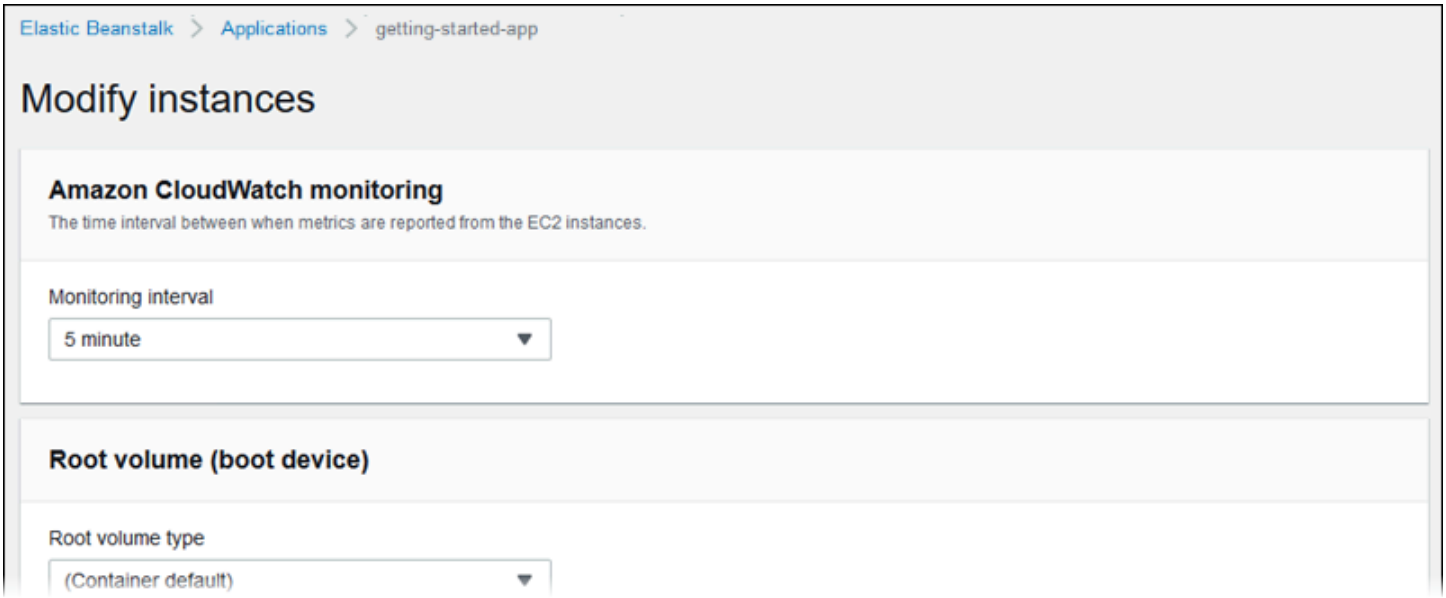
Paramètres du logiciel

Utilisez la page Modifier la configuration logicielle pour configurer le logiciel sur les instances Amazon Elastic Compute Cloud (Amazon EC2) qui exécutent votre application. Vous pouvez configurer les propriétés de l'environnement, le AWS X-Ray débogage, le stockage et le streaming des journaux d'instance, ainsi que les paramètres spécifiques à la plate-forme. Pour en savoir plus, consultez the section called “Variables d'environnement et paramètres logiciels”.



instances

Utilisez la page de configuration Modifier les instances pour configurer les EC2 instances Amazon qui exécutent votre application. Pour en savoir plus, consultez the section called “ EC2 Instances Amazon”.



Capacity

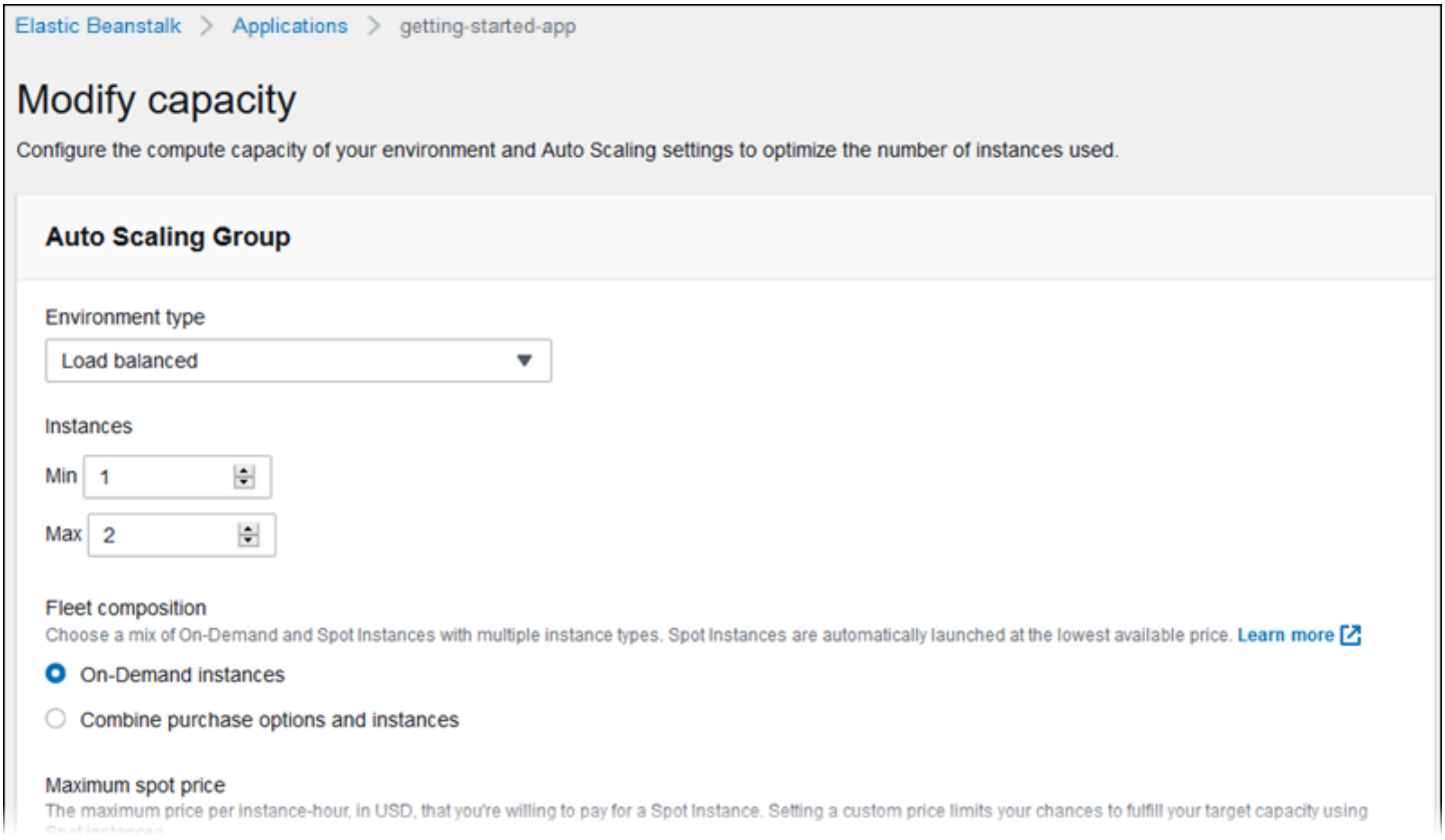
Utilisez la page de configuration Modify capacity (Modifier la capacité) pour configurer la capacité de calcul de votre environnement et les paramètres de groupe AUTO sCALING pour optimiser le

nombre et le type d'instances que vous utilisez. Vous pouvez également modifier la capacité de votre environnement en fonction des déclencheurs ou d'une planification.

Un environnement à charge équilibrée peut exécuter plusieurs instances pour une haute disponibilité et éviter les temps d'arrêt lors de mises à jour de configuration et de déploiements. Dans un environnement à charge équilibrée, le nom de domaine est mappé à l'équilibreur de charge. Dans un environnement d'instance unique, il est mappé à une adresse IP Elastic sur l'instance.

> **Warning**
>
> Un environnement d'instance unique n'est pas prêt pour la production. Si l'instance devient instable lors du déploiement, ou si Elastic Beanstalk résilie l'instance lors d'une mise à jour de configuration, votre application peut être indisponible pendant un certain temps. Utilisez des environnements d'instance unique pour le développement ou les tests, ou comme environnement intermédiaire. Utilisez des environnements à charge équilibrée pour la production.

Pour en savoir plus sur les paramètres de capacité de l'environnement, consultez the section called “Groupe Auto Scaling” et the section called “ EC2 Instances Amazon”.

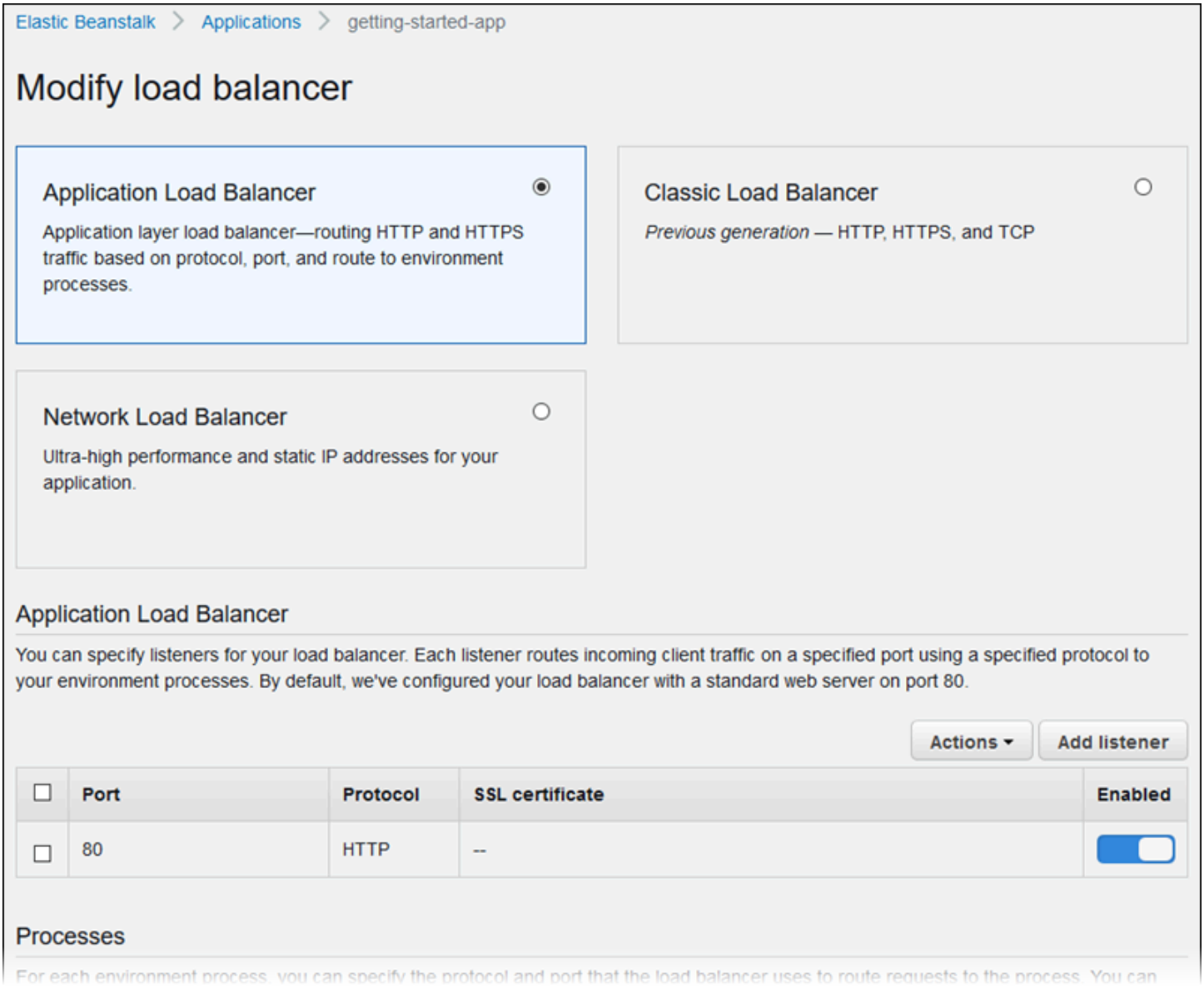


Equilibreur de charge

Utilisez la page de configuration Modifier l'équilibreur de charge pour sélectionner un type d'équilibreur de charge et configurer les paramètres correspondants. Dans un environnement à charge équilibrée, l'équilibreur de charge de votre environnement correspond au point d'entrée pour tout le trafic qui se dirige vers votre application. Elastic Beanstalk prend en charge plusieurs types d'équilibreur de charge. Par défaut, la console Elastic Beanstalk crée un équilibreur Application Load Balancer et le configure pour servir le trafic HTTP sur le port 80.

> **Note**
>
> Vous ne pouvez uniquement sélectionner le type d'équilibrage de charge de votre environnement lors de la création de l'environnement.

Pour plus d'informations sur les types et paramètres d'équilibreur de charge, consultez the section called “Équilibreur de charge” et the section called “HTTPS”.

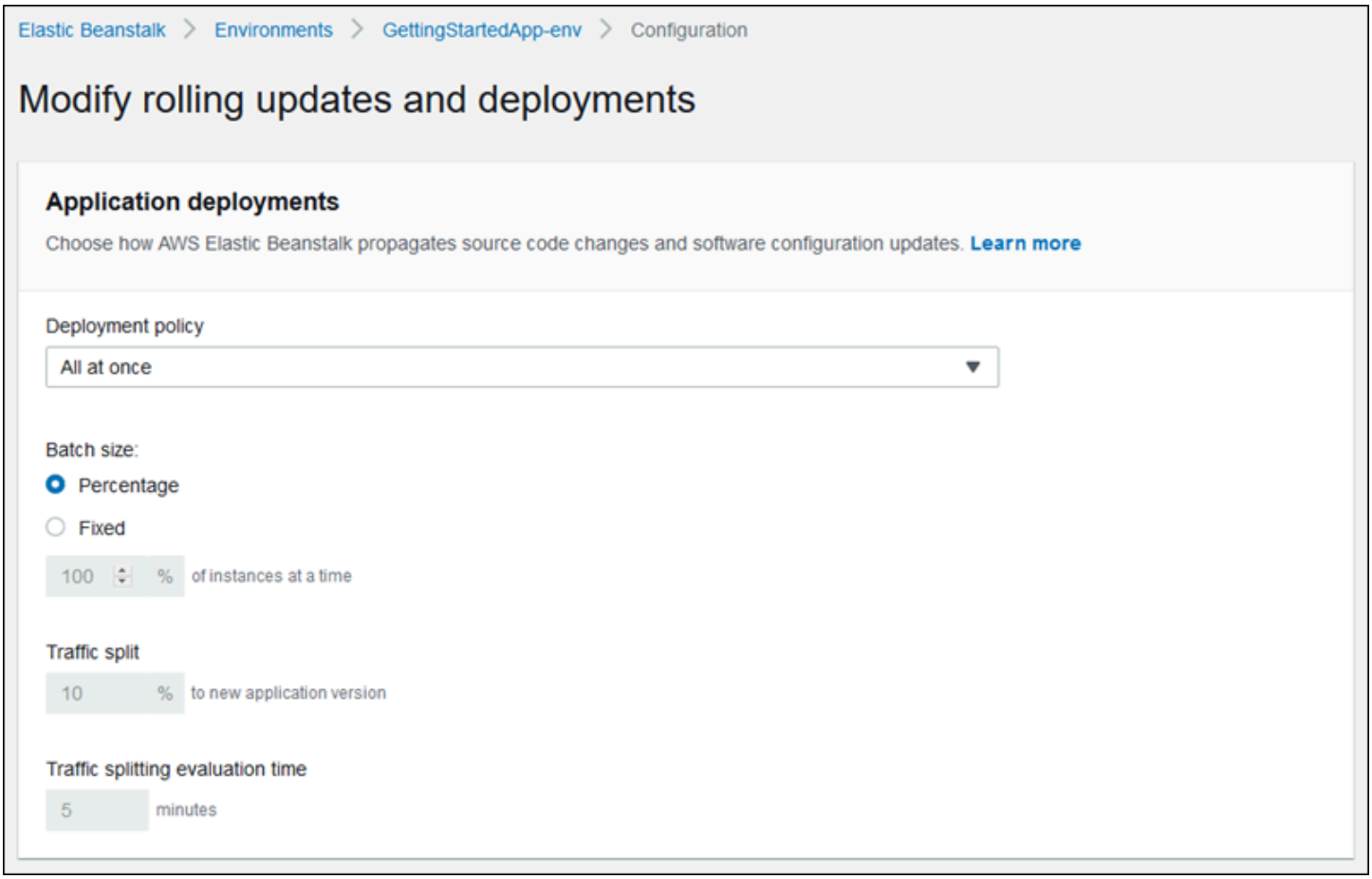


> **Note**
>
> L'option Classic Load Balancer (CLB) est désactivée dans l'assistant de création d'un environnement de la console. Si vous disposez d'un environnement existant configuré avec un Classic Load Balancer, vous pouvez en créer un nouveau en clonant l'environnement existant à l'aide de la console Elastic Beanstalk ou de l'interface de ligne de commande EB. Vous avez également la possibilité d'utiliser l'EB CLI ou AWS CLIde créer un nouvel environnement configuré avec un Classic Load Balancer. Ces outils de ligne de commande créeront un nouvel environnement avec un CLB, même s'il n'en existe pas déjà un dans votre compte.

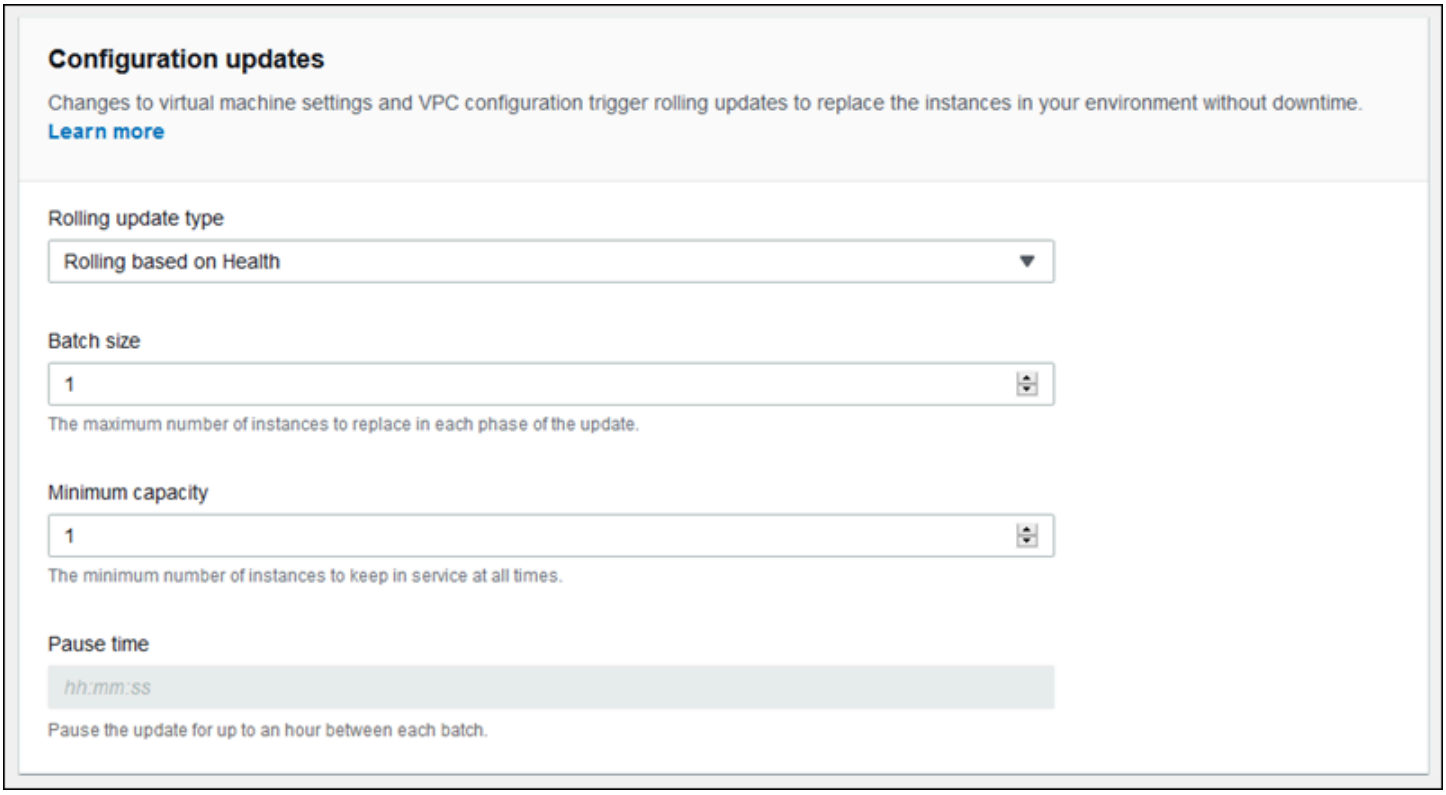
Mises à jour et déploiements continus

Utilisez la page Modify rolling updates and deployments (Modifier les mises à jour et les déploiements évolutifs) pour configurer le traitement des déploiements d'applications et des mises à jour de configuration par Elastic Beanstalk pour votre environnement.

Les déploiements d'applications se produisent lorsque vous téléchargez un bundle source d'application mis à jour et le déployez dans votre environnement. Pour plus d'informations sur la configuration des déploiements, consultez the section called “Options de déploiement”.



Les changements de configuration qui modifient la configuration du lancement ou les paramètres de VPC nécessitent de résilier toutes les instances de votre environnement et de les remplacer. Pour plus d'informations sur la définition du type de mise à jour et d'autres options, consultez the section called “Configuration changes”.



Sécurité

Utilisez la page Configure service access (Configuration de l’accès au service) pour configurer les paramètres de sécurité du service et de l'instance.

Pour obtenir une description des concepts de sécurité Elastic Beanstalk, veuillez consulter Permissions.

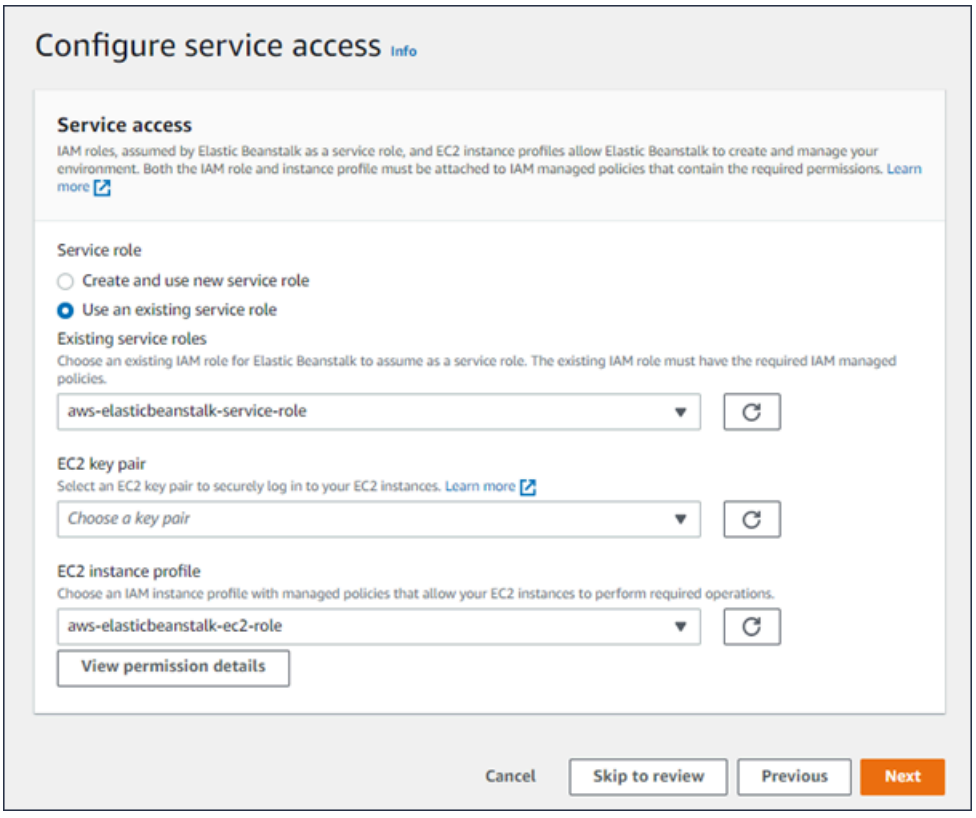
La première fois que vous créez un environnement dans la console Elastic Beanstalk, vous devez EC2 créer un profil d'instance avec un ensemble d'autorisations par défaut. Si la liste déroulante des profils d'EC2 instance n'affiche aucune valeur parmi laquelle choisir, développez la procédure ci-dessous. Il fournit des étapes pour créer un rôle que vous pouvez ensuite sélectionner pour le profil d'EC2 instance.

Créer un rôle IAM pour le profil d' EC2 instance

Pour créer ou sélectionner un profil d' EC2 instance

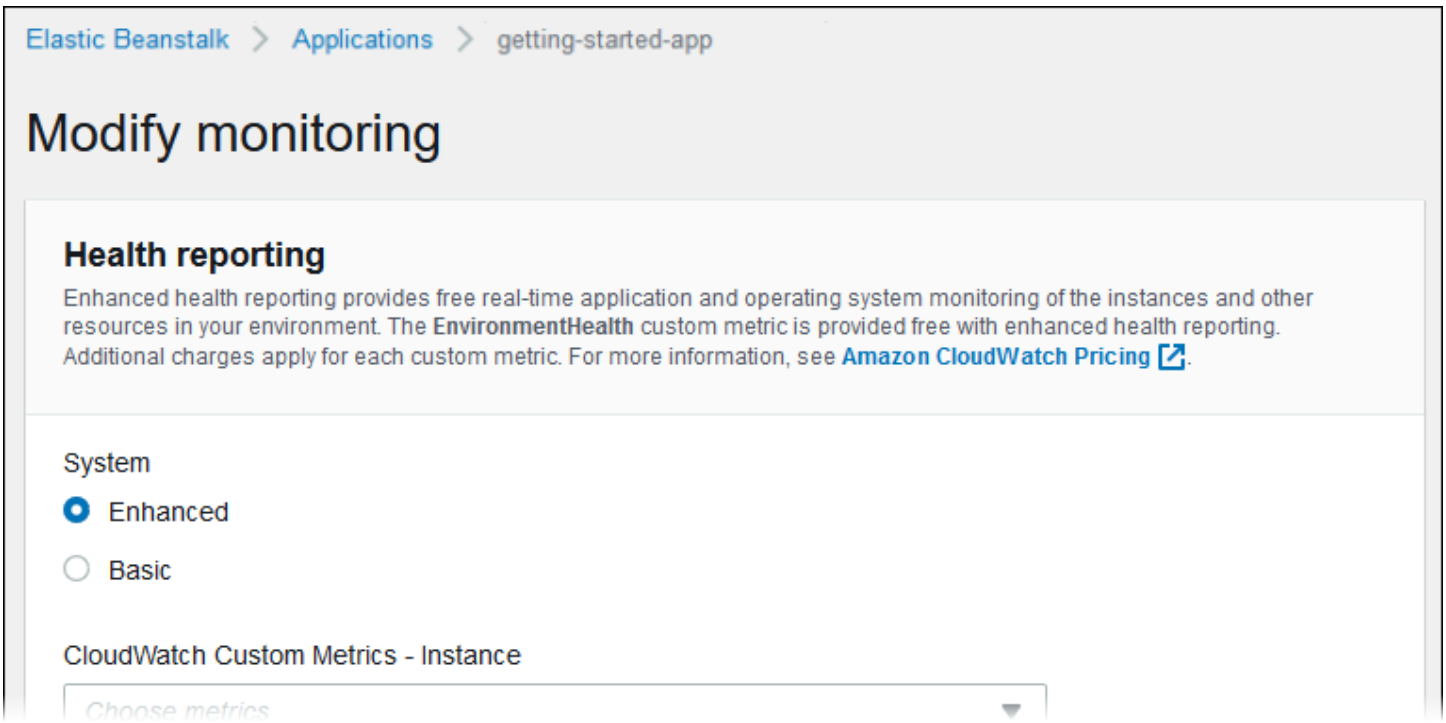
1. Si vous avez déjà créé un profil d'EC2 instance et que vous souhaitez en choisir un existant, sélectionnez la valeur dans le menu déroulant du profil d'EC2 instance et ignorez le reste de ces étapes pour créer un profil d' EC2instance.

2. Si aucune valeur n'est répertoriée pour le profil d'EC2 instance, ou si vous souhaitez en créer un nouveau, passez aux étapes suivantes.
3. Choisissez Créer un rôle.
4. Pour Type d’entité de confiance, choisissez Service AWS .
5. Dans le cas d'utilisation, choisissez Elastic Beanstalk — Compute.
6. Choisissez Suivant.
7. Vérifiez que les politiques d'autorisations incluent les éléments suivants, puis choisissez Next :
   - `AWSElasticBeanstalkWebTier`
   - `AWSElasticBeanstalkWorkerTier`
   - `AWSElasticBeanstalkMulticontainerDocker`
8. Choisissez Créer un rôle.
9. Retournez à l'onglet Configurer l'accès au service, actualisez la liste, puis sélectionnez le profil d' EC2 instance nouvellement créé.



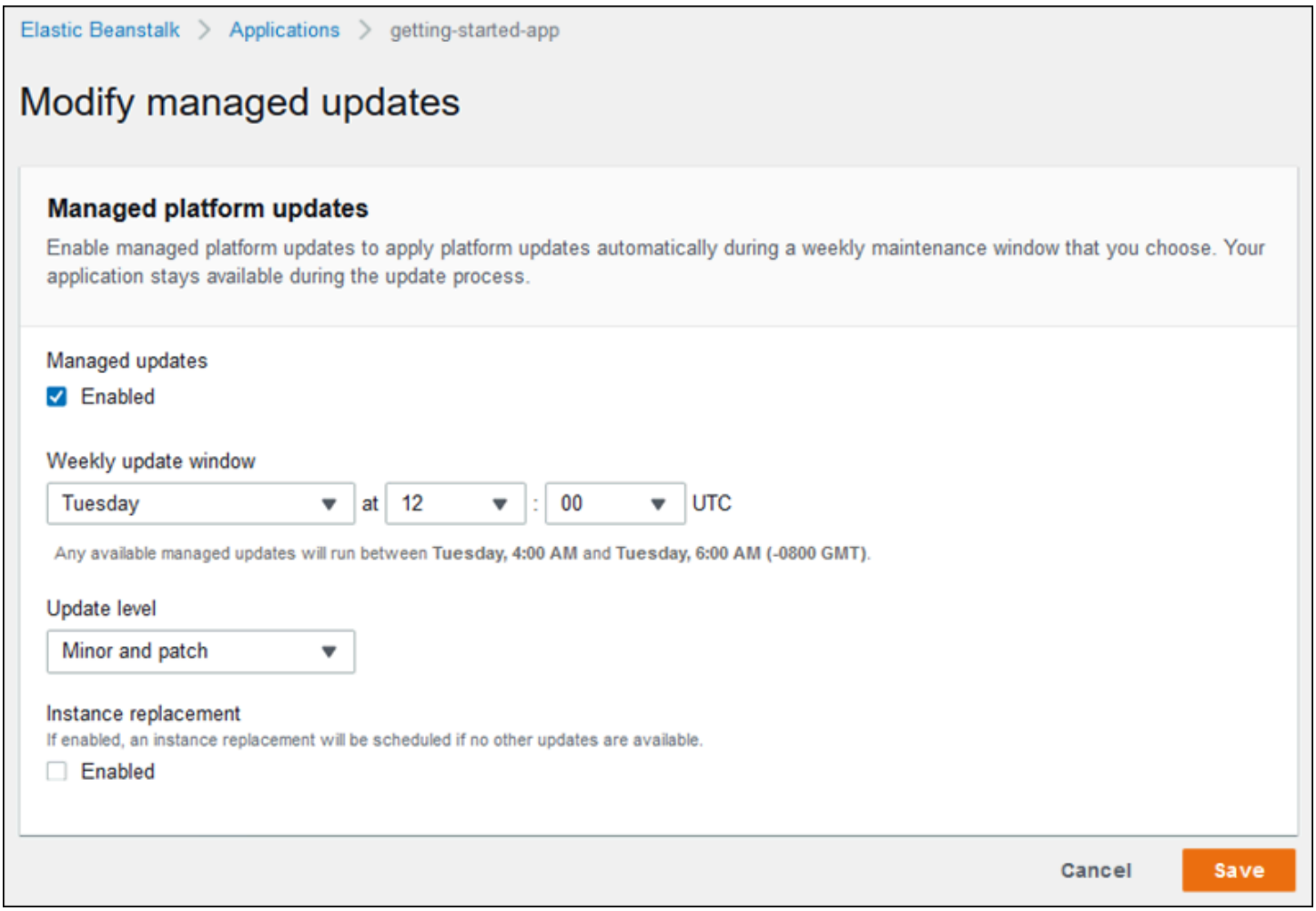
Contrôle

Utilisez la page Modify monitoring (Modifier la surveillance) pour configurer les rapports d'intégrité, les règles de surveillance et le streaming d'événements d'intégrité. Pour plus d'informations, consultez the section called “Activation des rapports d'intégrité améliorée”, the section called “Règles d'intégrité améliorée” et the section called “Diffusion des informations d'intégrité de l'environnement”.



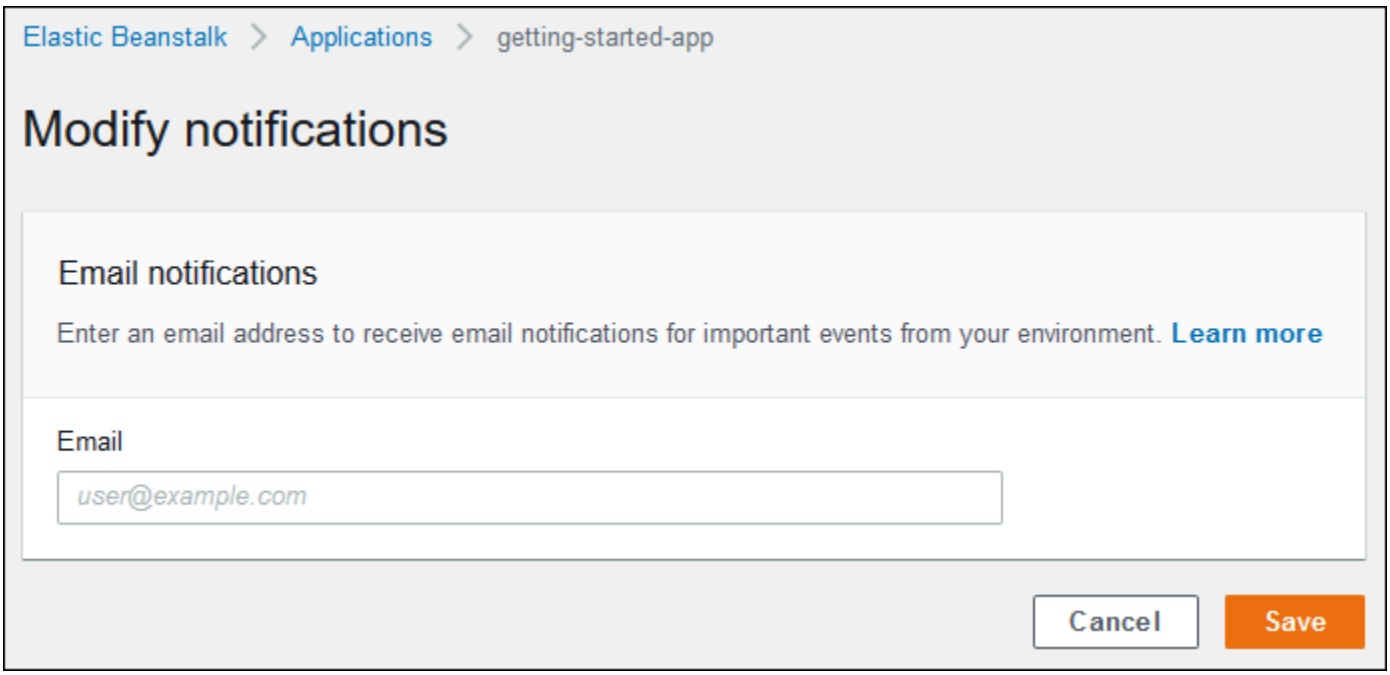
Mises à jour gérées

Utilisez la page Modify managed updates (Modifier les mises à jour gérées) pour configurer les mises à jour de plateformes gérées. Vous pouvez décider si vous voulez les activer, définir la planification et configurer d'autres propriétés. Pour plus d'informations, consultez the section called “Mises à jour gérées”.



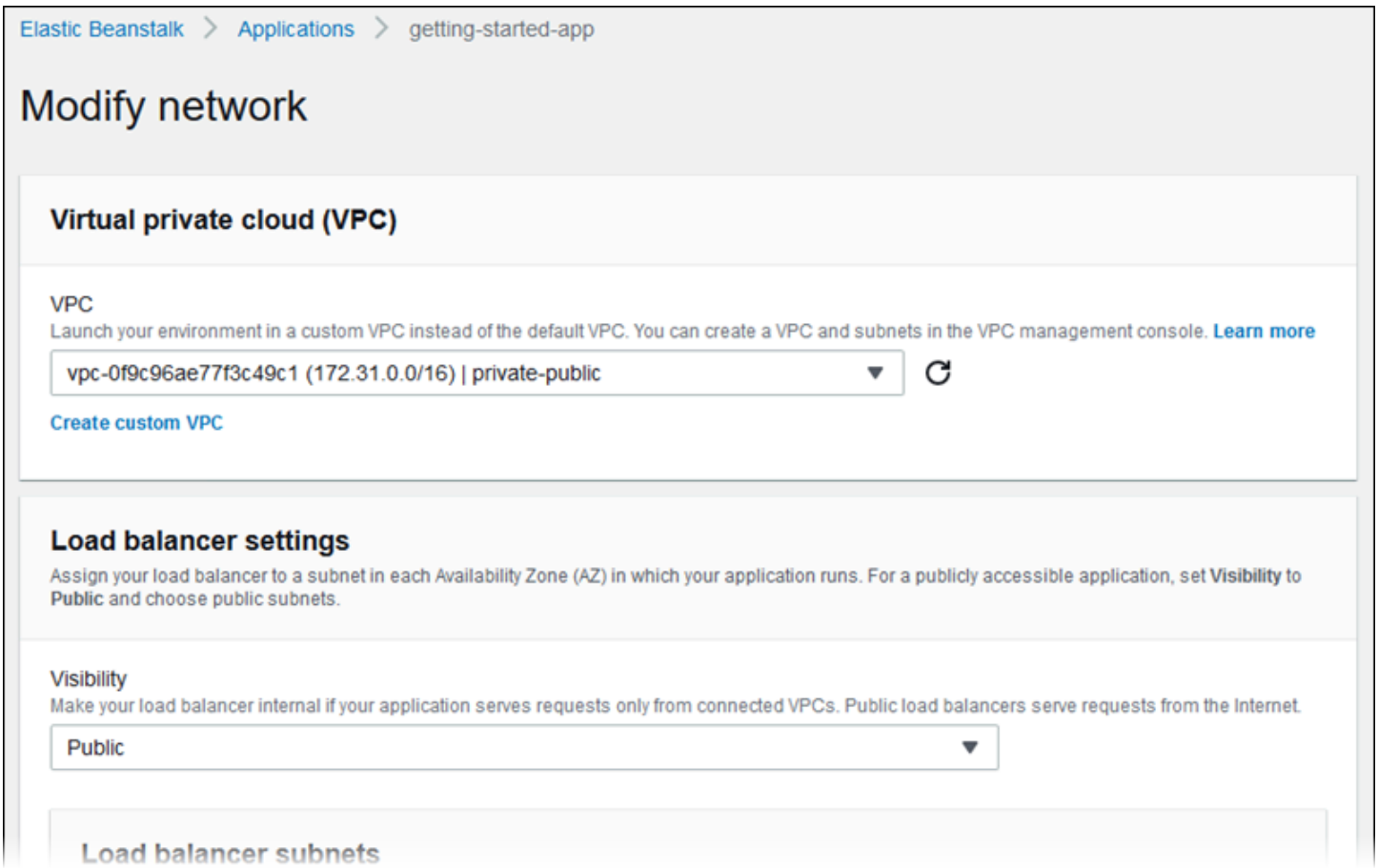
Notifications

Utilisez la page Modify notifications (Modifier les notifications) pour spécifier une adresse e-mail où recevoir des notifications par e-mail pour les événements importants de votre environnement.



Réseau

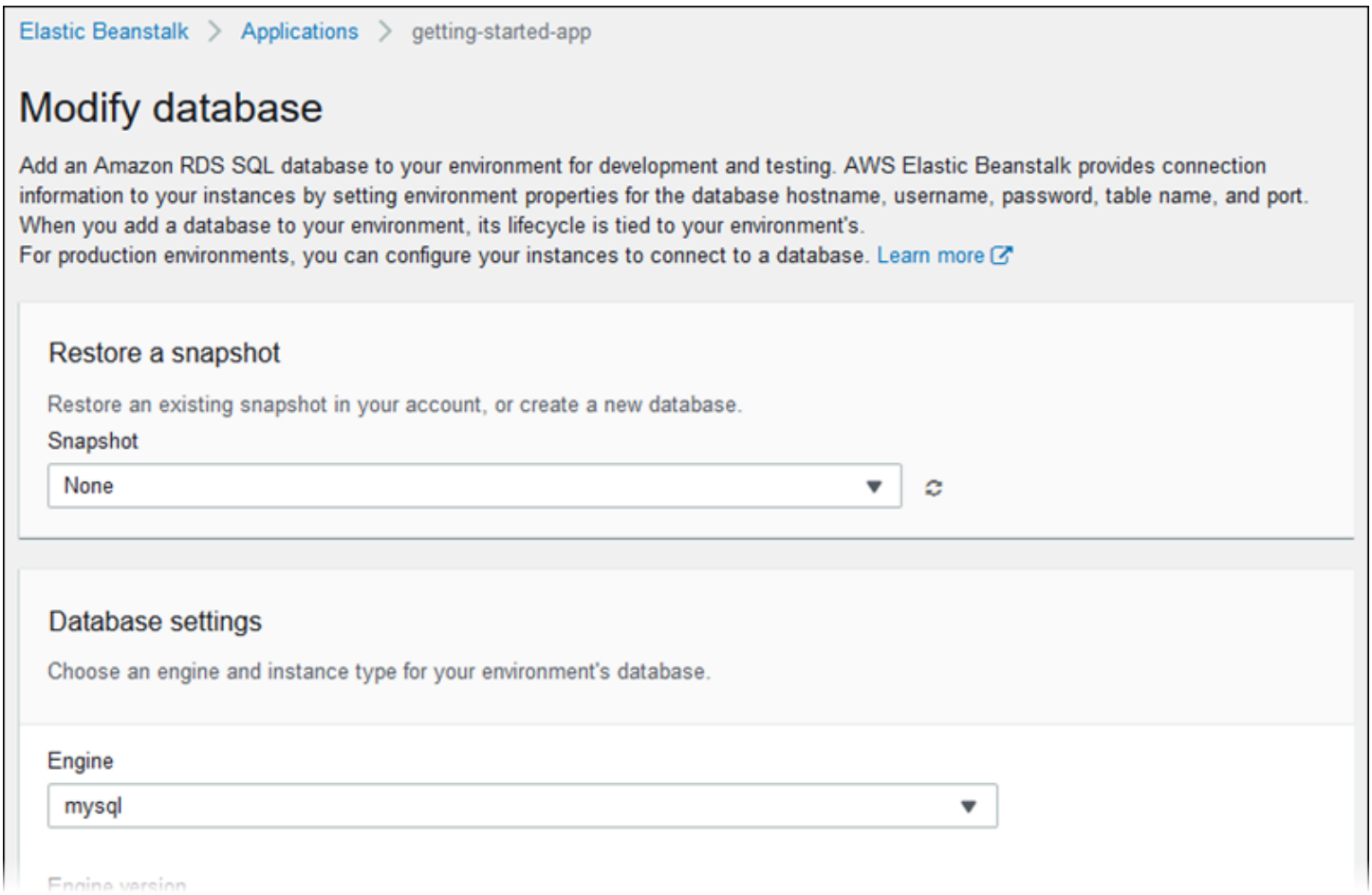
Si vous avez créé un VPC personnalisé, la page de configuration Modify network (Modifier le réseau) pour configurer votre environnement pour l'utiliser. Si vous ne choisissez pas de VPC, Elastic Beanstalk utilise le VPC et les sous-réseaux par défaut.



Base de données

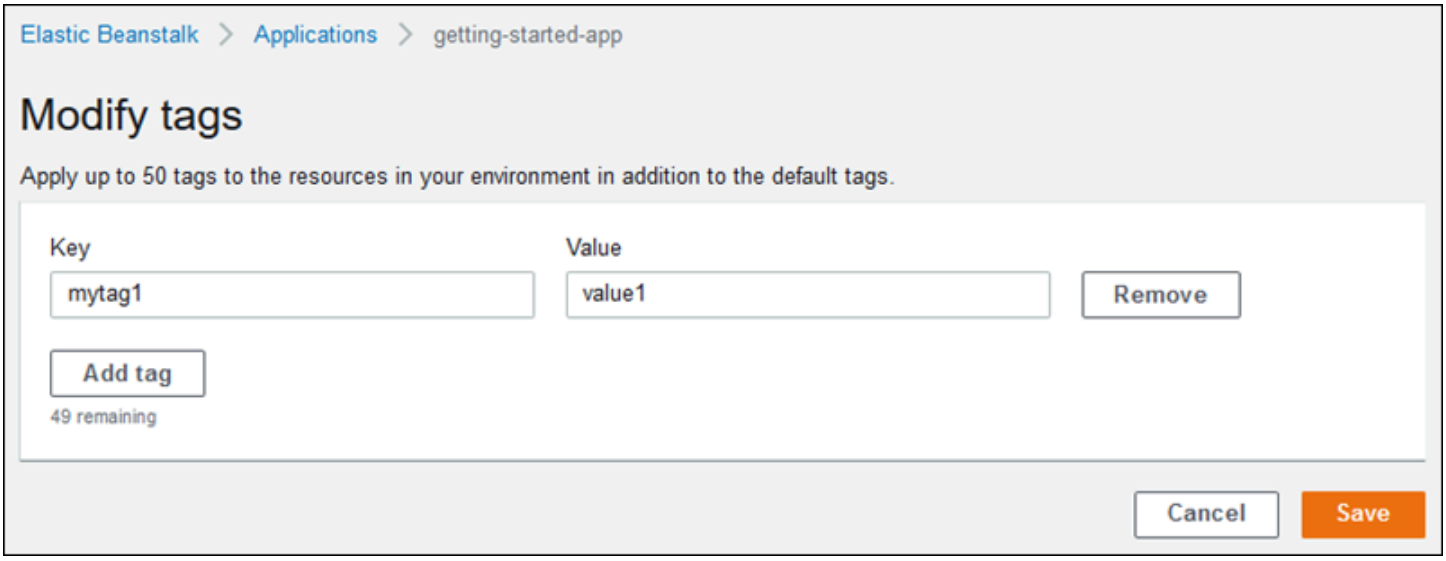
Utilisez la page Modify database configuration (Modifier la configuration de la base de données) pour ajouter une base de données Amazon Relational Database Service (Amazon RDS) à votre environnement à des fins de développement et de test. Elastic Beanstalk fournit les informations de connexion aux instances en définissant les propriétés de l'environnement pour le nom d'hôte de la base de données, le nom d'utilisateur, le mot de passe, le nom de la table et le port.

Pour en savoir plus, consultez the section called “Base de données”.



Étiquettes

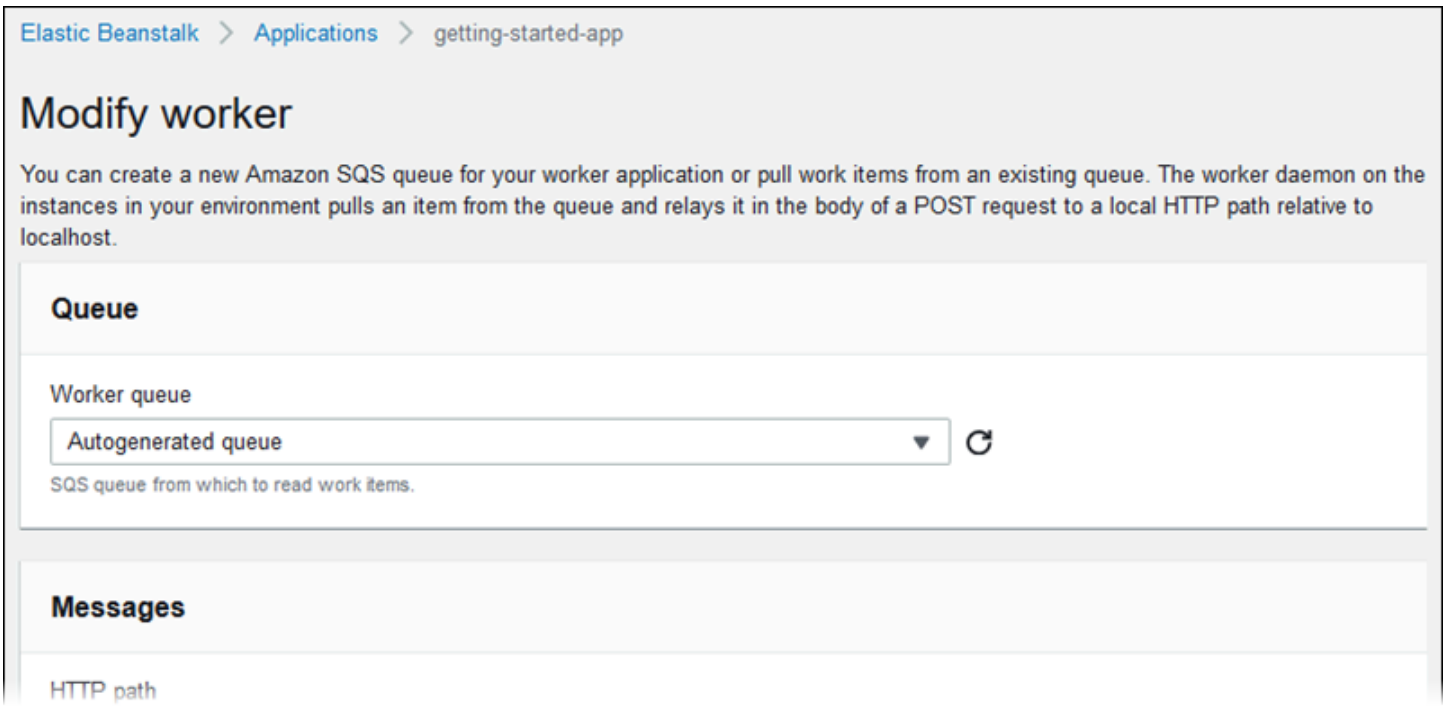
Utilisez la page de configuration Modify tags (Modifier les balises) pour ajouter des balises aux ressources de votre environnement. Pour plus d'informations sur l'utilisation de balises d'environnement, consultez Balisage des ressources dans vos environnements Elastic Beanstalk.



Environnement de travail

Si vous créez un environnement au niveau du travail, utilisez la page Modify worker (Modifier l'environnement de travail) pour configurer l'environnement de travail. Le démon de l'environnement de travail sur les instances de votre environnement extrait les éléments d'une file d'attente Amazon Simple Queue Service (Amazon SQS) et les relaie sous forme de messages de publication vers votre application de travail. Vous pouvez choisir la file d'attente Amazon SQS à partir de laquelle le démon de l'environnement de travail lit (générée automatiquement ou existante). Vous pouvez également configurer les messages que le démon de l'environnement de travail envoie à votre application.

Pour plus d'informations, consultez the section called “Environnements Worker”.



Clonage d'un environnement Elastic Beanstalk

Vous pouvez utiliser un environnement Elastic Beanstalk existant comme base d'un nouvel environnement en clonant l'environnement existant. Par exemple, vous pouvez créer un clone de façon à utiliser une version plus récente de la branche de plateforme utilisée par la plateforme de l'environnement d'origine. Elastic Beanstalk configure le clone avec les paramètres d'environnement utilisés par l'environnement d'origine. En clonant un environnement existant au lieu de créer un nouvel environnement, vous n'avez pas à configurer manuellement les paramètres des options, les variables d'environnement et les autres paramètres que vous avez définis avec le service Elastic Beanstalk. Elastic Beanstalk crée également une copie de toute ressource associée à AWS l'environnement d'origine.

Il est important de connaître les situations suivantes :

- Pendant le processus de clonage, Elastic Beanstalk ne copie pas les données d'Amazon RDS vers le clone.
- Elastic Beanstalk n'inclut aucune modification non gérée des ressources dans le clone. Les modifications que vous apportez aux ressources AWS à l'aide d'outils autres que la console de gestion Elastic Beanstalk, les outils de ligne de commande ou l'API sont considérées comme des modifications non gérées.

- Les groupes de sécurité pour l'entrée sont considérés comme des modifications non gérées. Les environnements Elastic Beanstalk clonés ne transfèrent pas les groupes de sécurité à des fins d'entrée, laissant l'environnement ouvert à tout le trafic Internet. Vous devrez rétablir les groupes de sécurité d'entrée pour l'environnement cloné.

Vous pouvez uniquement cloner un environnement vers une version de plateforme différente de la même branche de plateforme. Une autre branche de plateforme ne sera pas forcément compatible. Pour utiliser une autre branche de plateforme, vous devez créer manuellement un nouvel environnement, déployer votre code d'application et apporter les modifications nécessaires au code et aux options pour garantir que votre application fonctionne correctement sur la nouvelle branche de plateforme.

AWS console de gestion

> **Important**
>
> Les environnements Elastic Beanstalk clonés ne transfèrent pas les groupes de sécurité à des fins d'entrée, laissant l'environnement ouvert à tout le trafic Internet. Vous devrez rétablir les groupes de sécurité d'entrée pour l'environnement cloné.
> Vous pouvez voir les ressources qui ne peuvent pas être clonées en vérifiant l'état de dérive de la configuration de votre environnement. Pour plus d'informations, consultez la section Détecter la dérive sur une CloudFormation pile entière dans le Guide de AWS CloudFormation l'utilisateur.

Pour cloner un environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Sur la page de présentation de l'environnement, choisissez Actions.
4. Choisissez Clone environment (cloner l'environnement).
5. Sur la page Clone Environment (cloner l'environnement), examinez les informations de la section Original Environment (Environnement original), afin de vérifier si vous avez choisi l'environnement à partir duquel vous souhaitez créer un clone.

6. Dans la section New Environment (Nouvel environnement), vous pouvez modifier les valeurs des champs Environment name (Nom de l'environnement), Environment URL (URL de l'environnement), Description, Platform version (Version de la plateforme) et Service role (Fonction du service), qui sont définies automatiquement par Elastic Beanstalk en fonction de l'environnement d'origine.

   > **Note**
   >
   > Si la version de plateforme utilisée dans l'environnement d'origine n'est pas celle recommandée pour l'utiliser dans la branche de plateforme, vous êtes averti qu'une autre version de plateforme est recommandée. Choisissez Platform version (Version de la plateforme) pour afficher la version de la plateforme recommandée dans la liste : par exemple, 3.3.2 (Recommended).

7. Une fois que vous êtes prêt, choisissez Cloner.

Interface de ligne de commande (CLI) Elastic Beanstalk

> **Important**
>
> Les environnements Elastic Beanstalk clonés ne transfèrent pas les groupes de sécurité à des fins d'entrée, laissant l'environnement ouvert à tout le trafic Internet. Vous devrez rétablir les groupes de sécurité d'entrée pour l'environnement cloné.
> Vous pouvez voir les ressources qui ne peuvent pas être clonées en vérifiant l'état de dérive de la configuration de votre environnement. Pour plus d'informations, consultez la section Détecter la dérive sur une CloudFormation pile entière dans le Guide de AWS CloudFormation l'utilisateur.

Utilisez la commande eb clone pour cloner un environnement en cours d'exécution, comme suit :

```
~/workspace/my-app$ eb clone my-env1
Enter name for Environment Clone
(default is my-env1-clone): my-env2
Enter DNS CNAME prefix
(default is my-env1-clone): my-env2
```

Vous pouvez spécifier le nom de l'environnement source dans la commande de clonage ou l'omettre afin de cloner l'environnement par défaut pour le dossier de projet actuel. L'interface de ligne de commande (CLI) EB vous invite à saisir un nom et un préfixe DNS pour le nouvel environnement.

Par défaut, eb clone crée le nouvel environnement avec la dernière version disponible de la plateforme de l'environnement source. Pour forcer l'interface de ligne de commande (CLI) EB à utiliser la même version, même si une version plus récente est disponible, utilisez l'option `--exact`.

```
~/workspace/my-app$ eb clone --exact
```

Pour plus d'informations sur cette commande, consultez eb clone.

Arrêt d'un environnement Elastic Beanstalk

Vous pouvez mettre fin à un AWS Elastic Beanstalk environnement d'exécution à l'aide de la console Elastic Beanstalk. Ce faisant, vous évitez de payer des frais pour les ressources non utilisées AWS .

> **Note**
>
> Vous pouvez toujours lancer un nouvel environnement en utilisant la même version ultérieurement.

Si vous possédez des données d'un environnement que vous souhaitez conserver, définissez la stratégie de suppression de base de données sur `Retain` avant de résilier l'environnement. Cela permet de maintenir la base de données opérationnelle en dehors d'Elastic Beanstalk. Tous les environnements Elastic Beanstalk doivent ensuite s'y connecter en tant que base de données externe. Si vous souhaitez sauvegarder les données sans maintenir la base de données opérationnelle, définissez la stratégie de suppression pour qu'elle prenne un instantané de la base de données avant de résilier l'environnement. Pour de plus amples informations, consultez Cycle de vie de base de données dans le chapitre Configuration des environnements de ce guide.

Elastic Beanstalk peut ne pas réussir à arrêter votre environnement. Une raison courante est que le groupe de sécurité d'un autre environnement comporte une dépendance sur le groupe de sécurité de l'environnement que vous souhaitez résilier. Pour savoir comment éviter ce problème, consultez la EC2 groupes de sécurité page EC2Instances de ce guide.

> **Important**
>
> Si vous résiliez un environnement, vous devez également supprimer tous les mappages CNAME que vous avez créés, car d'autres clients peuvent réutiliser un nom d'hôte disponible. Veillez à supprimer les enregistrements DNS qui pointent vers votre environnement résilié, afin d'éviter toute entrée DNS en suspens. Une entrée DNS bloquée peut exposer le trafic Internet destiné à votre domaine à des failles de sécurité. Elle peut également présenter d'autres risques.
> Pour de plus amples informations, veuillez consulter Protection contre les enregistrements de délégation suspendus dans Route 53 dans le manuel du développeur Amazon Route 53. Vous pouvez également en savoir plus sur le blocage des entrées DNS dans Enhanced Domain Protections for Amazon CloudFront Requests sur le blog sur la AWS sécurité.

Console Elastic Beanstalk

Pour résilier un environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Terminate Environment (Résilier l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

   > **Note**
   >
   > Lorsque vous résiliez votre environnement, le CNAME associé à l'environnement résilié devient disponible pour que tout le monde puisse l'utiliser.

   Elastic Beanstalk met quelques minutes à arrêter les ressources en cours d'exécution AWS dans l'environnement.

AWS CLI

Pour résilier un environnement

- Exécutez la commande suivante.

```
$ aws elasticbeanstalk terminate-environment --environment-name my-env
```

API

Pour résilier un environnement

- Appelez `TerminateEnvironment` avec le paramètre suivant :

  `EnvironmentName` = `SampleAppEnv`

```
https://elasticbeanstalk.us-west-2.amazon.com/?EnvironmentName=SampleAppEnv
&Operation=TerminateEnvironment
&AuthParams
```

Création d'environnements Elastic Beanstalk avec la CLI AWS

Pour plus de détails sur les AWS CLI commandes pour Elastic Beanstalk, consultez le Command Reference.AWS CLI

1. Vérifiez si le CNAME pour l'environnement est disponible.

```
$ aws elasticbeanstalk check-dns-availability --cname-prefix my-cname
{
    "Available": true,
    "FullyQualifiedCNAME": "my-cname.elasticbeanstalk.com"
}
```

2. Assurez-vous que la version de votre application existe.

```
$ aws elasticbeanstalk describe-application-versions --application-name my-app --
version-label v1
```

Si vous ne disposez pas d'une version de l'application pour votre source, créez-la. Par exemple, la commande suivante crée une version d'application à partir d'un bundle de fichiers source dans Amazon Simple Storage Service (Amazon S3).

```
$ aws elasticbeanstalk create-application-version --application-name my-app --
version-label v1 --source-bundle S3Bucket=amzn-s3-demo-bucket,S3Key=my-source-
bundle.zip
```

3. Créez un modèle de configuration pour l'application.

```
$ aws elasticbeanstalk create-configuration-template --application-name my-app --
template-name v1 --solution-stack-name "64bit Amazon Linux 2015.03 v2.0.0 running
 Ruby 2.2 (Passenger Standalone)"
```

4. Créez un environnement.

```
$ aws elasticbeanstalk create-environment --cname-prefix my-cname --application-
name my-app --template-name v1 --version-label v1 --environment-name v1clone --
option-settings file://options.txt
```

Les paramètres d'option sont définis dans le fichier options.txt :

```
[
    {
        "Namespace": "aws:autoscaling:launchconfiguration",
        "OptionName": "IamInstanceProfile",
        "Value": "aws-elasticbeanstalk-ec2-role"
    }
]
```

Le paramètre d'option ci-dessus définit le profil d'instance IAM. Vous pouvez spécifier l'ARN ou le nom du profil.

5. Déterminez si le nouvel environnement est vert et prêt.

```
$ aws elasticbeanstalk describe-environments --environment-names my-env
```

Si le nouvel environnement n'est pas vert et prêt, vous devez décider si vous souhaitez recommencer l'opération ou laisser l'environnement dans son état actuel pour enquête. Veillez

à mettre fin à l'environnement lorsque vous avez terminé, puis nettoyez toutes les ressources inutilisées.

> **Note**
>
> Vous pouvez régler la période d'expiration si l'environnement ne se lance pas dans un délai raisonnable.

Création d'environnements Elastic Beanstalk avec l'API

1. Appelez `CheckDNSAvailability` avec le paramètre suivant :

   - `CNAMEPrefix` = `SampleApp`

   Example

   ```
   https://elasticbeanstalk.us-east-2.amazonaws.com/?CNAMEPrefix=sampleapplication
   &Operation=CheckDNSAvailability
   &AuthParams
   ```

2. Appelez `DescribeApplicationVersions` avec les paramètres suivants :

   - `ApplicationName` = `SampleApp`
   - `VersionLabel` = `Version2`

   Example

   ```
   https://elasticbeanstalk.us-east-2.amazonaws.com/?ApplicationName=SampleApp
   &VersionLabel=Version2
   &Operation=DescribeApplicationVersions
   &AuthParams
   ```

3. Appelez `CreateConfigurationTemplate` avec les paramètres suivants :

   - `ApplicationName` = `SampleApp`
   - `TemplateName` = `MyConfigTemplate`

- SolutionStackName = 64bit%20Amazon%20Linux%202015.03%20v2.0.0%20running%20Ruby%202.2%20(Passenger%20Standalone)

Example

```
https://elasticbeanstalk.us-east-2.amazonaws.com/?ApplicationName=SampleApp
&TemplateName=MyConfigTemplate
&Operation=CreateConfigurationTemplate
&SolutionStackName=64bit%20Amazon%20Linux%202015.03%20v2.0.0%20running%20Ruby
%202.2%20(Passenger%20Standalone)
&AuthParams
```

4. Appelez CreateEnvironment avec l'un des ensembles de paramètres suivants.

   a. Utilisez les paramètres suivants pour un niveau d'environnement « serveur web » :

   - EnvironmentName = SampleAppEnv2
   - VersionLabel = Version2
   - Description = description
   - TemplateName = MyConfigTemplate
   - ApplicationName = SampleApp
   - CNAMEPrefix = sampleapplication
   - OptionSettings.member.1.Namespace = aws:autoscaling:launchconfiguration
   - OptionSettings.member.1.OptionName = IamInstanceProfile
   - OptionSettings.member.1.Value = aws-elasticbeanstalk-ec2-role

   Example

```
https://elasticbeanstalk.us-east-2.amazonaws.com/?ApplicationName=SampleApp
&VersionLabel=Version2
&EnvironmentName=SampleAppEnv2
&TemplateName=MyConfigTemplate
&CNAMEPrefix=sampleapplication
&Description=description
&Operation=CreateEnvironment
&OptionSettings.member.1.Namespace=aws%3Aautoscaling%3Alaunchconfiguration
```

```
&OptionSettings.member.1.OptionName=IamInstanceProfile
&OptionSettings.member.1.Value=aws-elasticbeanstalk-ec2-role
&AuthParams
```

b. Utilisez les paramètres suivants pour un niveau d'environnement de travail :

- EnvironmentName = SampleAppEnv2
- VersionLabel = Version2
- Description = description
- TemplateName = MyConfigTemplate
- ApplicationName = SampleApp
- Tier = Worker
- OptionSettings.member.1.Namespace = aws:autoscaling:launchconfiguration
- OptionSettings.member.1.OptionName = IamInstanceProfile
- OptionSettings.member.1.Value = aws-elasticbeanstalk-ec2-role
- OptionSettings.member.2.Namespace = aws:elasticbeanstalk:sqsd
- OptionSettings.member.2.OptionName = WorkerQueueURL
- OptionSettings.member.2.Value = sqsd.elasticbeanstalk.us-east-2.amazonaws.com
- OptionSettings.member.3.Namespace = aws:elasticbeanstalk:sqsd
- OptionSettings.member.3.OptionName = HttpPath
- OptionSettings.member.3.Value = /
- OptionSettings.member.4.Namespace = aws:elasticbeanstalk:sqsd
- OptionSettings.member.4.OptionName = MimeType
- OptionSettings.member.4.Value = application/json
- OptionSettings.member.5.Namespace = aws:elasticbeanstalk:sqsd
- OptionSettings.member.5.OptionName = HttpConnections
- OptionSettings.member.5.Value = 75
- OptionSettings.member.6.Namespace = aws:elasticbeanstalk:sqsd
- OptionSettings.member.6.OptionName = ConnectTimeout
- OptionSettings.member.6.Value = 10
- OptionSettings.member.7.Namespace = aws:elasticbeanstalk:sqsd

- OptionSettings.member.7.OptionName = InactivityTimeout
- OptionSettings.member.7.Value = 10
- OptionSettings.member.8.Namespace = aws:elasticbeanstalk:sqsd
- OptionSettings.member.8.OptionName = VisibilityTimeout
- OptionSettings.member.8.Value = 60
- OptionSettings.member.9.Namespace = aws:elasticbeanstalk:sqsd
- OptionSettings.member.9.OptionName = RetentionPeriod
- OptionSettings.member.9.Value = 345600

Example

```
https://elasticbeanstalk.us-east-2.amazonaws.com/?ApplicationName=SampleApp
&VersionLabel=Version2
&EnvironmentName=SampleAppEnv2
&TemplateName=MyConfigTemplate
&Description=description
&Tier=Worker
&Operation=CreateEnvironment
&OptionSettings.member.1.Namespace=aws%3Aautoscaling%3Alaunchconfiguration
&OptionSettings.member.1.OptionName=IamInstanceProfile
&OptionSettings.member.1.Value=aws-elasticbeanstalk-ec2-role
&OptionSettings.member.2.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.2.OptionName=WorkerQueueURL
&OptionSettings.member.2.Value=sqsd.elasticbeanstalk.us-east-2.amazonaws.com
&OptionSettings.member.3.Namespace=aws%3elasticbeanstalk%3sqsd
&OptionSettings.member.3.OptionName=HttpPath
&OptionSettings.member.3.Value=%2F
&OptionSettings.member.4.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.4.OptionName=MimeType
&OptionSettings.member.4.Value=application%2Fjson
&OptionSettings.member.5.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.5.OptionName=HttpConnections
&OptionSettings.member.5.Value=75
&OptionSettings.member.6.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.6.OptionName=ConnectTimeout
&OptionSettings.member.6.Value=10
&OptionSettings.member.7.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.7.OptionName=InactivityTimeout
&OptionSettings.member.7.Value=10
```

```
&OptionSettings.member.8.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.8.OptionName=VisibilityTimeout
&OptionSettings.member.8.Value=60
&OptionSettings.member.9.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.9.OptionName=RetentionPeriod
&OptionSettings.member.9.Value=345600
&AuthParams
```

Construction d'une URL Launch Now

Vous pouvez créer une URL personnalisée afin que tout le monde puisse rapidement déployer et exécuter une application web prédéfinie dans AWS Elastic Beanstalk. On l'appelle une URL Launch Now. Vous pourrez avoir besoin d'une URL Launch Now, par exemple, pour illustrer une application web construite pour s'exécuter sur Elastic Beanstalk. Avec une URL Launch Now, vous pouvez utiliser des paramètres pour ajouter les informations requises à l'assistant Create Application à l'avance. Une fois ces informations ajoutées à l'assistant, tout le monde peut utiliser le lien de l'URL pour lancer un environnement Elastic Beanstalk avec votre source d'application web en quelques étapes. Cela signifie que les utilisateurs n'ont pas besoin de charger manuellement ou d'indiquer l'emplacement de l'offre groupée de source d'application. Ils ne doivent pas non plus fournir d'informations supplémentaires à l'assistant.

Une URL Launch Now fournit à Elastic Beanstalk les informations minimales requises pour créer une application : le nom de l'application, la pile de solutions, le type d'instance et le type d'environnement. Elastic Beanstalk utilise les valeurs par défaut pour d'autres détails de configuration qui ne sont pas explicitement spécifiés dans votre URL Launch Now personnalisée.

Une URL Launch Now utilise la syntaxe d'URL standard. Pour plus d'informations, consultez RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax.

Paramètres d'URL

L'URL doit contenir les paramètres suivants, qui sont sensibles à la casse :

- région — Spécifiez une AWS région. Pour obtenir la liste des régions prises en charge par Elastic Beanstalk, consultez Points de terminaison et quotas AWS Elastic Beanstalk dans la Références générales AWS.
- applicationName : spécifiez le nom de votre application. Elastic Beanstalk affiche le nom de l'application dans la console Elastic Beanstalk pour le distinguer des autres applications. Par

défaut, le nom de l'application constitue également la base du nom de l'environnement et de l'URL de l'environnement.

- platform : spécifiez la version de plateforme à utiliser pour l'environnement. Utilisez l'une des méthodes suivantes, puis encodez votre choix sous forme d'URL :
  - Spécifiez un ARN de plateforme sans version. Elastic Beanstalk sélectionne la dernière version majeure de la plateforme correspondante. Par exemple, pour sélectionner la dernière version de la plateforme Python 3.6, spécifiez `Python 3.6 running on 64bit Amazon Linux`.
  - Spécifiez le nom de la plateforme. Elastic Beanstalk sélectionne la dernière version du dernier runtime linguistique de la plateforme (par exemple, `Python`).

  Pour obtenir la description de toutes les plateformes disponibles et de leurs versions, veuillez consulter Plateformes prises en charge par Elastic Beanstalk.

  Vous pouvez utiliser le AWS Command Line Interface(AWS CLI) pour obtenir une liste de toutes les versions de plateforme disponibles avec leurs versions respectives ARNs. La commande `list-platform-versions` affiche des informations détaillées sur toutes les versions de plateforme disponibles. Utilisez l'argument `--filters` pour définir la liste. Par exemple, vous pouvez restreindre la liste de manière à n'afficher que les versions de plateforme d'un langage spécifique.

  L'exemple suivant interroge toutes les versions de plateforme Python et affiche le résultat via une série de commandes. Le résultat est une liste des versions de la plateforme ARNs (sans la */version* queue), dans un format lisible par l'homme, sans encodage d'URL.

```
$ aws elasticbeanstalk list-platform-versions --filters
 'Type="PlatformName",Operator="contains",Values="Python"' | grep PlatformArn | awk -
F '"' '{print $4}' | awk -F '/' '{print $2}'
Preconfigured Docker - Python 3.4 running on 64bit Debian
Preconfigured Docker - Python 3.4 running on 64bit Debian
Python 2.6 running on 32bit Amazon Linux
Python 2.6 running on 32bit Amazon Linux 2014.03
...
Python 3.6 running on 64bit Amazon Linux
```

  L'exemple suivant ajoute une commande Perl pour le dernier exemple, afin de coder la sortie sous forme d'URL.

```
$ aws elasticbeanstalk list-platform-versions --filters
 'Type="PlatformName",Operator="contains",Values="Python"' | grep PlatformArn | awk
```

```
 -F '"' '{print $4}' | awk -F '/' '{print $2}' | perl -MURI::Escape -ne 'chomp;print
 uri_escape($_),"\n"'
Preconfigured%20Docker%20-%20Python%203.4%20running%20on%2064bit%20Debian
Preconfigured%20Docker%20-%20Python%203.4%20running%20on%2064bit%20Debian
Python%202.6%20running%20on%2032bit%20Amazon%20Linux
Python%202.6%20running%20on%2032bit%20Amazon%20Linux%202014.03
...
Python%203.6%20running%20on%2064bit%20Amazon%20Linux
```

Une URL Launch Now peut en option contenir les paramètres suivants. Si vous n'incluez pas les paramètres facultatifs dans votre URL Launch Now, Elastic Beanstalk utilise les valeurs par défaut pour créer et exécuter votre application. Lorsque vous n'incluez pas le sourceBundleUrlparamètre, Elastic Beanstalk utilise l'exemple d'application par défaut pour la plate-forme spécifiée.

- sourceBundleUrl— Spécifiez l'emplacement du bundle source de votre application Web au format URL. Par exemple, si vous avez chargé votre bundle source dans un compartiment Amazon S3, vous pouvez spécifier la valeur du sourceBundleUrlparamètre sous la forme`https://amzn-s3-demo-bucket.s3.amazonaws.com/myobject`.

  > **Note**
  >
  > Vous pouvez spécifier la valeur du sourceBundleUrlparamètre sous forme d'URL HTTP, mais le navigateur Web de l'utilisateur convertira les caractères selon les besoins en appliquant un codage URL HTML.

- environmentType : indiquez si l'environnement est à charge équilibrée et évolutive ou s'il s'agit d'une seule instance. Pour plus d'informations, consultez Types d'environnement. Vous pouvez spécifier `LoadBalancing` ou `SingleInstance` comme la valeur de paramètre.
- tierName : spécifiez si l'environnement prend en charge une application web traitant les demandes web ou une application web exécutant les tâches en arrière-plan. Pour plus d'informations, consultez Environnements de travail Elastic Beanstalk. Vous pouvez spécifier `WebServer` ou `Worker`,
- instanceType : spécifiez un serveur dont les caractéristiques (y compris la taille de la mémoire et la puissance de l'UC) sont les mieux adaptées à votre application. Pour plus d'informations sur les familles et les types d' EC2 instances Amazon, consultez la section Types d'instances dans le guide de EC2 l'utilisateur Amazon. Pour plus d'informations sur les types d'instances disponibles

dans les différentes régions, consultez la section Types d'instances disponibles dans le guide de EC2 l'utilisateur Amazon.

- withVpc : spécifiez si vous créez l'environnement dans un Amazon VPC. Vous pouvez spécifier `true` ou `false`. Pour de plus amples informations sur l'utilisation d'Elastic Beanstalk avec Amazon VPC, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon VPC.
- withRds : spécifiez si vous créez une instance de base de données Amazon RDS avec cet environnement. Pour plus d'informations, consultez Utilisation d'Elastic Beanstalk avec Amazon RDS. Vous pouvez spécifier `true` ou `false`.
- rds DBEngine — Spécifiez le moteur de base de données que vous souhaitez utiliser pour vos EC2 instances Amazon dans cet environnement. Vous pouvez spécifier `mysql`, `oracle-sel`, `sqlserver-ex`, `sqlserver-web` ou `sqlserver-se`. La valeur par défaut est `mysql`.
- DBAllocatedStockage rds — Spécifiez la taille de stockage de base de données allouée en gigaoctets (Go). Vous pouvez spécifier les valeurs suivantes :
  - MySQL – `5` à `1024`. La valeur par défaut est `5`.
  - Oracle – `10` à `1024`. La valeur par défaut est `10`.
  - Microsoft SQL Server Express Edition – `30`.
  - Microsoft SQL Server Web Edition – `30`.
  - Microsoft SQL Server Standard Edition – `200`.
- DBInstanceClasse rds — Spécifiez le type d'instance de base de données. La valeur par défaut est `db.t2.micro` (`db.m1.large` pour un environnement qui ne s'exécute pas dans un Amazon VPC). Pour obtenir la liste des classes d'instance de base de données prises en charge par Amazon RDS, reportez-vous à Classe d'instance de base de données dans le Guide de l'utilisateur Amazon Relational Database Service.
- RDSmulti AZDatabase — Spécifiez si Elastic Beanstalk doit créer l'instance de base de données dans plusieurs zones de disponibilité. Vous pouvez spécifier `true` ou `false`. Pour de plus amples informations sur les déploiements multi-AZ avec Amazon RDS, consultez Régions et zones de disponibilité dans le Guide de l'utilisateur Amazon Relational Database Service.
- DBDeletionPolitique RDS — Spécifiez s'il faut supprimer ou capturer l'instance de base de données à la fin de l'environnement. Vous pouvez spécifier `Delete` ou `Snapshot`.

Exemple

Voici un exemple d'URL Launch Now. Après que vous avez construit la vôtre, vous pouvez la donner à vos utilisateurs. Par exemple, vous pouvez intégrer l'URL dans une page web ou dans des

documents de formation. Lorsque les utilisateurs créez une application à l'aide de l'URL Launch Now, l'assistant Elastic Beanstalk de création d'une application n'a besoin d'aucune entrée supplémentaire.

```
https://console.aws.amazon.com/elasticbeanstalk/home?region=us-west-2#/newApplication?
applicationName=YourCompanySampleApp
      &platform=PHP%207.3%20running%20on%2064bit%20Amazon%20Linux&sourceBundleUrl=
        http://s3.amazonaws.com/amzn-s3-demo-bucket/
myobject&environmentType=SingleInstance&tierName=WebServer
      &instanceType=m1.small&withVpc=true&withRds=true&rdsDBEngine=

 postgres&rdsDBAllocatedStorage=6&rdsDBInstanceClass=db.m1.small&rdsMultiAZDatabase=
          true&rdsDBDeletionPolicy=Snapshot
```

Pour utiliser l'URL Launch Now

1. Cliquez sur l'URL Launch Now.
2. Une fois que la console Elastic Beanstalk s'ouvre, sur la page Création d'une application web, cliquez sur Review and launch (Revoir et lancer) pour afficher les paramètres qu'Elastic Beanstalk utilise pour créer l'application et lancer l'environnement dans lequel l'application s'exécute.
3. Sur la page Configurer, choisissez Créer une application pour créer l'application.

Création et mise à jour de groupes d'environnements Elastic Beanstalk

Avec l' AWS Elastic Beanstalk ComposeEnvironmentsAPI, vous pouvez créer et mettre à jour des groupes d'environnements Elastic Beanstalk au sein d'une seule application. Chaque environnement du groupe peut exécuter un composant distinct d'une application d'architecture orientée services. L'API `Compose Environments` utilise une liste des versions d'application et un nom de groupe facultatif. Le service Elastic Beanstalk crée un environnement pour chaque version d'application. Si les environnements existent déjà, il y déploie les versions d'application.

Créez des liens entre les environnements Elastic Beanstalk afin de définir une dépendance entre deux environnements. Lorsque vous créez un groupe d'environnements avec l'API `Compose Environments`, Elastic Beanstalk crée des environnements dépendants, mais uniquement une fois que leurs dépendances sont opérationnelles. Pour de plus amples informations sur les liens d'environnement, veuillez consulter Création de liens entre les environnements Elastic Beanstalk.

L'API `Compose Environments` utilise un manifeste d'environnement pour stocker les détails de configuration qui sont partagés par les groupes d'environnements. Chaque composant doit inclure

un fichier de configuration `env.yaml` dans le bundle source de son application, qui spécifie les paramètres utilisés pour créer son environnement.

Pour utiliser `Compose Environments`, vous devez spécifier les paramètres `EnvironmentName` et `SolutionStack` dans le manifeste d'environnement pour chaque composant d'application.

Vous pouvez utiliser l'`Compose Environments`API avec l'interface de ligne de commande Elastic Beanstalk (EB CLI) AWS CLI, le ou un SDK. Pour obtenir les instructions sur l'interface de ligne de commande Elastic Beanstalk (EB CLI), veuillez consulter Gestion de plusieurs environnements Elastic Beanstalk en tant que groupe avec l'interface de ligne de commande EB.

Utilisation de l'API **`Compose Environments`**

Par exemple, vous pouvez créer une application nommée `Media Library` qui permet aux utilisateurs de charger et gérer des images et des vidéos stockées dans Amazon Simple Storage Service (Amazon S3). L'application dispose d'un environnement frontal, `front`, qui exécute une application web permettant aux utilisateurs de charger et de télécharger des fichiers individuels, de visualiser leur bibliothèque et d'initier des tâches de traitement par lots.

Au lieu de traiter les tâches directement, l'application frontale les ajoute à une file d'attente Amazon SQS. Le deuxième environnement, `worker`, extrait les tâches de la file d'attente et les traite. `worker` utilise un type d'instance G2 doté d'un GPU très performant, tandis que `front` peut s'exécuter sur un type d'instance générique plus économique.

Vous pouvez organiser le dossier du projet, `Media Library`, dans des répertoires distincts pour chaque composant. Chaque répertoire contient un fichier de définition d'environnement (`env.yaml`) avec le code source pour chacun d'eux :

```
~/workspace/media-library
|-- front
|   `-- env.yaml
`-- worker
    `-- env.yaml
```

Les listes suivantes spécifient le fichier `env.yaml` pour chaque composant de l'application.

`~/workspace/media-library/front/env.yaml`

```
EnvironmentName: front+
EnvironmentLinks:
  "WORKERQUEUE" : "worker+"
```

```
AWSConfigurationTemplateVersion: 1.1.0.0
EnvironmentTier:
  Name: WebServer
  Type: Standard
SolutionStack: 64bit Amazon Linux 2015.09 v2.0.4 running Java 8
OptionSettings:
  aws:autoscaling:launchconfiguration:
    InstanceType: m4.large
```

~/workspace/media-library/worker/env.yaml

```
EnvironmentName: worker+
AWSConfigurationTemplateVersion: 1.1.0.0
EnvironmentTier:
  Name: Worker
  Type: SQS/HTTP
SolutionStack: 64bit Amazon Linux 2015.09 v2.0.4 running Java 8
OptionSettings:
  aws:autoscaling:launchconfiguration:
    InstanceType: g2.2xlarge
```

Une fois que vous avez créé une version d'application pour les deux composants de l'application (frontal (`front-v1`) et travail (`worker-v1`)), vous devez appeler l'API `Compose Environments` avec les noms de version. Dans cet exemple, nous utilisons la AWS CLI pour appeler l'API.

```
# Create application versions for each component:
~$ aws elasticbeanstalk create-application-version --application-name media-
library --version-label front-v1 --process --source-bundle S3Bucket="amzn-s3-demo-
bucket",S3Key="front-v1.zip"
  {
    "ApplicationVersion": {
        "ApplicationName": "media-library",
        "VersionLabel": "front-v1",
        "Description": "",
        "DateCreated": "2015-11-03T23:01:25.412Z",
        "DateUpdated": "2015-11-03T23:01:25.412Z",
        "SourceBundle": {
            "S3Bucket": "amzn-s3-demo-bucket",
            "S3Key": "front-v1.zip"
        }
    }
  }
```

```
~$ aws elasticbeanstalk create-application-version --application-name media-library
 --version-label worker-v1 --process --source-bundle S3Bucket="amzn-s3-demo-
bucket",S3Key="worker-v1.zip"
  {
    "ApplicationVersion": {
        "ApplicationName": "media-library",
        "VersionLabel": "worker-v1",
        "Description": "",
        "DateCreated": "2015-11-03T23:01:48.151Z",
        "DateUpdated": "2015-11-03T23:01:48.151Z",
        "SourceBundle": {
            "S3Bucket": "amzn-s3-demo-bucket",
            "S3Key": "worker-v1.zip"
        }
    }
  }
# Create environments:
~$ aws elasticbeanstalk compose-environments --application-name media-library --group-
name dev --version-labels front-v1 worker-v1
```

Le troisième appel crée deux environnements : `front-dev` et `worker-dev`. L'API crée les noms des environnements en concaténant la valeur `EnvironmentName` spécifiée dans le fichier `env.yaml` avec l'option `group name` spécifiée dans l'appel `Compose Environments`, séparées par un tiret. La longueur totale de ces deux options et du trait d'union ne doit pas dépasser la longueur maximale autorisée pour le nom d'un environnement, soit 23 caractères.

L'application en cours d'exécution dans l'environnement `front-dev` peut accéder au nom de la file d'attente Amazon SQS; associée à l'environnement `worker-dev` en lisant la variable `WORKERQUEUE`. Pour de plus amples informations sur les liens d'environnement, veuillez consulter Création de liens entre les environnements Elastic Beanstalk.

Gestion de plusieurs environnements Elastic Beanstalk en tant que groupe avec l'interface de ligne de commande EB

Vous pouvez utiliser la CLI EB pour créer des groupes d' AWS Elastic Beanstalk environnements, chacun exécutant un composant distinct d'une application d'architecture orientée services. L'EB CLI gère ces groupes à l'aide de l'ComposeEnvironmentsAPI.

> **Note**
>
> Les groupes d'environnements sont différents des conteneurs multiples qui composent un environnement Docker multiconteneurs. Avec les groupes d'environnements, chaque composant de votre application s'exécute dans un environnement Elastic Beanstalk distinct, avec son propre ensemble d'instances Amazon EC2. Chaque composant peut être mis à l'échelle séparément. Avec Docker multiconteneurs, vous pouvez combiner plusieurs composants d'une application en un seul environnement. Tous les composants partagent le même ensemble d'instances Amazon EC2, et chaque instance exécute plusieurs conteneurs Docker. Choisissez l'une de ces architectures en fonction des besoins de votre application. Pour plus de détails sur Docker multi-conteneurs, consultez Utilisation de la branche de plateforme Docker gérée par ECS dans Elastic Beanstalk.

Organisez les composants de votre application selon la structure de dossier suivante :

```
~/project-name
|-- component-a
|   `-- env.yaml
`-- component-b
    `-- env.yaml
```

Chaque sous-dossier contient le code source d'un composant indépendant d'une application qui s'exécutera dans son propre environnement et un fichier de définition d'environnement nommé `env.yaml`. Pour de plus amples informations sur le format `env.yaml`, veuillez consulter Manifeste d'environnement (`env.yaml`).

Pour utiliser l'API `Compose Environments`, commencez par exécuter eb init à partir du dossier de projet, en spécifiant chaque composant par le nom du dossier qui le contient via l'option `--modules` :

```
~/workspace/project-name$ eb init --modules component-a component-b
```

L'interface de ligne de commande EB vous invite à configurer chaque composant, puis crée le répertoire `.elasticbeanstalk` dans le dossier de chaque composant. L'interface de ligne de commande EB ne crée pas les fichiers de configuration dans le répertoire parent.

```
~/project-name
|-- component-a
```

```
|   |-- .elasticbeanstalk
|   `-- env.yaml
`-- component-b
    |-- .elasticbeanstalk
    `-- env.yaml
```

Ensuite, exécutez la commande eb create avec une liste des environnements à créer (un pour chaque composant) :

```
~/workspace/project-name$ eb create --modules component-a component-b --env-group-suffix group-name
```

Cette commande crée un environnement pour chaque composant. Les noms des environnements sont créés en concaténant la valeur `EnvironmentName` spécifiée dans le fichier `env.yaml` avec le nom du groupe, séparé par un trait d'union. La longueur totale de ces deux options et du trait d'union ne doit pas dépasser la longueur maximale autorisée pour le nom d'un environnement, soit 23 caractères.

Pour mettre à jour l'environnement, utilisez la commande eb deploy :

```
~/workspace/project-name$ eb deploy --modules component-a component-b
```

Vous pouvez mettre à jour chaque composant individuellement ou les mettre à jour en tant que groupe. Spécifiez les composants que vous souhaitez mettre à jour avec l'option `--modules`.

L'interface de ligne de commande EB stocke le nom du groupe que vous avez utilisé avec eb create dans la section `branch-defaults` du fichier de configuration de l'interface de ligne de commande EB, sous `/.elasticbeanstalk/config.yml`. Pour déployer votre application vers un autre groupe, utilisez l'option `--env-group-suffix` lorsque vous exécutez eb deploy. Si le groupe n'existe pas encore, l'interface de ligne de commande EB crée un groupe d'environnements :

```
~/workspace/project-name$ eb deploy --modules component-a component-b --env-group-suffix group-2-name
```

Pour mettre des environnements hors service, exécutez eb terminate dans le dossier pour chaque module. Par défaut, l'interface de ligne de commande EB affiche une erreur si vous essayez de suspendre un environnement dont dépend un autre environnement en cours d'exécution. Commencez par suspendre l'environnement dépendant ou utilisez l'option `--ignore-links` pour remplacer le comportement par défaut :

```
~/workspace/project-name/component-b$ eb terminate --ignore-links
```

Déploiement d'applications dans des environnements Elastic Beanstalk

Vous pouvez utiliser la AWS Elastic Beanstalk console pour télécharger un bundle source mis à jour et le déployer dans votre environnement Elastic Beanstalk, ou redéployer une version précédemment téléchargée.

Chaque utilisateur est identifié par un ID de déploiement. Le déploiement IDs commence 1 et augmente d'une unité à chaque modification du déploiement et de la configuration de l'instance. Si vous activez les rapports améliorés sur l'état, Elastic Beanstalk affiche l'ID de déploiement dans la console de surveillance de l'état et dans la CLI EB quand il rapporte l'état de santé de l'instance. L'ID de déploiement vous aide à déterminer l'état de votre environnement lorsqu'une mise à jour propagée échoue.

Elastic Beanstalk fournit plusieurs stratégies et paramètres de déploiement. Pour de plus amples informations sur la configuration d'une stratégie et d'autres paramètres, veuillez consulter the section called “Options de déploiement”. Le tableau suivant répertorie les stratégies et les types d'environnements qui les prennent en charge.

Stratégies de déploiement prises en charge

Stratégie de déploiement	Environnements à charge équilibrée	Environnements d'instance unique	Environnements Windows Server existants†
Simultanément	✓ Oui	✓ Oui	✓ Oui
Propagation	✓ Oui	✗ Non	✓ Oui
Propagation avec un lot supplémentaire	✓ Oui	✗ Non	✗ Non
Immuable	✓ Oui	✓ Oui	✗ Non

Stratégie de déploiement	Environnements à charge équilibrée	Environnements d'instance unique	Environnements Windows Server existants†
Répartition du trafic	✓ Oui (Équilibreur de charge Application Load Balancer)	X Non	X Non

† Dans ce tableau, un environnement Windows Server existant est un environnement basé sur une configuration de plateforme Windows Server qui utilise une version d'IIS antérieure à IIS 8.5.

> **Warning**
>
> Certaines politiques remplacent toutes les instances pendant le déploiement ou la mise à jour. Cela entraîne la perte de tous les EC2 soldes brutaux accumulés sur Amazon. Une telle situation se produit dans les cas suivants :
>
> - Mises à jour de la plate-forme gérée avec le remplacement d'instance activé
> - Mises à jour immuables
> - Déploiements avec mises à jour immuables ou fractionnement du trafic activé

Choix d'une stratégie de déploiement

Le choix de la stratégie de déploiement adaptée pour votre application est un compromis entre plusieurs considérations et dépend de vos besoins particuliers. La page the section called “Options de déploiement” contient de plus amples informations sur chaque stratégie et une description détaillée du fonctionnement de certaines d'entre elles.

La liste suivante fournit des informations récapitulatives sur les différentes stratégies de déploiement et ajoute des considérations connexes.

- Simultanée – Méthode de déploiement la plus rapide. Convient si vous pouvez accepter une courte perte de service et si des déploiements rapides sont importants pour vous. Avec cette méthode, Elastic Beanstalk déploie la nouvelle version de l'application sur chaque instance. Ensuite, le proxy Web ou le serveur d'applications peut avoir besoin de redémarrer. Par conséquent, votre

application peut être indisponible pour les utilisateurs (ou présenter une faible disponibilité) pendant une courte période.

- Propagation – Évite les temps d'arrêt et minimise la disponibilité réduite, au prix d'un temps de déploiement plus long. Convient si vous ne pouvez accepter aucune période de service complètement perdu. Avec cette méthode, votre application est déployée dans votre environnement, un lot d'instances à la fois. La plupart de la bande passante est conservée tout au long du déploiement.
- Propagation avec un lot supplémentaire – Évite toute disponibilité réduite, au prix d'un temps de déploiement encore plus long que la méthode Propagation. Convient si vous devez conserver la même bande passante tout au long du déploiement. Avec cette méthode, Elastic Beanstalk lance un lot supplémentaire d'instances, puis effectue un déploiement par propagation. Le lancement du lot supplémentaire prend du temps et garantit que la même bande passante est conservée tout au long du déploiement.
- Immuable – Méthode de déploiement plus lente, qui garantit que la nouvelle version de votre application est toujours déployée sur les nouvelles instances, au lieu de mettre à jour les instances existantes. Elle présente également l'avantage supplémentaire d'une restauration rapide et sûre en cas d'échec du déploiement. Avec cette méthode, Elastic Beanstalk effectue une mise à jour immuable pour déployer votre application. Dans une mise à jour immuable, un second groupe Auto Scaling est lancé dans votre environnement et la nouvelle version sert le trafic parallèlement à l'ancienne version jusqu'à ce que les nouvelles instances transmettent les vérifications de l'état.
- Répartition du trafic – Méthode de déploiement avec tests Canary. Convient si vous souhaitez tester l'intégrité de la nouvelle version de votre application en utilisant une partie du trafic entrant, tout en conservant le reste du trafic servi par l'ancienne version de l'application.

Le tableau suivant compare les propriétés des méthodes de déploiement :

Méthodes de déploiement

Method	Impact d'un échec de déploiement	Temps de déploiement	Aucune interruption	Pas de modification DNS	Process de restaura ion	Code déployé sur
Simulta ment	Temps d'arrêt	🕒	✕ Non	✓ Oui	Redéplo ment manuel	Instances existante s
Propaga on	Lot unique hors service. Tout lot ayant abouti avant l'échec de l'exécution de la nouvelle version de l'application	🕒	✓ Oui	✓ Oui	Redéplo ment manuel	Instances existante s
Propaga on avec un lot supplén taire	Minime si le premier lot échoue, sinon similaire à Propagation.	🕒	✓ Oui	✓ Oui	Redéplo ment manuel	Instances nouvelles et existante s
Immuab	Minimale	🕒	✓ Oui	✓ Oui	Résilier les nouvelle instance	Nouvelles instances
Répartit on du trafic	Pourcentage du trafic client acheminé vers la nouvelle version temporairement affecté	🕒	✓ Oui	✓ Oui	Réache er le trafic et résilier les nouvelle instance	Nouvelles instances

Method	Impact d'un échec de déploiement	Temps de déploiement	Aucune interruption	Pas de modification DNS	Processus de restauration	Code déployé sur
Bleu/vert	Minimale	🕒	✓ Oui	X Non	Permutation URL	Nouvelles instances

† Varie en fonction de la taille du lot.

†† Varie en fonction du réglage de l'option du temps d'évaluation.

Déploiement d'une nouvelle version de l'application

Vous pouvez effectuer des déploiements à partir du tableau de bord de votre environnement.

Pour déployer une nouvelle version de l'application dans un environnement Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Upload and Deploy (Charger et déployer).
4. Utilisez le formulaire à l'écran pour télécharger le bundle source de l'application.
5. Choisissez Déployer.

Redéploiement d'une version précédente

Vous pouvez également déployer une version précédemment chargée de votre application sur n'importe lequel de ses environnements depuis la page des versions d'application.

Pour déployer une version de l'application existante dans un environnement existant

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS

2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.
3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Application versions (Versions d'application).
4. Sélectionnez la version d'application à déployer.
5. Choisissez Actions, puis Deploy (Déployer).
6. Choisissez un environnement, puis Deploy (Déployer).

Autres méthodes de déploiement de votre application

Si vous procédez à des déploiements fréquents, pensez à utiliser l'interface de ligne de commande Elastic Beanstalk (CLI EB) pour gérer vos environnements. La CLI EB crée un référentiel parallèlement à votre code source. Elle peut également créer un bundle de fichiers source, le charger dans Elastic Beanstalk et le déployer à l'aide d'une seule commande.

Pour les déploiements qui reposent sur des modifications de configuration de ressources ou une nouvelle version qui ne peut pas s'exécuter parallèlement à l'ancienne version, vous pouvez lancer un nouvel environnement avec la nouvelle version et effectuer un échange CNAME pour un déploiement bleu/vert.

Pour automatiser vos processus de création, de test et de déploiement, vous pouvez implémenter l'intégration continue et le déploiement continu (CI/CD) dans votre environnement Elastic Beanstalk. Pour de plus amples informations, veuillez consulter CI/CD Implémentation de l'intégration avec votre environnement Elastic Beanstalk.

Paramètres et stratégies de déploiement

AWS Elastic Beanstalk propose plusieurs options pour le traitement des déploiements, notamment des politiques de déploiement (tout en une fois, roulement, lancement avec un lot supplémentaire, immuable et division du trafic) et des options qui vous permettent de configurer la taille des lots et le comportement des contrôles de santé pendant les déploiements. Par défaut, votre environnement utilise des all-at-once déploiements. Si vous avez créé l'environnement avec la CLI EB et s'il s'agit d'un environnement évolutif (vous n'avez pas spécifié l'option `--single`), il utilise des déploiement par propagation.

Grâce aux déploiements progressifs, Elastic Beanstalk divise les instances Amazon EC2 de l'environnement en lots et déploie la nouvelle version de l'application en un lot à la fois. Les instances

restantes sont laissées dans l'environnement exécutant l'ancienne version de l'application. Lors d'un déploiement par propagation, certaines instances traitent les demandes avec l'ancienne version d'application, tandis que les instances incluses dans les lots terminés traitent les autres demandes avec la nouvelle version. Pour en savoir plus, consultez the section called “Fonctionnement de la propagation des déploiements”.

Afin de maintenir une pleine capacité au cours des déploiements, vous pouvez configurer votre environnement de façon à lancer un nouveau lot d'instances avant de mettre des instances hors service. Cette option correspond à un déploiement par propagation avec un lot supplémentaire. Une fois le déploiement terminé, Elastic Beanstalk résilie le lot d'instances supplémentaire.

Les déploiements immuables effectuent une mise à jour immuable pour lancer un ensemble complet de nouvelles instances exécutant la nouvelle version de l'application dans un groupe Auto Scaling distinct, parallèlement aux instances qui exécutent l'ancienne version. Les déploiements immuables peuvent éviter les problèmes causés par des propagations de déploiements partiellement terminées. Si les nouvelles instances ne réussissent pas les vérifications de l'état, Elastic Beanstalk les résilie tout en laissant intactes les instances d'origine.

Les déploiements avec répartition du trafic vous permettent d'effectuer des tests Canary dans le cadre du déploiement de votre application. Dans un déploiement avec répartition du trafic, Elastic Beanstalk lance un ensemble complet de nouvelles instances comme lors d'un déploiement immuable. Il transmet ensuite un pourcentage spécifié du trafic client entrant vers la nouvelle version de l'application pour une période d'évaluation spécifiée. Si les nouvelles instances restent en bonne santé, Elastic Beanstalk leur transmet l'ensemble du trafic et résilie les anciennes instances. Si les nouvelles instances ne réussissent pas les vérifications de l'état ou si vous choisissez d'abandonner le déploiement, Elastic Beanstalk renvoie le trafic vers les anciennes instances et résilie les nouvelles. Il n'y a jamais d'interruption de service. Pour en savoir plus, consultez the section called “Fonctionnement des déploiements avec répartition du trafic”.

> **Warning**
>
> Certaines politiques remplacent toutes les instances pendant le déploiement ou la mise à jour. Cela entraîne la perte de tous EC2 les soldes cumulés d'Amazon. Une telle situation se produit dans les cas suivants :
>
> - Mises à jour de la plate-forme gérée avec le remplacement d'instance activé
> - Mises à jour immuables
> - Déploiements avec mises à jour immuables ou fractionnement du trafic activé

Si votre application ne réussit pas toutes les vérifications d'état, mais fonctionne tout de même correctement avec un statut d'état inférieur, vous pouvez autoriser les instances à réussir les vérifications d'état avec un statut inférieur (`Warning`, par exemple), en modifiant l'option Seuil de bonne santé. Si vos déploiements échouent parce qu'ils ne réussissent pas les vérifications d'état et si vous avez besoin de forcer une mise à jour quel que soit le statut de l'état, sélectionnez l'option Ignorer la vérification de l'état.

Lorsque vous spécifiez une taille de lot pour la propagation des mises à jour, Elastic Beanstalk utilise également cette valeur pour les redémarrages progressifs de l'application. Utilisez les redémarrages progressifs lorsque vous souhaitez redémarrer les serveurs proxy et d'application exécutés sur les instances de votre environnement sans temps d'arrêt.

Configuration des déploiements d'application

Dans la console de gestion de l'environnement, activez et configurez les déploiements de version d'application par lots en modifiant l'option Updates and Deployments (Mises à jour et déploiements) sur la page Configuration de l'environnement.

Pour configurer des déploiements (console)

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Rolling updates and deployments (Propagation des mises à jour et déploiements), choisissez Edit (Modifier).
5. Dans la section Déploiements de l'application, choisissez une valeur pour l'option Stratégie de déploiement, les paramètres de lots et les options de vérification de l'état.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

La section Déploiements de l'application de la page Propagation des mises à jour et déploiements propose les options suivantes pour les déploiements d'applications :

- Deployment policy (Stratégie de déploiement) – Choisissez l'une des options de déploiement suivantes :

- All at once (Simultanée) – Déployez la nouvelle version dans toutes les instances en même temps. Pendant le déploiement, toutes les instances de votre environnement sont mises hors service pendant un temps limité.
- Rolling (Propagation) – Déployez la nouvelle version par lots. Chaque lot est suspendu pendant la phase de déploiement. Par conséquent, la capacité de votre environnement est diminuée du nombre d'instances dans un lot.
- Rolling with additional batch (Propagation avec un lot supplémentaire) – Déployez la nouvelle version par lots, en commençant par lancer un nouveau lot d'instances afin de garantir la pleine capacité au cours du processus de déploiement.
- Immutable (Immuable) – Déployez la nouvelle version dans un nouveau groupe d'instances en effectuant une mise à jour immuable.
- Traffic splitting (Répartition du trafic) – Déployez la nouvelle version dans un nouveau groupe d'instances et répartissez temporairement le trafic client entrant entre la version de l'application existante et la nouvelle.

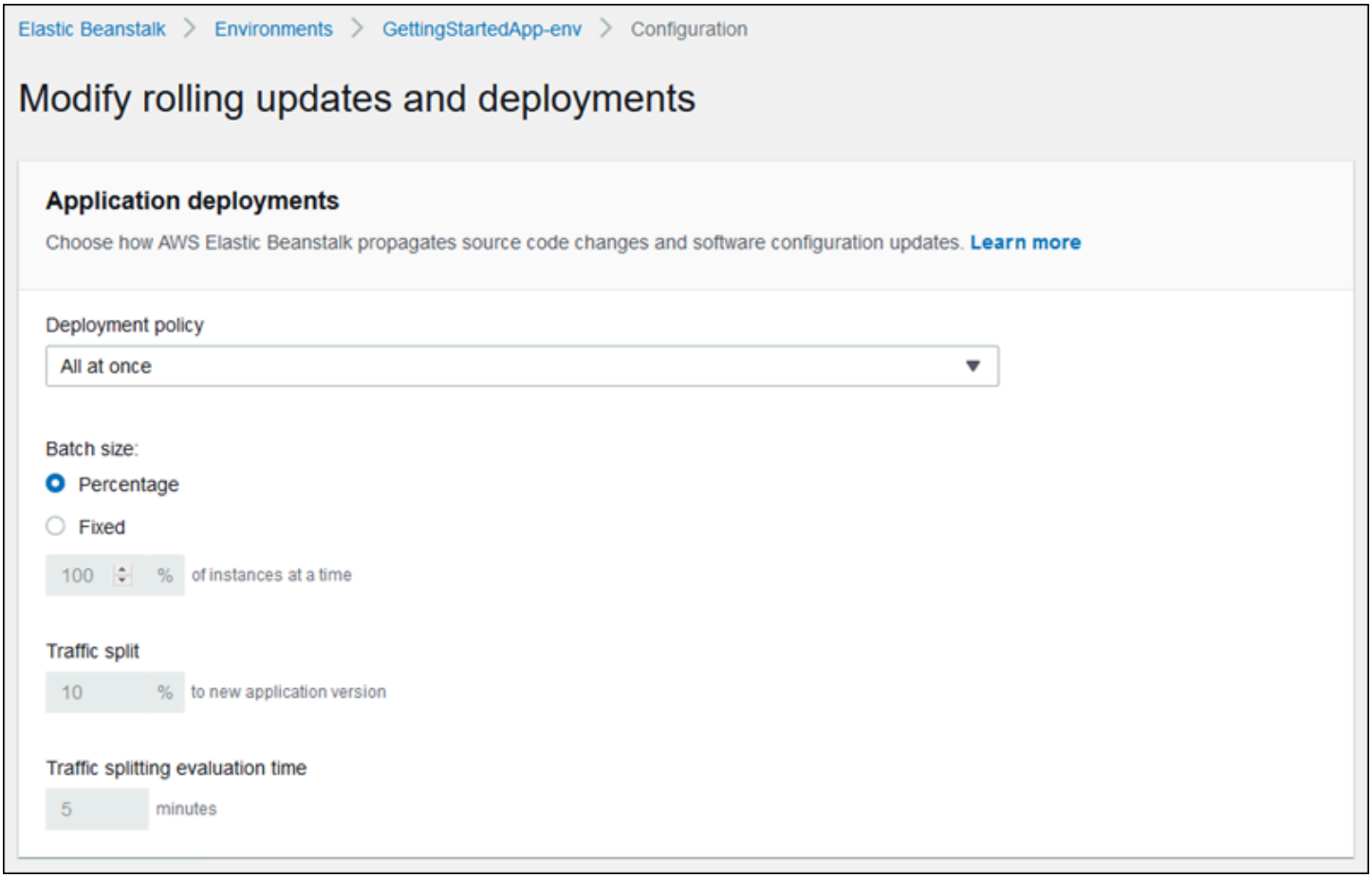
Pour les stratégies de déploiement Propagation et Propagation avec un lot supplémentaire, vous pouvez configurer :

- Batch size (Taille de lot) – Taille de l'ensemble d'instances à déployer dans chaque lot.

  Choisissez Pourcentage pour configurer un pourcentage du nombre total d' EC2 instances du groupe Auto Scaling (jusqu'à 100 %), ou choisissez Fixed pour configurer un nombre fixe d'instances (jusqu'au nombre maximum d'instances dans la configuration Auto Scaling de votre environnement).

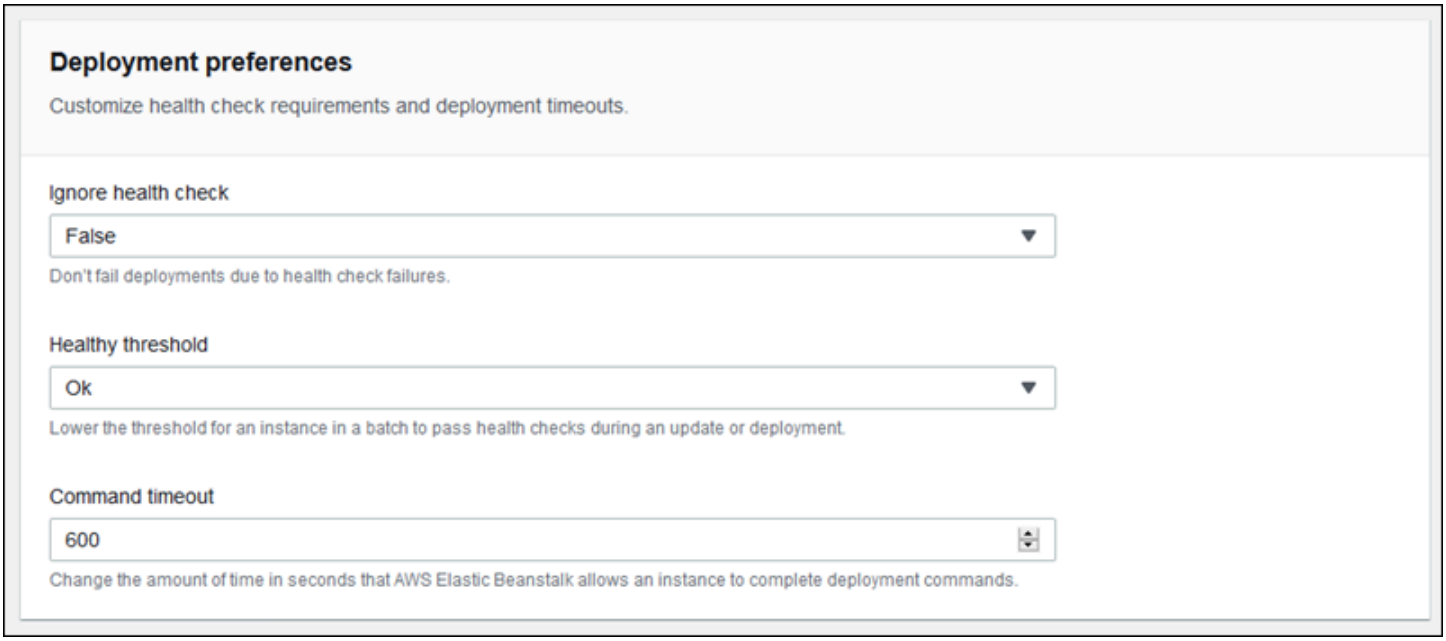
Pour la stratégie de déploiement Répartition du trafic, vous pouvez configurer les éléments suivants :

- Traffic split (Répartition du trafic) – Pourcentage initial du trafic client entrant que Elastic Beanstalk déplace vers les instances d'environnement exécutant la nouvelle version de l'application que vous déployez.
- Traffic splitting evaluation time (Temps d'évaluation de la répartition du trafic) – Période d'attente d'Elastic Beanstalk, en minutes, à la suite d'un déploiement sain initial avant de déplacer l'ensemble du trafic client entrant vers la nouvelle version de l'application que vous déployez.



La section Préférences de déploiement inclut des options relatives aux vérifications de l'état.

- Ignore health check (Ignorer la vérification de l'état) – Empêche la restauration du déploiement lorsqu'un lot ne passe pas avec succès les vérifications de l'état avant l'expiration du délai de commande.
- Healthy threshold (Seuil de bonne santé) – Réduit le seuil au niveau auquel une instance est considérée comme saine pendant les déploiements par propagation, les mises à jour par propagation et les mises à jour immuables.
- Command timeout (Délai de commande) – Nombre de secondes d'attente pour qu'une instance devienne saine avant d'annuler le déploiement ou, si l'option Ignore health check (Ignorer la vérification de l'état) est sélectionnée, avant de passer au lot suivant.



Fonctionnement de la propagation des déploiements

Lors du traitement d'un lot, Elastic Beanstalk détache de l'équilibreur de charge toutes les instances du lot, déploie la nouvelle version de l'application, puis attache de nouveau les instances. Si vous activez le drainage des connexions, Elastic Beanstalk vide les connexions existantes des instances EC2 Amazon de chaque lot avant de commencer le déploiement.

Après avoir associé de nouveau les instances d'un lot à l'équilibreur de charge, Elastic Load Balancing attend qu'elles réussissent un nombre minimal de vérifications de l'état Elastic Load Balancing (valeur du paramètre Healthy check count threshold (Seuil du nombre de vérifications de l'état saines)), puis commence à acheminer le trafic vers ces instances. Si aucune URL de vérification de l'état n'est configurée, cela peut se produire très rapidement, car une instance réussit la vérification de l'état dès qu'elle peut accepter une connexion TCP. Si une URL de vérification de l'état est configurée, l'équilibreur de charge n'achemine pas le trafic vers les instances mises à jour tant qu'elles n'ont pas renvoyé un code d'état `200 OK` en réponse à une demande `HTTP GET` envoyée à l'URL de vérification de l'état.

Elastic Beanstalk attend que toutes les instances du lot soient saines avant de passer au lot suivant. Avec les rapports de base sur l'état, l'état de l'instance dépend du statut de vérification de l'état d'Elastic Load Balancing. Lorsque toutes les instances du lot ont réussi suffisamment de vérifications de l'état pour être considérées comme saines par Elastic Load Balancing, le traitement du lot est terminé. Si les rapports améliorés sur l'état sont activés, Elastic Beanstalk tient compte de plusieurs autres facteurs, dont le résultat des demandes entrantes. Avec les rapports sur l'état de santé

améliorés, toutes les instances doivent réussir 12 vérifications de l'état consécutives avec un statut OK en deux minutes pour les environnements de serveurs web et 18 vérifications de l'état en 3 minutes pour les environnements de travail.

Si un lot d'instances ne devient pas sain avant l'expiration du délai de commande, le déploiement échoue. Dès lors qu'un déploiement a échoué, vérifiez l'état des instances de votre environnement pour obtenir des informations sur la cause de la défaillance. Ensuite, effectuez un autre déploiement avec une version corrigée de votre application à restaurer ou une version dont vous savez qu'elle est correcte.

Si un déploiement échoue alors qu'un ou plusieurs lots ont été correctement créés, les lots terminés exécutent la nouvelle version de votre application, tandis que les lots en attente continuent d'exécuter l'ancienne version. Vous pouvez identifier la version en cours d'exécution sur les instances de votre environnement sur la page Santé de la console. Cette page affiche l'ID du déploiement le plus récent ayant été exécuté sur chaque instance de votre environnement. Si vous résiliez des instances à partir du déploiement ayant échoué, Elastic Beanstalk les remplace par des instances exécutant la version de l'application à partir du déploiement le plus récent ayant réussi.

Fonctionnement des déploiements avec répartition du trafic

Les déploiements avec répartition du trafic vous permettent d'effectuer des tests Canary. Vous dirigez une partie du trafic client entrant vers la nouvelle version de votre application pour vérifier l'état de l'application avant de valider la nouvelle version et de diriger tout le trafic vers celle-ci.

Au cours d'un déploiement avec répartition du trafic, Elastic Beanstalk crée un nouvel ensemble d'instances dans un groupe Auto Scaling temporaire distinct. Elastic Beanstalk demande ensuite à l'équilibreur de charge de diriger un certain pourcentage du trafic entrant de votre environnement vers les nouvelles instances. Ensuite, Elastic Beanstalk contrôle l'état du nouvel ensemble d'instances pendant une durée configurée. Si tout va bien, Elastic Beanstalk déplace le trafic restant vers les nouvelles instances et les attache au groupe Auto Scaling d'origine de l'environnement, en remplaçant les anciennes instances. Elastic Beanstalk procède ensuite à un nettoyage en résiliant les anciennes instances et en supprimant le groupe Auto Scaling temporaire.

> **Note**
>
> La capacité de l'environnement ne change pas lors d'un déploiement avec répartition du trafic. Elastic Beanstalk lance le même nombre d'instances dans le groupe Auto Scaling temporaire que dans le groupe Auto Scaling d'origine au début du déploiement. Il maintient ensuite un nombre constant d'instances dans les deux groupes Auto Scaling pour toute la

durée du déploiement. Tenez compte de ce fait lors de la configuration du temps d'évaluation de la répartition du trafic de l'environnement.

La restauration du déploiement vers la version précédente de l'application est rapide et n'affecte pas le service sur le trafic client. Si les nouvelles instances ne réussissent pas les vérifications de l'état ou si vous choisissez d'abandonner le déploiement, Elastic Beanstalk renvoie le trafic vers les anciennes instances et résilie les nouvelles. Pour abandonner un déploiement, accédez à la page de présentation de l'environnement dans la console Elastic Beanstalk et choisissez Abort current operation (Interrompre l'opération en cours) dans Environment actions (Actions d'environnement). Vous pouvez également appeler l'AbortEnvironmentUpdateAPI ou la AWS CLI commande équivalente.

Les déploiements avec répartition du trafic exigent un équilibreur de charge Application Load Balancer. Elastic Beanstalk utilise ce type d'équilibreur de charge par défaut lorsque vous créez votre environnement à l'aide de la console Elastic Beanstalk ou de l'interface de ligne de commande (CLI) EB.

Espaces de noms pour les options de déploiements

Vous pouvez utiliser les options de configuration dans l'espace de noms `aws:elasticbeanstalk:command` pour configurer vos déploiements. Si vous choisissez la stratégie avec répartition du trafic, des options supplémentaires pour cette stratégie sont disponibles dans l'espace de noms `aws:elasticbeanstalk:trafficsplitting`.

Utilisez l'option `DeploymentPolicy` pour définir le type de déploiement. Les valeurs suivantes sont prises en charge :

- `AllAtOnce` – Désactive les propagations de déploiements et effectue systématiquement un déploiement dans toutes les instances en même temps.
- `Rolling` – Permet d'effectuer des propagations de déploiements standard.
- `RollingWithAdditionalBatch` – Lance un lot d'instances supplémentaire avant de commencer le déploiement afin de maintenir une pleine capacité.
- `Immutable` – Effectue une mise à jour immuable pour chaque déploiement.
- `TrafficSplitting` – Effectue des déploiements avec répartition du trafic pour réaliser des tests Canary sur les déploiements de votre application.

Lorsque vous activez les propagations de déploiements, définissez les options `BatchSize` et `BatchSizeType` afin de configurer la taille de chaque lot. Par exemple, pour déployer 25 % de toutes les instances dans chaque lot, spécifiez les options et valeurs suivantes.

Example.ebextensions/rolling-updates.config

```
option_settings:
  aws:elasticbeanstalk:command:
    DeploymentPolicy: Rolling
    BatchSizeType: Percentage
    BatchSize: 25
```

Pour déployer jusqu'à cinq instances dans chaque lot, quel que soit le nombre d'instances en cours d'exécution, et mettre en place un lot supplémentaire de cinq instances exécutant la nouvelle version avant de mettre des instances hors service, spécifiez les options et valeurs ci-après.

Example.ebextensions/rolling-additionalbatch.config

```
option_settings:
  aws:elasticbeanstalk:command:
    DeploymentPolicy: RollingWithAdditionalBatch
    BatchSizeType: Fixed
    BatchSize: 5
```

Pour effectuer une mise à jour immuable pour chaque déploiement avec un seuil de vérification de l'état défini sur Avertissement et poursuivre le déploiement même si les instances d'un lot ne réussissent pas les vérifications de l'état dans un délai de 15 minutes, spécifiez les options et les valeurs ci-après.

Example.ebextensions/immutable-ignorehealth.config

```
option_settings:
  aws:elasticbeanstalk:command:
    DeploymentPolicy: Immutable
    HealthCheckSuccessThreshold: Warning
    IgnoreHealthCheck: true
    Timeout: "900"
```

Pour effectuer des déploiements avec répartition du trafic, avec le transfert de 15 % du trafic client vers la nouvelle version de l'application et l'évaluation de l'intégrité pendant 10 minutes, spécifiez les options et valeurs suivantes.

Example.ebextensions/traffic-splitting.config

```
option_settings:
  aws:elasticbeanstalk:command:
    DeploymentPolicy: TrafficSplitting
  aws:elasticbeanstalk:trafficsplitting:
    NewVersionPercent: "15"
    EvaluationTime: "10"
```

L'interface de ligne de commande (CLI) EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Consultez Valeurs recommandées pour plus de détails.

Déploiements bleu/vert avec Elastic Beanstalk

Dans la AWS Elastic Beanstalk mesure où une mise à jour est effectuée sur place lorsque vous mettez à jour les versions de votre application, celle-ci peut devenir indisponible pour les utilisateurs pendant une courte période. Pour éviter cela, effectuez un blue/green déploiement. Pour ce faire, déployez la nouvelle version dans un environnement distinct, puis échangez les CNAMEs deux environnements pour rediriger instantanément le trafic vers la nouvelle version.

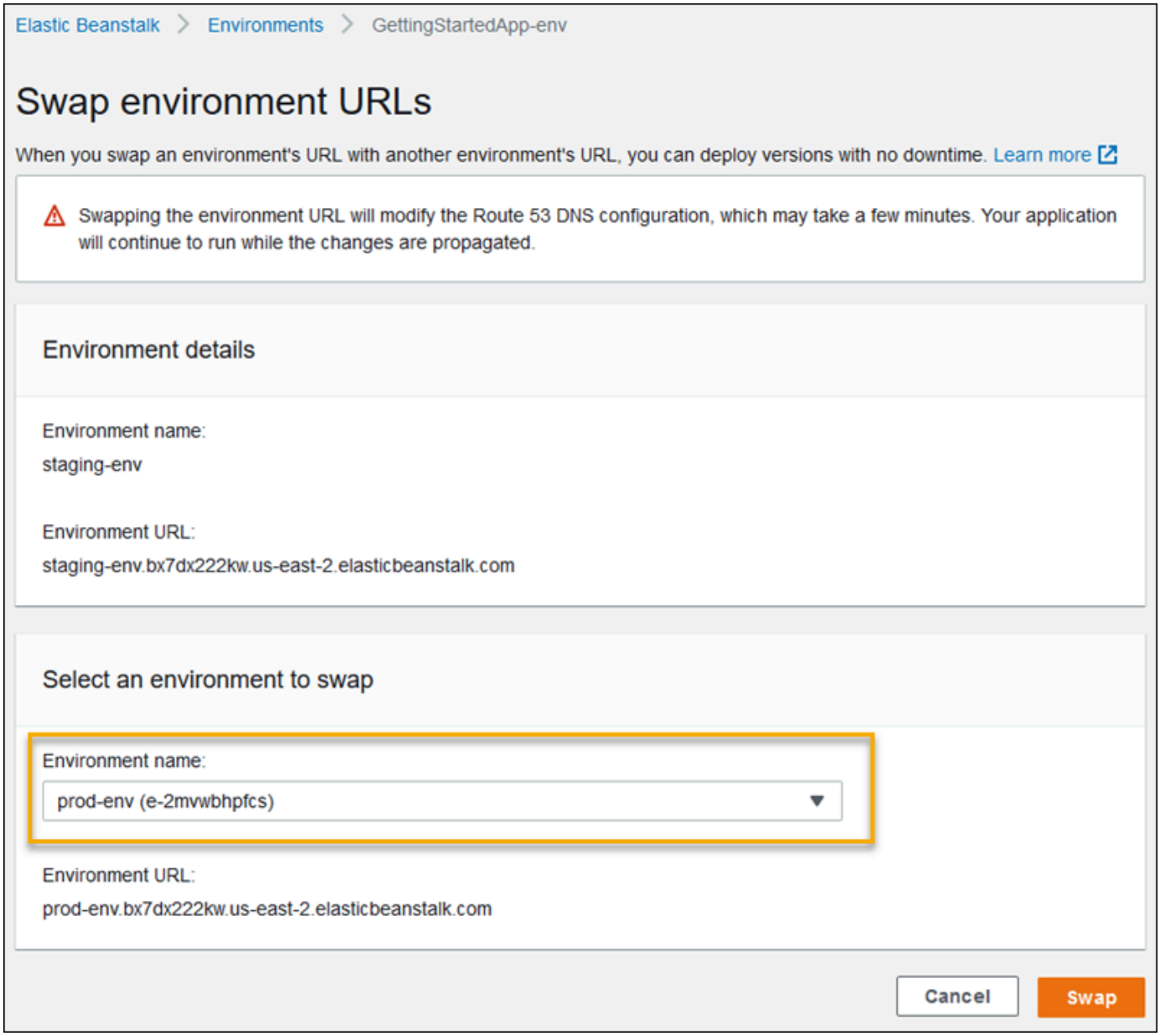
Un blue/green déploiement est également nécessaire si vous souhaitez mettre à jour un environnement vers une version de plate-forme incompatible. Pour de plus amples informations, veuillez consulter the section called “Mises à jour de plateforme”.

Les déploiements bleu/vert nécessitent que votre environnement s'exécute indépendamment de votre base de données de production, si votre application en utilise une. Si votre environnement inclut une base de données créée par Elastic Beanstalk en votre nom, la base de données et la connexion de l'environnement ne sont pas préservées, sauf si vous effectuez des actions spécifiques. Si vous possédez une base de données que vous souhaitez conserver, utilisez l'une des options de cycle de vie de base de données Elastic Beanstalk. Vous pouvez choisir l'option Retain (Conserver) afin de garder la base de données et l'environnement opérationnels après le découplage de la base de données. Pour de plus amples informations, consultez Cycle de vie de base de données dans le chapitre Configuration des environnements de ce guide.

Pour de plus amples informations sur la configuration de votre application pour la connecter à une instance Amazon RDS qui n'est pas gérée par Elastic Beanstalk, consultez Utilisation d'Elastic Beanstalk avec Amazon RDS.

Pour effectuer un blue/green déploiement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Clonez votre environnement actuel ou lancez un nouvel environnement pour exécuter la version de la plateforme souhaitée.
3. Déployez la nouvelle version d'application dans le nouvel environnement.
4. Testez la nouvelle version sur le nouvel environnement.
5. Sur la page de présentation de l'environnement, choisissez Actions, puis Swap environment URLs.
6. Dans Environment name (Nom de l'environnement), sélectionnez l'environnement actuel.



7. Choisissez Permuter.

Elastic Beanstalk échange les enregistrements CNAME de l'ancien et du nouvel environnement, en redirigeant le trafic depuis l'ancienne version vers la nouvelle version.

Une fois l'opération d'échange effectuée par Elastic Beanstalk, assurez-vous que le nouvel environnement répond lorsque vous essayez de vous connecter à l'URL de l'ancien environnement. Toutefois, avant de résilier votre ancien environnement, attendez que les modifications DNS se propagent et que vos anciens enregistrements DNS arrivent à expiration. Les serveurs DNS

n'effacent pas toujours les anciens enregistrements de leur cache, selon la durée de vie (TTL, time-to-live) que vous avez définie dans vos enregistrements DNS.

CI/CD Implémentation de l'intégration avec votre environnement Elastic Beanstalk

Elastic Beanstalk s'intègre à de nombreux outils pour automatiser le flux de travail CI/CD de développement de vos applications. CI/CD les pratiques vous permettent de créer, de tester et de déployer automatiquement vos applications avec une intervention manuelle minimale. Continuous delivery/deployment (CD) étend l'intégration continue (CI) en automatisant le processus de déploiement. Vous pouvez créer des pipelines de déploiement rationalisés à l'aide de AWS services tels que Jenkins CodePipeline ou d'outils tiers tels que Jenkins, GitLab afin de garantir des déploiements cohérents et fiables dans vos environnements Elastic Beanstalk.

Rubriques

- AWS sources pour commencer
- Ressources supplémentaires
- Utilisation d' GitHub actions pour le déploiement sur Elastic Beanstalk

AWS sources pour commencer

La liste suivante présente les CI/CD outils et les AWS ressources correspondantes qui fournissent des step-by-step conseils pour créer des pipelines de déploiement automatisés dans les environnements Elastic Beanstalk :

- AWS CodePipeline— Ce didacticiel du centre de ressources AWS Getting Started explique comment configurer un pipeline de déploiement continu vers Elastic GitHub Beanstalk à partir de S3 ou. AWS CodeCommit
- GitHub Actions — Découvrez comment configurer Utilisation d' GitHub actions pour le déploiement sur Elastic Beanstalk des flux de travail basés sur YAML afin de configurer un pipeline de déploiement continu vers Elastic Beanstalk directement depuis. GitHub
- GitLab— Ce billet de blog consacré à la productivité des AWS DevOps développeurs explique comment configurer des pipelines GitLab continus pour déployer des applications Node.js dans les environnements Elastic Beanstalk Docker.

- Azure DevOps — Ce billet .NET on AWS Blog vous explique comment implémenter un pipeline de déploiement continu à partir d'un référentiel Azure DevOps Git vers Elastic Beanstalk à l'aide d'Azure Pipelines.

Ressources supplémentaires

Les outils et ressources tiers suivants peuvent vous aider à implémenter des pipelines de déploiement automatisés dans les environnements Elastic Beanstalk :

- Jenkins — Le plugin AWS EBDeployment Jenkins permet un déploiement direct dans les environnements Elastic Beanstalk à partir de votre page Jenkins Job Configuration.
- Circle CI : — The Orbs for Elastic Beanstalk fournit des packages de configuration réutilisables pour déployer et dimensionner des applications sur Elastic Beanstalk.
- Bitbucket Pipelines — L'article Deploy Elastic Beanstalk Application using Bitbucket Pipelines fournit un exemple de configuration de base pour implémenter des pipelines Bitbucket avec Elastic Beanstalk.

Utilisation d' GitHub actions pour le déploiement sur Elastic Beanstalk

GitHub Les actions peuvent déployer automatiquement votre application sur Elastic Beanstalk lorsque vous envoyez des modifications de code à votre référentiel. L'action Elastic Beanstalk Deploy fournit une interface YAML simple qui gère la création de versions d'applications, le téléchargement de bundles de sources sur Amazon S3 et le déploiement dans votre environnement Elastic Beanstalk.

Exemple de flux de travail

L'exemple de flux de travail suivant déploie une application dans un environnement Elastic Beanstalk chaque fois que vous la transférez vers la branche. `main` Créez un `.yml` fichier dans votre dépôt sous`.github/workflows/`.

Example GitHub Workflow d'actions pour le déploiement d'Elastic Beanstalk

```
name: Deploy to Elastic Beanstalk

on:
  push:
    branches:
```

```
      - main

permissions:
  id-token: write
  contents: read

jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout code
        uses: actions/checkout@v4

      - name: Configure AWS credentials
        uses: aws-actions/configure-aws-credentials@v4
        with:
          role-to-assume: arn:aws:iam::123456789012:role/my-github-actions-role
          aws-region: us-east-1

      - name: Deploy to Elastic Beanstalk
        uses: aws-actions/aws-elasticbeanstalk-deploy@v1.0.0
        with:
          aws-region: us-east-1
          application-name: my-application
          environment-name: my-application-env
```

Ce flux de travail extrait votre référentiel, utilise OpenID Connect (OIDC) pour s'authentifier AWS via l'action Configure AWS Credentials, puis déploie votre application sur Elastic Beanstalk. L'action de déploiement regroupe le contenu de votre référentiel, télécharge le bundle source sur Amazon S3, crée une nouvelle version de l'application et crée ou met à jour votre environnement. Par défaut, il attend que le déploiement soit terminé et que l'environnement revienne à un état sain.

Pour plus d'options de configuration et des exemples avancés, consultez l'action README sur laquelle vous pouvez accéder à Elastic Beanstalk Deploy. GitHub

Ressources supplémentaires

- Action de déploiement d'Elastic Beanstalk sur GitHub
- Action de configuration AWS des informations d'identification sur GitHub
- Configuration d'OpenID Connect dans Amazon Web Services (documentation) GitHub

Configuration changes

Lorsque vous modifiez les paramètres des options de configuration dans la section Configuration de la console de gestion de l'environnement, AWS Elastic Beanstalk les modifications sont répercutées sur toutes les ressources concernées. Ces ressources incluent l'équilibreur de charge qui distribue le trafic vers les EC2 instances Amazon exécutant votre application, le groupe Auto Scaling qui gère ces instances et les EC2 instances elles-mêmes.

De nombreux changements de configuration peuvent s'appliquer à un environnement en cours d'exécution sans remplacer les instances existantes. Par exemple, la définition d'une URL de vérification de l'état déclenche une mise à jour de l'environnement pour modifier les paramètres de l'équilibreur de charge, mais n'entraîne aucun temps d'arrêt, car les instances qui exécutent votre application continuent à traiter les demandes pendant la propagation de la mise à jour.

Les changements de configuration qui modifient la configuration du lancement ou les paramètres de VPC nécessitent de résilier toutes les instances de votre environnement et de les remplacer. Par exemple, lorsque vous modifiez le type d'instance ou le paramètre de clé SSH de votre environnement, les EC2 instances doivent être résiliées et remplacées. Elastic Beanstalk fournit plusieurs stratégies qui déterminent la façon dont ce remplacement est effectué.

- Mises à jour propagées – Elastic Beanstalk applique vos modifications de configuration par lots, en conservant un nombre minimal d'instances en cours d'exécution et en servant le trafic à tout moment. Cette approche empêche les temps d'arrêt pendant le processus de mise à jour. Pour en savoir plus, consultez Mises à jour continues.
- Mises à jour immuables – Elastic Beanstalk lance un groupe Auto Scaling temporaire en dehors de votre environnement avec un ensemble distinct d'instances s'exécutant avec la nouvelle configuration. Ensuite, Elastic Beanstalk place ces instances derrière l'équilibreur de charge de votre environnement. Les anciennes et les nouvelles instances servent le trafic jusqu'à ce que les nouvelles instances réussissent les contrôles d'intégrité. À ce moment-là, Elastic Beanstalk déplace les nouvelles instances dans le groupe Auto Scaling de votre environnement et résilie le groupe temporaire et les anciennes instances. Pour en savoir plus, consultez Mises à jour immuables.
- Désactivé – Elastic Beanstalk ne tente pas d'éviter les temps d'arrêt. Il met fin aux instances existantes de votre environnement et les remplace par de nouvelles instances exécutées avec la nouvelle configuration.

> **Warning**
>
> Certaines politiques remplacent toutes les instances pendant le déploiement ou la mise à jour. Cela entraîne la perte de tous EC2 les soldes cumulés d'Amazon. Une telle situation se produit dans les cas suivants :
>
> - Mises à jour de la plate-forme gérée avec le remplacement d'instance activé
> - Mises à jour immuables
> - Déploiements avec mises à jour immuables ou fractionnement du trafic activé

Types de mise à jour pris en charge

Paramètre de mise à jour propagée	Environnements à charge équilibrée	Environnements d'instance unique	Environnements Windows Server existants†
Désactivé	✓ Oui	✓ Oui	✓ Oui
Propagation en fonction de la santé	✓ Oui	X Non	✓ Oui
Propagation en fonction de la durée	✓ Oui	X Non	✓ Oui
Immuable	✓ Oui	✓ Oui	X Non

† Pour les besoins de cette table, un environnement Windows Server existant est un environnement basé sur une configuration de plateforme Windows Server qui utilise une version IIS antérieure à IIS 8.5.

Rubriques

- Mises à jour propagées de la configuration de l'environnement Elastic Beanstalk
- Mises à jour immuables de l'environnement

Mises à jour propagées de la configuration de l'environnement Elastic Beanstalk

Lorsqu'une modification de configuration nécessite de remplacer les instances, Elastic Beanstalk peut effectuer la mise à jour par lots afin d'éviter les temps d'arrêt pendant que la modification est propagée. Pendant une mise à jour propagée, la capacité est réduite uniquement de la taille d'un seul lot, que vous pouvez configurer. Elastic Beanstalk prend un lot d'instances hors service, le résilie, puis lance un lot avec la nouvelle configuration. Une fois que le nouveau lot commence à traiter les demandes, Elastic Beanstalk passe au lot suivant.

Les lots de mise à jour de configuration peuvent être traités périodiquement (en fonction du temps), avec un délai entre chaque lot, ou sur selon l'intégrité. Pour les mises à jour propagées basées sur le temps, vous pouvez configurer la durée d'attente pour Elastic Beanstalk, après le lancement d'un lot d'instances avant de passer au lot suivant. Ce temps d'interruption permet à votre application d'amorcer et de commencer à traiter les demandes.

Avec des mises à jour propagées basées sur l'état, Elastic Beanstalk attend que des instances dans un lot transmettent des vérifications de l'état avant de passer au lot suivant. L'intégrité d'une instance est déterminée par le système de création de rapports d'intégrité, qui peut être de base ou améliorée. Avec l'état de base, un lot est considéré comme sain dès que toutes les instances qu'il contient passent les vérifications de l'état Elastic Load Balancing (ELB).

Avec la création de rapports d'état amélioré, toutes les instances dans un lot doivent réussir plusieurs vérifications de l'état consécutives avant qu'Elastic Beanstalk passe au lot suivant. Outre les vérifications de l'état ELB, qui ne vérifient que vos instances, les rapports améliorés sur l'état de santé surveillent les journaux d'application et l'état des autres ressources de votre environnement. Dans un environnement de serveur web à l'intégrité améliorée, toutes les instances doivent réussir 12 vérifications de l'état au cours des deux minutes (18 vérifications au cours des trois minutes pour les environnements de travail). Si une instance échoue à une vérification de l'état, le nombre se réinitialise.

Si un lot ne devient pas sain dans le délai de mise à jour propagée (la valeur par défaut est de 30 minutes), la mise à jour est annulée. L'expiration de la mise à jour propagée est une option de configuration qui est disponible dans l'espace de noms `aws:autoscaling:updatepolicy:rollingupdate`. Si votre application ne réussit pas les vérifications de l'état avec le statut `Ok` mais qu'elle est stable à un autre niveau, vous pouvez définir l'option `HealthCheckSuccessThreshold` dans l'espace de noms

`aws:elasticbeanstalk:healthreporting:system` afin de modifier le niveau auquel Elastic Beanstalk considère une instance comme étant saine.

Si le processus de mise à jour propagée échoue, Elastic Beanstalk commence une autre mise à jour propagée pour restaurer la configuration précédente. Une mise à jour propagée peut échouer en raison de vérifications de l'état ayant échoué, ou si le lancement de nouvelles instances vous conduit à dépasser les quotas de votre compte. Si vous atteignez un quota sur le nombre d' EC2 instances Amazon, par exemple, la mise à jour continue peut échouer lorsqu'elle tente de provisionner un lot de nouvelles instances. Dans ce cas, la restauration échoue également.

Si la restauration échoue, le processus de mise à jour est interrompu et votre environnement reste défaillant. Les lots non traités continuent à exécuter des instances avec l'ancienne configuration, tandis que les lots qui ont été traités correctement disposent de la nouvelle configuration. Pour réparer un environnement après une restauration qui a échoué, commencez par résoudre le problème sous-jacent qui a provoqué l'échec de la mise à jour, puis lancez une autre mise à jour de l'environnement.

Une autre méthode consiste à déployer la nouvelle version de votre application dans un environnement différent, puis à effectuer une permutation CNAME pour rediriger le trafic sans interruption. Pour plus d'informations, consultez Déploiements bleu/vert avec Elastic Beanstalk.

Mises à jour propagées et déploiements propagés

Les mises à jour continues se produisent lorsque vous modifiez les paramètres qui nécessitent le provisionnement de nouvelles EC2 instances Amazon pour votre environnement. Cela inclut des modifications apportées à la configuration du groupe Auto Scaling, telles que le type d'instance et des paramètres de paire de clés et les modifications apportées aux paramètres VPC. Dans une mise à jour propagée, chaque lot d'instances est résilié avant l'allocation d'un nouveau lot pour le remplacer.

Les déploiements propagés se produisent lorsque vous déployez votre application, et peuvent généralement être effectués sans remplacer d'instances dans votre environnement. Elastic Beanstalk met chaque lot hors service, déploie la nouvelle version de l'application, puis le remet en service.

L'exception à cela est si vous modifiez les paramètres qui ont besoin de remplacement d'instance en même temps vous déployez une nouvelle version de l'application. Par exemple, si vous modifiez les paramètres de nom de clé dans un fichier de configuration dans votre bundle source et que vous les déployez dans votre environnement, vous déclenchez une mise à jour propagée. Au lieu de déployer votre nouvelle version de l'application dans chaque lot d'instances existantes, un nouveau lot d'instances est fourni avec la nouvelle configuration. Dans ce cas, aucun déploiement distinct ne se produit, car les nouvelles instances sont installées avec la nouvelle version de l'application.

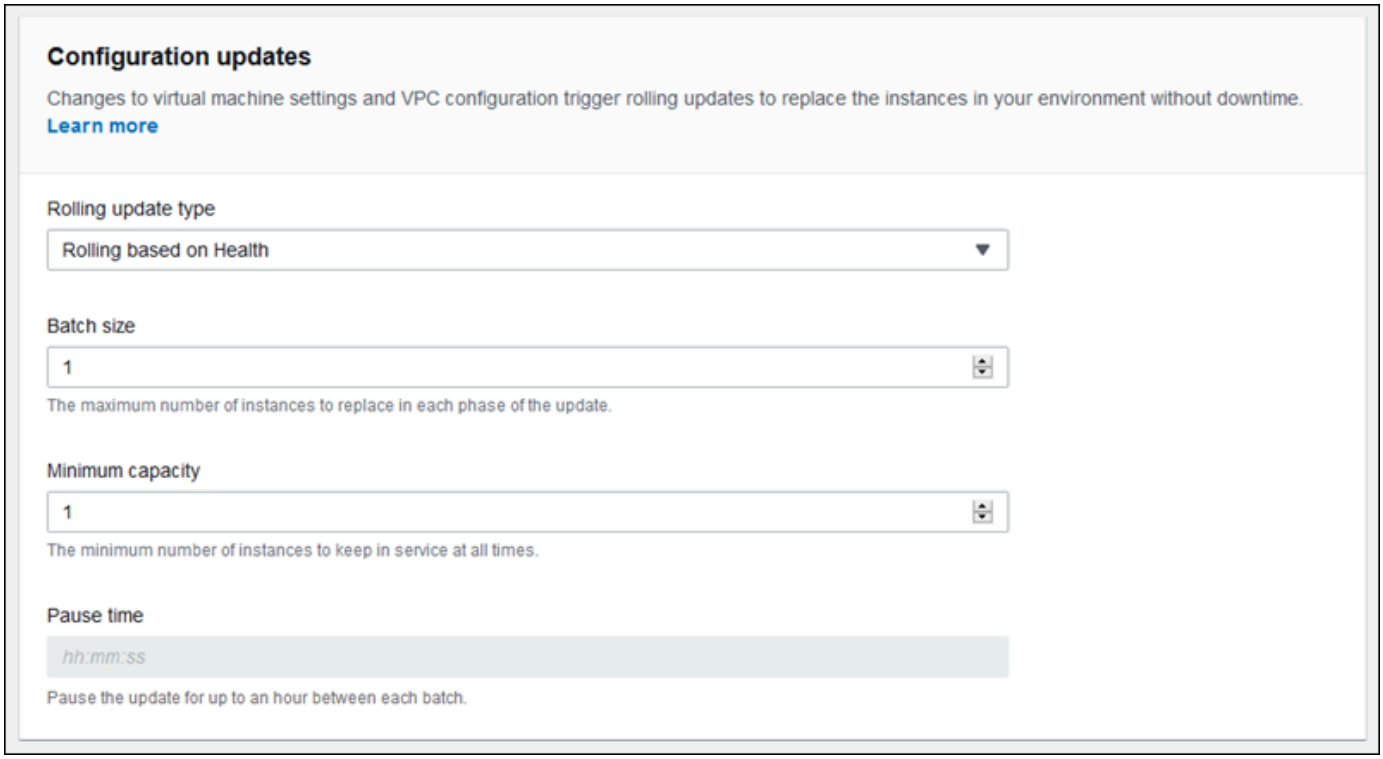
Chaque fois que de nouvelles instances sont déployées dans le cadre d'une mise à jour de l'environnement, une phase de déploiement est effectuée, au cours de laquelle le code source de votre application est déployé dans les nouvelles instances et tous les paramètres de configuration modifiant le système d'exploitation ou le logiciel sur les instances sont appliqués. Les paramètres de vérification de l'état du déploiement (Ignorer la vérification de l'état, Seuil de bonne santé et Délai de commande) s'appliquent également aux mises à jour propagées basées sur l'intégrité et aux mises à jour immuables lors de la phase de déploiement.

Configuration des mises à jour propagées

Vous pouvez activer et configurer les mises à jour propagées dans la console Elastic Beanstalk.

Pour activer des mises à jour propagées

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Rolling updates and deployments (Propagation des mises à jour et déploiements), choisissez Edit (Modifier).
5. Dans la section Mises à jour de la configuration, pour Propagation du type de mises à jour, sélectionnez l'une des options Propagation.



6. Choisissez Taille de lot, Capacité minimale et les paramètres Temps d'interruption.
7. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

La section Mises à jour de la configuration de la page Propagation des mises à jour et déploiements propose les options suivantes pour la propagation des mises à jour :

- Rolling update type (Propagation du type de mise à jour) – Elastic Beanstalk attend la fin de la mise à jour d'un lot d'instances avant de passer au lot suivant, afin de permettre à ces instances de finaliser l'action d'amorçage et de commencer à traiter le trafic. Sélectionnez l'une des options suivantes :
  - Rolling based on Health (Propagation en fonction de la santé) – Patientez jusqu'à ce que les instances dans le lot actuel soient saines avant de mettre des instances en service et de démarrer le lot suivant.
  - Rolling based on Time(Propagation en fonction de la durée) – Spécifiez un laps de temps d'attente entre le lancement de nouvelles instances et leur mise en service avant de démarrer le lot suivant.
  - Immutable (Immuable) – Appliquez la modification de la configuration à un nouveau groupe d'instances en effectuant une mise à jour immuable.

- Batch size (Taille de lot) – Le nombre d'instances à remplacer dans chaque lot, entre **1** et **10000**. Par défaut, cette valeur représente un tiers de la taille minimale du groupe Auto Scaling, arrondi au nombre entier.
- Minimum capacity (Capacité minimum) – Le nombre minimum d'instances à maintenir en cours d'exécution pendant que d'autres instances sont mises à jour, entre **0** et **9999**. La valeur par défaut est soit la taille minimale du groupe Auto Scaling ou un niveau en dessous de la taille maximale du groupe Auto Scaling, le chiffre le plus bas prévalant.
- Pause time (Temps d'interruption) (basé sur le temps uniquement) – La durée d'attente après qu'un lot est mis à jour avant de passer au lot suivant, pour permettre à votre application de commencer à recevoir du trafic. Entre 0 seconde et 1 heure.

L'espace de noms aws:autoscaling:updatepolicy:rollingupdate

Vous pouvez également utiliser les options de configuration dans l'espace de noms `aws:autoscaling:updatepolicy:rollingupdate` pour configurer des mises à jour propagées.

Utilisez l'option `RollingUpdateEnabled` pour activer les mises à jour propagées, et `RollingUpdateType` pour choisir le type de mise à jour. Les valeurs suivantes sont prises en charge pour `RollingUpdateType` :

- `Health` – Patientez jusqu'à ce que les instances dans le lot actuel soient saines avant de mettre des instances en service et de démarrer le lot suivant.
- `Time` – Spécifie un laps de temps d'attente entre le lancement de nouvelles instances et leur mise en service avant de démarrer le lot suivant.
- `Immutable` – Appliquez la modification de la configuration à un nouveau groupe d'instances en effectuant une mise à jour immuable.

Lorsque vous activez les mises à jours propagées, définissez les options `MaxBatchSize` et `MinInstancesInService` afin de configurer la taille de chaque lot. Pour les mises à jour propagées basées sur l'intégrité et basées sur le temps, vous pouvez également configurer un `PauseTime` et `Timeout`, respectivement.

Par exemple, afin de lancer jusqu'à cinq instances à la fois, tout en conservant au moins deux instances en service, puis attendre cinq minutes et 30 secondes entre les lots, spécifiez les options et les valeurs suivantes.

Example.ebextensions/timebased.config

```
option_settings:
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateEnabled: true
    MaxBatchSize: 5
    MinInstancesInService: 2
    RollingUpdateType: Time
    PauseTime: PT5M30S
```

Pour activer les mises à jour propagées basées sur l'intégrité, avec une expiration à 45 minutes pour chaque lot, spécifiez les valeurs et les options suivantes.

Example.ebextensions/healthbased.config

```
option_settings:
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateEnabled: true
    MaxBatchSize: 5
    MinInstancesInService: 2
    RollingUpdateType: Health
    Timeout: PT45M
```

`Timeout`et `PauseTime` les valeurs doivent être spécifiées dans ISO8601 duration :PT#H#M#S, où chaque # est le nombre d'heures, de minutes ou de secondes, respectivement.

L'interface de ligne de commande (CLI) EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Consultez Valeurs recommandées pour plus de détails.

Mises à jour immuables de l'environnement

Les mises à jour de l'environnement immuables sont une alternative aux mises à jour propagées. Les mises à jour de l'environnement immuables garantissent que les modifications de configuration qui nécessitent le remplacement d'instances sont appliquées efficacement et en toute sécurité. En cas de défaillance d'une mise à jour de l'environnement immuable, le processus de restauration nécessite uniquement l'arrêt d'un groupe Auto Scaling. Un échec de mise à jour propagée, en revanche, nécessite de procéder à une mise à jour propagée supplémentaire pour restaurer les modifications.

Pour effectuer une mise à jour de l'environnement immuable, Elastic Beanstalk crée un second groupe Auto Scaling temporaire derrière l'équilibreur de charge de votre environnement pour contenir les nouvelles instances. Tout d'abord, Elastic Beanstalk lance une instance unique avec la nouvelle configuration dans le nouveau groupe. Cette instance traite le trafic en même temps que toutes les instances dans le groupe Auto Scaling d'origine qui exécutent la configuration précédente.

Lorsque la première instance passe les vérifications de l'état, Elastic Beanstalk lance des instances supplémentaires avec la nouvelle configuration, correspondant au nombre d'instances en cours d'exécution dans le groupe Auto Scaling d'origine. Lorsque toutes les nouvelles instances réussissent les vérifications de l'état, Elastic Beanstalk les transfère au groupe Auto Scaling d'origine et résilie le groupe Auto Scaling temporaire et les anciennes instances.

> **Note**
>
> Pendant une mise à jour d'environnement immuable, la capacité de votre environnement double pendant une brève durée lorsque les instances dans le nouveau groupe Auto Scaling commencent à traiter les demandes et avant que les instances du groupe Auto Scaling d'origine soient résiliées. Si votre environnement a de nombreuses instances, ou des instances avec un faible quota d'instances à la demande, assurez-vous d'avoir suffisamment de capacité pour effectuer une mise à jour de l'environnement immuable. Si vous êtes proche du quota, pensez à utiliser des mises à jour propagées à la place.

Les mises à jour immuables nécessitent des rapports sur l'état de santé améliorés afin d'évaluer l'intégrité de votre environnement pendant la mise à jour. Les rapports sur l'état de santé améliorés associent des vérifications de l'état standard de l'équilibreur de charge avec la surveillance de l'instance afin de garantir que les instances en cours d'exécution avec la nouvelle configuration traitent les demandes avec succès.

Vous pouvez également utiliser des mises à jour immuables pour déployer de nouvelles versions de votre application, comme une alternative aux déploiement propagés. Lorsque vous configurez Elastic Beanstalk pour utiliser des mises à jour immuables pour des déploiements d'applications, il remplace toutes les instances de votre environnement chaque fois que vous déployez une nouvelle version de votre application. Si un déploiement d'application immuable échoue, Elastic Beanstalk rétablit immédiatement les modifications en résiliant le nouveau groupe Auto Scaling. Cela peut empêcher des déploiements de flottes partiels qui peuvent se produire lorsqu'un déploiement propagé échoue après que certains lots sont déjà terminés.

> **Warning**
>
> Certaines politiques remplacent toutes les instances pendant le déploiement ou la mise à jour. Cela entraîne la perte de tous EC2 les soldes cumulés d'Amazon. Une telle situation se produit dans les cas suivants :
>
> - Mises à jour de la plate-forme gérée avec le remplacement d'instance activé
> - Mises à jour immuables
> - Déploiements avec mises à jour immuables ou fractionnement du trafic activé

Si une mise à jour immuable échoue, les nouvelles instances téléchargent les journaux groupés vers Amazon S3 avant qu'Elastic Beanstalk ne les résilie. Elastic Beanstalk laisse les journaux d'une mise à jour immuable ayant échoué dans Amazon S3 pendant une heure avant de les supprimer, au lieu des 15 minutes standard pour les journaux groupés et de processus.

> **Note**
>
> Si vous utilisez des mises à jour immuables pour des déploiements de version d'application, mais pas pour la configuration, vous pouvez rencontrer une erreur si vous tentez de déployer une version de l'application qui contient des modifications de configuration qui déclencheraient normalement une mise à jour propagée (par exemple, des configurations modifiant le type d'instance). Pour éviter ce problème, modifiez la configuration dans une mise à jour séparée, ou configurez les mises à jour immuables à la fois pour les changements de configuration et pour les déploiements.

Vous ne pouvez pas effectuer une mise à jour immuable de concert avec les changements de configuration de ressource. Par exemple, vous ne pouvez pas modifier les paramètres qui ont besoin d'un remplacement d'instance tout en mettant également à jour d'autres paramètres, ni exécuter un déploiement immuable avec des fichiers de configuration qui modifient les paramètres de configuration ou les ressources supplémentaires dans votre code source. Si vous essayez de modifier les paramètres de ressource (par exemple, les paramètres d'équilibreur de charge) et de procéder simultanément à une mise à jour immuable, Elastic Beanstalk renvoie une erreur.

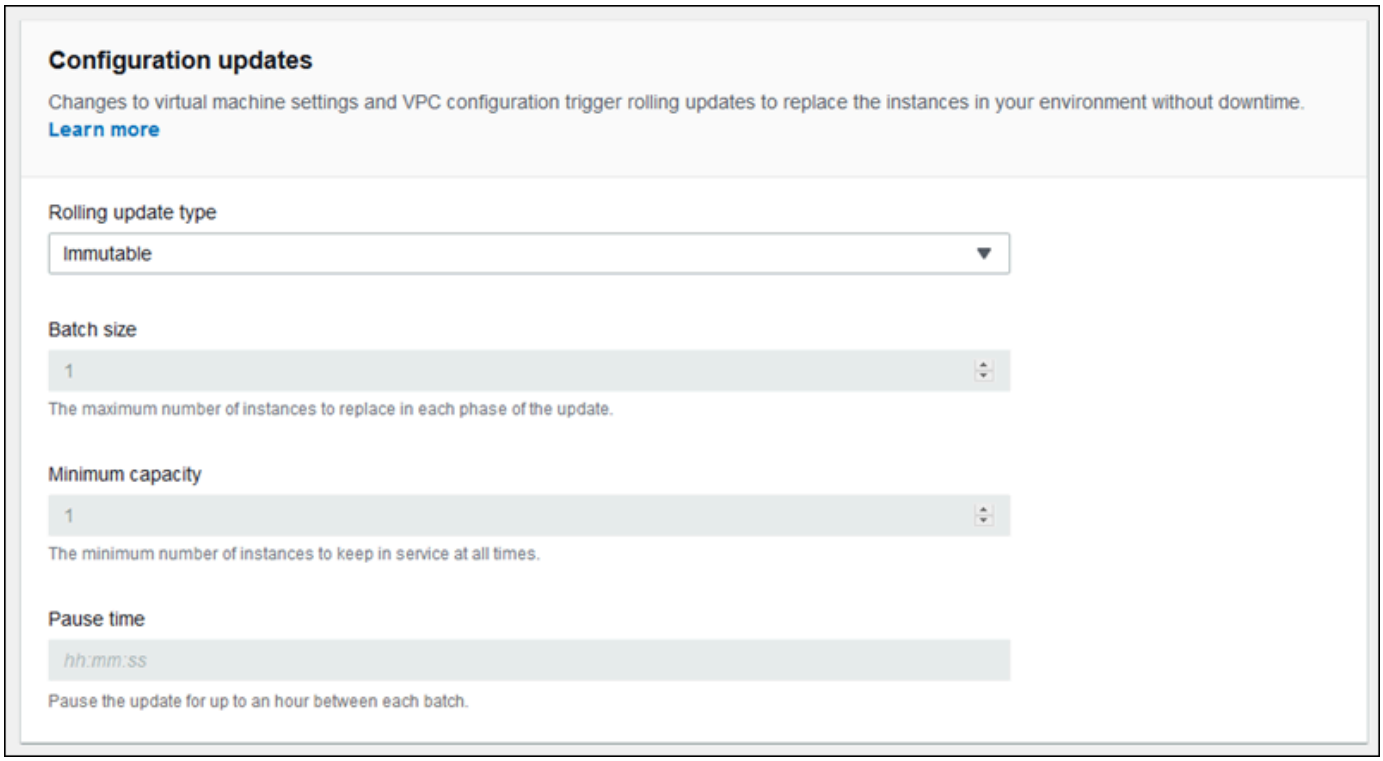
Si vos modifications de configuration de ressource ne dépendent pas de la modification de votre code source ou de la configuration d'instance, exécutez-les dans deux mises à jour. Si elles sont dépendantes, effectuez plutôt un déploiement bleu/vert.

Configuration de mises à jour immuables

Vous pouvez activer et configurer des mises à jour immuables dans la console Elastic Beanstalk.

Pour activer des mises à jour immuables (console)

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Rolling updates and deployments (Propagation des mises à jour et déploiements), choisissez Edit (Modifier).
5. Dans la section Mises à jour de la configuration, définissez Rolling update type (Type des mises à jour propagées) sur Immuable.



6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

L'espace de noms aws:autoscaling:updatepolicy:rollingupdate

Vous pouvez également utiliser les options dans l'espace de noms `aws:autoscaling:updatepolicy:rollingupdate` pour configurer des mises à jour immuables. L'exemple suivant de fichier de configuration active les mises à jour immuables pour les changements de configuration.

Example.ebextensions/immutable-updates.config

```
option_settings:
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateType: Immutable
```

L'exemple suivant active les mises à jour immuables pour les changements de configuration et les déploiements.

Example.ebextensions/immutable-all.config

```
option_settings:
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateType: Immutable
  aws:elasticbeanstalk:command:
    DeploymentPolicy: Immutable
```

L'interface de ligne de commande (CLI) EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Consultez Valeurs recommandées pour plus de détails.

Mise à jour de la version de la plateforme de votre environnement Elastic Beanstalk

Elastic Beanstalk publie régulièrement de nouvelles versions de plateforme pour mettre à jour toutes les plateformes basées sur Linux et sur Windows Server. Les nouvelles versions de plateforme fournissent des mises à jour des composants logiciels existants et prennent en charge de nouvelles fonctionnalités ainsi que des options de configuration. Pour en savoir plus sur les plateformes et les versions de plateforme, veuillez consulter Glossaire des plateformes Elastic Beanstalk.

Vous pouvez utiliser la console Elastic Beanstalk ou l'interface de ligne de commande EB pour mettre à jour la version de la plateforme de votre environnement. En fonction de la version de plateforme que vous souhaitez mettre à jour, Elastic Beanstalk recommande l'une des deux méthodes pour effectuer des mises à jour de la plateforme.

- Méthode 1 – Mettre à jour la version de la plateforme de votre environnement. Nous vous recommandons d'utiliser cette méthode lorsque vous mettez à jour vers la dernière version de la plateforme dans une branche de plateforme : avec le même environnement d'exécution, le même serveur web, le même serveur d'application et le même système d'exploitation, et sans modifier la version de la plateforme principale. Il s'agit de la mise à jour de plateforme la plus courante et la plus régulière.
- Méthode 2 — Effectuer un Blue/Green déploiement. Nous vous recommandons d'utiliser cette méthode lorsque vous effectuez une mise à jour vers une version de plateforme dans une autre branche de plateforme : avec un environnement d'exécution, un serveur web, un serveur d'applications ou un système d'exploitation différents, ou vers une autre version majeure de plateforme. Il s'agit d'une bonne approche lorsque vous voulez tirer parti des nouvelles fonctionnalités d'exécution ou des dernières fonctionnalités Elastic Beanstalk, ou encore lorsque vous voulez quitter une branche de plateforme obsolète ou hors service.

  La migration à partir d'une ancienne version de plate-forme nécessite un blue/green déploiement, car ces versions de plate-forme sont incompatibles avec les versions actuellement prises en charge.

  La migration d'une application Linux vers Amazon Linux 2 nécessite un blue/green déploiement, car les versions de la plateforme Amazon Linux 2 sont incompatibles avec les versions précédentes de la plate-forme Amazon Linux AMI.

Pour plus d'informations sur le choix de la meilleure méthode de mise à jour de la plateforme, développez la section pour la plateforme de votre environnement.

Docker

Utilisez la méthode 1 pour effectuer des mises à jour de la plateforme.

Docker multi-conteneurs

Utilisez la méthode 1 pour effectuer des mises à jour de la plateforme.

Docker préconfiguré

Considérons les cas suivants :

- Si vous migrez votre application vers une autre plateforme, par exemple de Go 1.4 (Docker) vers Go 1.11 ou de Python 3.4 (Docker) vers Python 3.6, utilisez la méthode 2.
- Si vous migrez votre application vers une autre version de conteneur Docker, par exemple de Glassfish 4.1 (Docker) vers Glassfish 5.0 (Docker), utilisez la méthode 2.
- Si vous mettez à jour vers une version plus récente de la plateforme sans aucune modification de version majeure ou de conteneur, utilisez la méthode 1.

Go

Utilisez la méthode 1 pour effectuer des mises à jour de la plateforme.

Java SE

Considérons les cas suivants :

- Si vous migrez votre application vers une autre version d'exécution Java, par exemple de Java 7 vers Java 8, utilisez la méthode 2.
- Si vous mettez à jour vers une version plus récente de la plateforme, sans modification de la version d'exécution, utilisez la méthode 1.

Java avec Tomcat

Considérons les cas suivants :

- Si vous migrez votre application vers une autre version d'environnement d'exécution Java ou version de serveur d'application Tomcat, par exemple de Java 7 avec Tomcat 7 vers Java 8 avec Tomcat 8.5, utilisez la méthode 2.
- Si vous migrez votre application sur les versions majeures de Java avec la plateforme Tomcat (v1.x.x, v2.x.x et v3.x.x), utilisez la méthode 2.
- Si vous mettez à jour vers une version plus récente de la plateforme, sans modifier la version d'exécution, la version du serveur d'application ou la version majeure, utilisez la méthode 1.

.NET sur serveur Windows avec IIS

Considérons les cas suivants :

- Si vous migrez votre application vers une autre version du système d'exploitation Windows, par exemple de Windows Server 2008 R2 vers Windows Server 2016, utilisez la méthode 2.
- Si vous migrez votre application à travers les principales versions de plateforme Windows Server, veuillez consulter Migration à partir de versions majeures antérieures de la plateforme Windows Server, et utilisez la méthode 2.
- Si votre application est en cours d'exécution sur une plateforme Windows Server V2.x.x et que vous mettez à jour vers une version plus récente de la plateforme, utilisez la méthode 1.

> **Note**
>
> Les versions de plateforme Windows Server antérieures à la version v2 ne sont pas sémantiquement versionnées. Vous ne pouvez lancer que la dernière version de ces versions de plateforme majeures Windows Server et vous ne pouvez pas restaurer après une mise à niveau.

Node.js

Utilisez la méthode 2 pour effectuer des mises à jour de la plateforme.

PHP

Considérons les cas suivants :

- Si vous migrez votre application vers une autre version d'exécution PHP, par exemple de PHP 5.6 vers PHP 7.2, utilisez la méthode 2.
- Si vous migrez votre application sur les versions majeures de la plateforme PHP (v1.x.x et v2.x.x), utilisez la méthode 2.
- Si vous mettez à jour vers une version plus récente de la plateforme, sans modifier la version d'exécution ou la version majeure, utilisez la méthode 1.

Python

Considérons les cas suivants :

- Si vous migrez votre application vers un autre environnement d'exécution Python, par exemple de Python 2.7 vers Python 3.6, utilisez la méthode 2.
- Si vous migrez votre application sur les versions majeures de la plateforme Python (v1.x.x et v2.x.x), utilisez la méthode 2.
- Si vous mettez à jour vers une version plus récente de la plateforme, sans modifier la version d'exécution ou la version majeure, utilisez la méthode 1.

Ruby

Considérons les cas suivants :

- Si vous migrez votre application vers une autre version d'exécution de Ruby ou une version de serveur d'application, par exemple de Ruby 2.3 avec Puma vers Ruby 2.6 avec Puma, utilisez la méthode 2.
- Si vous migrez votre application sur les versions majeures de la plateforme Ruby (v1.x.x et v2.x.x), utilisez la méthode 2.
- Si vous mettez à jour vers une version plus récente de la plateforme, sans modifier la version d'exécution, la version du serveur d'application ou la version majeure, utilisez la méthode 1.

Méthode 1 – Mettre à jour la version de la plateforme de votre environnement

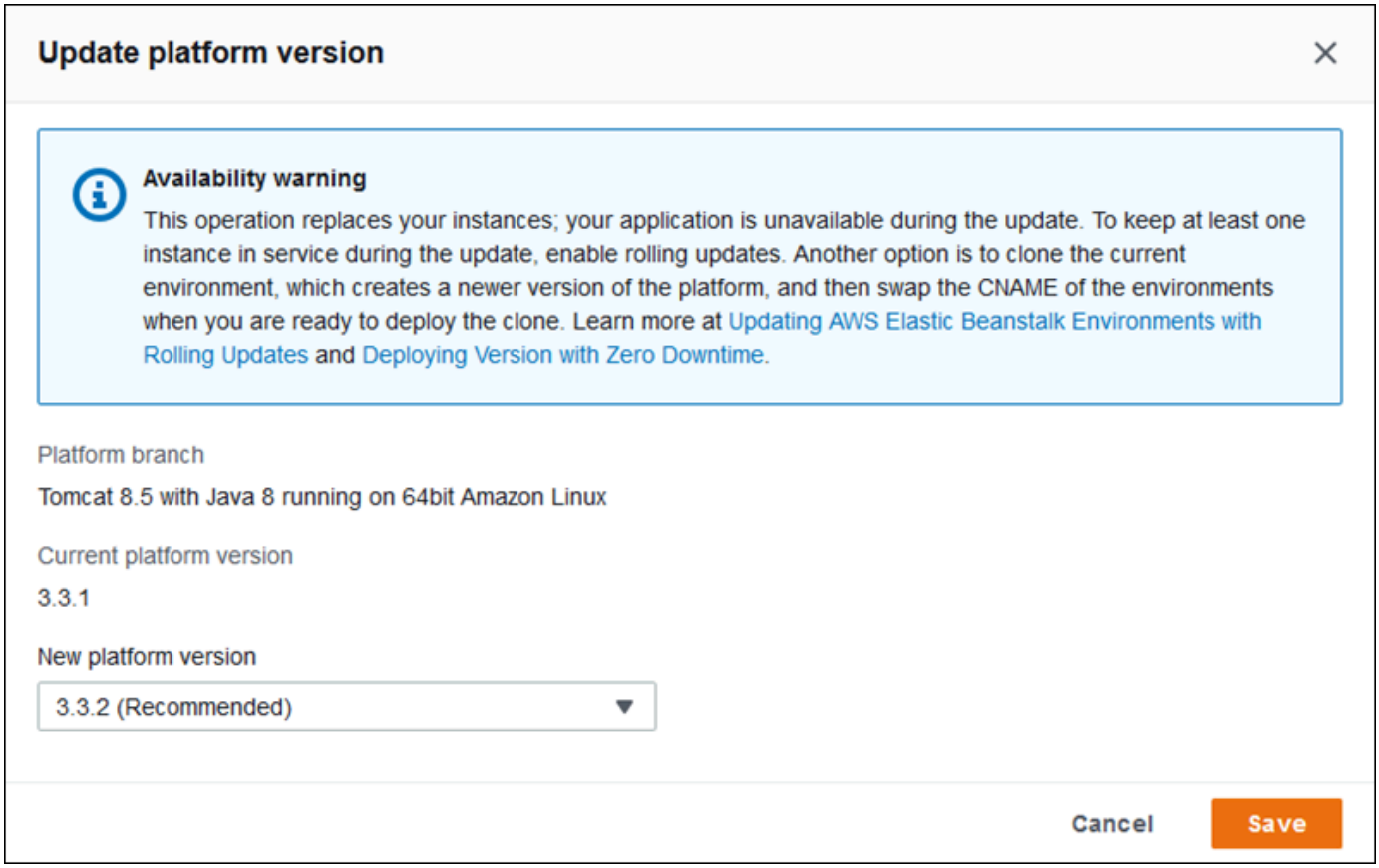
Utilisez cette méthode pour mettre à jour vers la dernière branche de plateforme de votre environnement. Si vous avez préalablement créé un environnement à l'aide d'une version de plateforme antérieure ou mis à niveau votre environnement à partir d'une version plus ancienne, vous pouvez également utiliser cette méthode pour revenir à une version de plateforme précédente, à condition qu'elle se trouve dans la même branche de plateforme.

Pour mettre à jour la version de la plateforme de votre environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans la page de présentation de l'environnement, sous Platform (Plateforme), choisissez Change (Changer).

4. Dans la boîte de dialogue Update platform version (Mettre à jour la version de la plateforme) sélectionnez une version de la plate-forme. La version de la plateforme la plus récente (recommandée) de la branche est sélectionnée automatiquement. Vous pouvez mettre à jour vers n'importe quelle version que vous avez utilisée dans le passé.



5. Choisissez Enregistrer.

Pour davantage simplifier les mises à jour des plateformes, Elastic Beanstalk peut les gérer pour vous. Vous pouvez configurer votre environnement pour appliquer automatiquement des mises à jour de versions mineures et de correctifs pendant un créneau de maintenance hebdomadaire configurable. Elastic Beanstalk applique des mises à jour gérées sans interruption ni réduction de capacité et annule la mise à jour immédiatement si les instances exécutant votre application sur la nouvelle version échouent aux vérifications de l'état. Pour en savoir plus, consultez Mises à jour gérées de la plateforme.

Méthode 2 — Effectuer un Blue/Green déploiement

Utilisez cette méthode pour effectuer une mise à jour vers une autre branche de plateforme : avec un environnement d'exécution, un serveur web, un serveur d'applications ou un système d'exploitation différents, ou vers une autre version principale de plateforme. Cela est généralement nécessaire lorsque vous souhaitez tirer parti des nouvelles fonctionnalités d'exécution ou des dernières fonctionnalités Elastic Beanstalk. Cela est également requis lorsque vous migrez hors d'une branche de plateforme obsolète ou hors service.

Lorsque vous migrez sur plusieurs versions de plate-forme majeures ou vers des versions de plate-forme avec des mises à jour de composants principaux composant, il y a plus de chances que votre application ou certains aspects de celle-ci peuvent ne pas fonctionner comme prévu sur la nouvelle version de la plateforme, et peuvent nécessiter des changements.

Avant de réaliser la migration, mettez à jour votre machine de développement local vers les dernières versions d'exécution et d'autres composants de la plateforme que vous prévoyez de migrer. Vérifiez que votre application fonctionne comme prévu et apportez toutes les modifications et les correctifs de code nécessaire. Ensuite, utilisez la procédure de bonnes pratiques suivantes pour migrer votre environnement en toute sécurité vers la nouvelle version de la plateforme.

Pour migrer votre environnement vers une version de plateforme avec des mises à jour majeures

1. Créez un environnement en utilisant la nouvelle version de plateforme, puis déployez-y votre code d'application. Le nouvel environnement doit être dans l'application Elastic Beanstalk qui contient l'environnement que vous migrez. N'arrêtez pas encore l'environnement existant.
2. Utilisez le nouvel environnement pour migrer votre code d'application. En particulier :

   - Trouvez et corrigez les problèmes de compatibilité de l'application que vous n’avez pas pu détecter au cours de la phase de développement.
   - Assurez-vous que toutes les personnalisations que votre application effectue en utilisant les fichiers de configuration fonctionnent correctement dans le nouvel environnement. Celles-ci peuvent inclure les paramètres d'options, les packages installés supplémentaires, les stratégies de sécurité personnalisées, et les fichiers de configuration ou de script installés sur les instances de l'environnement.
   - Si votre application utilise une Amazon Machine Image (AMI) personnalisée, créez une AMI personnalisée basée sur l'AMI de la nouvelle version de la plateforme. Pour en savoir plus, consultez la section Utilisation d'une image machine Amazon (AMI) personnalisée dans votre environnement Elastic Beanstalk. Plus précisément, cette action est obligatoire si votre application utilise la plateforme Windows Server avec une AMI personnalisée, et que vous migrez vers une version de la plateforme Windows Server V2. Dans ce cas, veuillez consulter également Migration à partir de versions majeures antérieures de la plateforme Windows Server.

   Itérez les tests et déployer vos correctifs jusqu'à ce que vous soyez satisfait de l'application sur le nouvel environnement.

3. Activez le nouvel environnement en tant qu'environnement de production en échangeant son CNAME avec le CNAME de l'environnement de production existant. Pour plus d'informations, consultez Déploiements bleu/vert avec Elastic Beanstalk.
4. Lorsque vous êtes satisfait de l'état de votre nouvel environnement de production, arrêtez l'ancien environnement. Pour en savoir plus, consultez Arrêt d'un environnement Elastic Beanstalk.

Mises à jour gérées de la plateforme

AWS Elastic Beanstalk publie régulièrement des mises à jour de plate-forme pour fournir des correctifs, des mises à jour logicielles et de nouvelles fonctionnalités. Grâce aux mises à jour gérées de la plateforme, vous pouvez configurer votre environnement pour effectuer automatiquement les mises à niveau vers la dernière version d'une plateforme au cours d'un créneau de maintenance planifié. Votre application reste en service pendant le processus de mise à jour, sans réduction de capacité. Les mises à jour gérées sont disponibles sur les environnements à l'instance unique et avec équilibrage de charge.

> **Note**
>
> Cette fonction n'est pas disponible dans les versions de la plateforme Windows Server antérieures à la version 2 (v2).

Vous pouvez configurer votre environnement pour appliquer automatiquement les mises à jour de version corrective ou les mises à jour des deux versions : mineure et corrective. Les mises à jour gérées de la plateforme ne prennent pas en charge les mises à jour entre les branches de la plateforme (mises à jour vers différentes versions majeures de composants de la plateforme tels que le système d'exploitation, l'exécution ou les composants Elastic Beanstalk), car elles peuvent introduire des modifications non rétrocompatibles.

Vous pouvez également configurer Elastic Beanstalk pour remplacer toutes les instances de votre environnement pendant la fenêtre de maintenance, même si aucune mise à jour de la plateforme n'est disponible. Le remplacement de toutes les instances de votre environnement est utile si votre application est confrontée à des bogues ou à des problèmes de mémoire lorsqu'elle est exécutée pendant une longue période.

Sur les environnements créés le 25 novembre 2019 ou ultérieurement à l'aide de la console Elastic Beanstalk, les mises à jour gérées sont activées par défaut dans la mesure du possible. Les

mises à jour gérées nécessitent que l'état amélioré soit activé. L'état amélioré est activé par défaut lorsque vous sélectionnez un des paramètres prédéfinis de configuration, et désactivé lorsque vous sélectionnez Configuration personnalisée. La console ne peut pas activer les mises à jour gérées pour les anciennes versions de plate-forme qui ne prennent pas en charge l'état amélioré, ou lorsque l'état amélioré est désactivé. Lorsque la console active les mises à jour gérées pour un nouvel environnement, la fenêtre de mise à jour hebdomadaire est définie sur un jour aléatoire de la semaine à un moment aléatoire. Niveau de mise à jour est défini sur Mineures et correctifs et Remplacement d'instance est désactivé. Vous pouvez désactiver ou reconfigurer les mises à jour gérées avant l'étape finale de création de l'environnement.

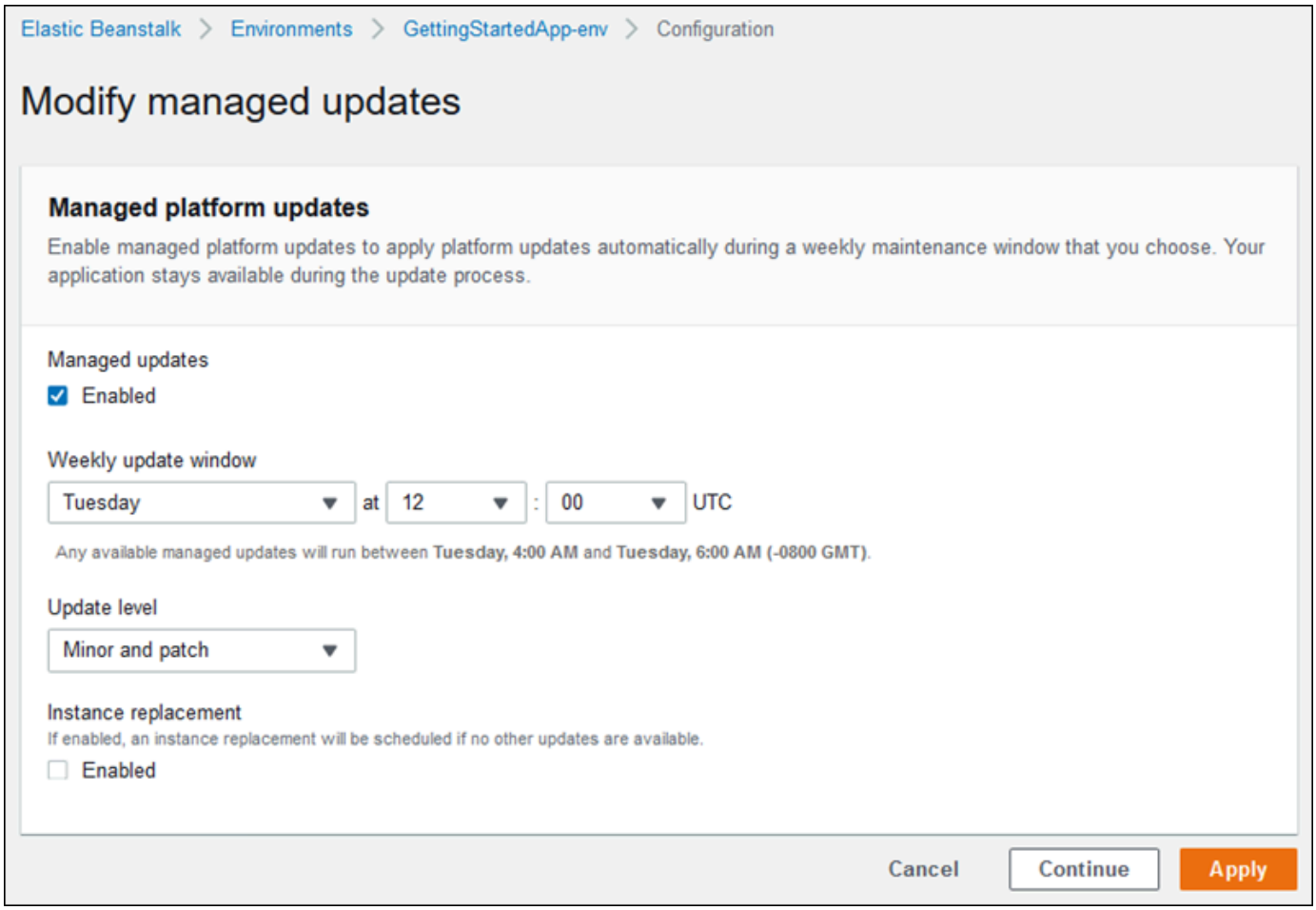
Pour un environnement existant, utilisez la console Elastic Beanstalk à tout moment pour configurer les mises à jour des plateformes gérées.

> **Important**
>
> Un grand nombre d'environnements Beanstalk dans un compte AWS peut présenter un risque de problèmes de limitation lors des mises à jour gérées. Un grand nombre correspond à une quantité relative qui dépend étroitement de la manière dont vous planifiez les mises à jour gérées pour vos environnements. Plus de 200 environnements dans un compte planifié étroitement peuvent causer des problèmes de limitation, bien qu'un nombre inférieur puisse également poser problème.
>
> Pour équilibrer la charge de ressources pour les mises à jour gérées, nous vous conseillons de répartir les fenêtres de maintenance planifiées pour les environnements présents dans un même compte.
>
> Pensez également à une stratégie à plusieurs comptes. Pour plus d'informations, consultez la section Organisation de votre AWS environnement à l'aide de plusieurs comptes sur le site AWS Web des livres blancs et des guides.

Pour configurer les mises à jour des plates-formes gérées

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie Managed updates (Mises à jour gérées), choisissez Edit (Modifier).

5. Désactivez ou activez Mises à jour gérées.
6. Si les mises à jour gérées sont activées, sélectionnez une fenêtre de maintenance, puis un niveau de mise à jour.
7. (Facultatif) Sélectionnez Remplacement de l'instance pour activer le remplacement d'instance hebdomadaire.



8. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Les mises à jour gérées de la plateforme utilisent les rapports sur l'état de santé améliorés afin de déterminer si l'état de votre application est suffisamment satisfaisant pour estimer que la mise à jour de la plateforme a réussi. Pour obtenir des instructions, consultez Activation des rapports améliorés sur l'état Elastic Beanstalk.

Sections

- Autorisations requises pour effectuer des mises à jour gérées de la plateforme
- Fenêtre de maintenance des mises à jour gérées

- Mises à jour des versions mineures et correctives
- Mises à jour immuables de l'environnement
- Gestion des mises à jour gérées
- Espaces de noms avec options d'action gérée

Autorisations requises pour effectuer des mises à jour gérées de la plateforme

Elastic Beanstalk a besoin d'autorisations pour lancer une mise à jour de la plateforme en votre nom. Pour obtenir ces autorisations, Elastic Beanstalk assume le rôle de service de mises à jour gérées. Lorsque vous utilisez le rôle de service par défaut pour votre environnement, la console Elastic Beanstalk l'utilise également comme rôle de service de mises à jour gérées. La console affecte la politique gérée `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy` à votre fonction du service. Cette politique dispose de toutes les autorisations dont Elastic Beanstalk a besoin pour effectuer des mises à jour de la plateforme gérée.

Pour plus d'informations sur les autres méthodes de définition de la fonction du service de mises à jour gérées, consultez the section called “Rôles du service”.

> **Note**
>
> Si vous utilisez des fichiers de configuration pour étendre votre environnement afin d'y inclure des ressources supplémentaires, vous pouvez avoir besoin d'ajouter des autorisations au rôle de service des mises à jour gérées de votre environnement. En général, vous devez ajouter des autorisations lorsque vous faites référence au nom de ces ressources dans d'autres sections ou fichiers.

En cas d'échec d'une mise à jour, vous pouvez trouver la raison de l'échec sur la page des mises à jour gérées.

Fenêtre de maintenance des mises à jour gérées

Lors de AWS la publication d'une nouvelle version de la plateforme de votre environnement, Elastic Beanstalk planifie une mise à jour gérée de la plateforme lors de la prochaine période de maintenance hebdomadaire. Une fenêtre de maintenance dure deux heures. Elastic Beanstalk démarre une mise à jour planifiée pendant la fenêtre de maintenance. La mise à jour ne se termine avant la fin de la fenêtre.

> **Note**
>
> Dans la plupart des cas, Elastic Beanstalk planifie la mise à jour gérée de telle sorte qu'elle se produise pendant la fenêtre de maintenance hebdomadaire suivante. Le système considère différents aspects de sécurité et de disponibilité des services lors de la planification des mises à jour gérées. Dans de rares cas, une mise à jour peut ne pas être planifiée pour la première fenêtre de maintenance à venir. Si cela se produit, le système tente à nouveau durant la fenêtre de maintenance suivante. Pour procéder manuellement à une mise à jour gérée, choisissez Appliquer maintenant comme expliqué dans la section Gestion des mises à jour gérées sur cette page.

Mises à jour des versions mineures et correctives

Vous pouvez autoriser les mises à jour gérées de la plateforme à appliquer des mises à jour de version corrective uniquement, ou des mises à jour des deux versions : mineure et corrective. Les mises à jour de version corrective incluent des correctifs de bogues et des améliorations des performances. Elles peuvent aussi comporter des modifications mineures de configuration pour les logiciels sur les instances, les scripts et les options de configuration. Les mises à jour mineures de version prennent en charge les nouvelles fonctions d'Elastic Beanstalk. Dans le cadre des mises à jour gérées de la plateforme, vous ne pouvez pas appliquer de mises à jour de version majeure, lesquelles peuvent entraîner des modifications non rétrocompatibles.

Dans le numéro de version d'une plateforme, le deuxième numéro correspond à la mise à jour de version mineure, et le troisième numéro correspond à la version corrective. Par exemple, une plateforme version 2.0.7 a une version mineure de 0 et une version corrective de 7.

Mises à jour immuables de l'environnement

Les mises à jour gérées de la plateforme exécutent des mises à jour immuables de l'environnement pour mettre à niveau votre environnement vers une nouvelle version de la plateforme. Les mises à jour immuables mettent à jour votre environnement sans suspendre aucune instance ni modifier votre environnement, avant de vérifier que les instances qui exécutent la nouvelle version réussissent les vérifications de l'état.

Dans une mise à jour immuable, Elastic Beanstalk déploie autant d'instances qu'il y en a en cours d'exécution avec la nouvelle version de la plateforme. Les nouvelles instances commencent à accepter des demandes en même temps que celles qui exécutent l'ancienne version. Si le nouvel

ensemble d'instances réussit toutes les vérifications de l'état, Elastic Beanstalk résilie l'ancien ensemble d'instances, en ne laissant que les instances associées à la nouvelle version.

Les mises à jour gérées de la plateforme effectuent toujours des mises à jour immuables, même lorsque vous les appliquez en dehors de la fenêtre de maintenance. Si vous modifiez la version de la plateforme dans le tableau de bord, Elastic Beanstalk applique la politique de mise à jour que vous avez choisie pour les mises à jour de configuration.

> **Warning**
>
> Certaines politiques remplacent toutes les instances pendant le déploiement ou la mise à jour. Cela entraîne la perte de tous les équilibres de rafale Amazon EC2 cumulés. Une telle situation se produit dans les cas suivants :
>
> - Mises à jour de la plate-forme gérée avec le remplacement d'instance activé
> - Mises à jour immuables
> - Déploiements avec mises à jour immuables ou fractionnement du trafic activé

Gestion des mises à jour gérées

La console Elastic Beanstalk présente des informations détaillées sur les mises à jour gérées sur la page Présentation des mises à jour gérées.

Pour afficher les informations relatives aux mises à jour gérées (console)

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Managed updates (Mises à jour gérées).

La section Présentation des mises à jour gérées fournit des informations sur les mises à jour gérées planifiées et en attente. La section Historique répertorie les mises à jour ayant réussi et les tentatives ayant échoué.

Vous pouvez choisir d'appliquer une mise à jour planifiée immédiatement au lieu d'attendre la fenêtre de maintenance.

Pour appliquer une mise à jour gérée de la plateforme immédiatement (console)

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Managed updates (Mises à jour gérées).
4. Choisissez Appliquer maintenant.
5. Vérifiez les détails de la mise à jour, puis choisissez Apply (Appliquer).

Lorsque vous appliquez une mise à jour gérée de la plateforme en dehors de la fenêtre de maintenance, Elastic Beanstalk effectue une mise à jour immuable. Si vous mettez à jour la plateforme de l'environnement dans le tableau de bord ou via un autre client, Elastic Beanstalk utilise le type de mise à jour que vous avez sélectionné pour les changements de configuration.

Si vous n'avez planifié aucune mise à jour gérée, il est possible que votre environnement exécute déjà la dernière version. Voici d'autres raisons possibles pour lesquelles aucune mise à jour n'est planifiée :

- Une mise à jour de version mineure est disponible, mais votre environnement est configuré pour n'appliquer automatiquement que les mises à jour correctives de version.
- Votre environnement n'a pas été analysé depuis la publication de la mise à jour. Elastic Beanstalk vérifie généralement les mises à jour toutes les heures.
- Une mise à jour est en attente ou déjà en cours.

Lorsque la fenêtre de maintenance démarre ou que vous choisissez Appliquer maintenant, les mises à jour planifiées passent à l'état « en attente » avant l'exécution.

Espaces de noms avec options d'action gérée

Vous pouvez utiliser les options de configuration des espaces de noms `aws:elasticbeanstalk:managedactions` et `aws:elasticbeanstalk:managedactions:platformupdate` pour activer et configurer les mises à jour gérées de la plateforme.

L'option `ManagedActionsEnabled` active les mises à jour gérées de la plateforme. Définissez cette option sur `true` pour activer les mises à jour gérées de la plateforme, et utilisez les autres options pour configurer le comportement de mise à jour.

`PreferredStartTime`À utiliser pour configurer le début de la fenêtre de maintenance hebdomadaire *`minute`* au format *`day hour`* : :.

Définissez `UpdateLevel` sur `minor` ou `patch` afin d'appliquer respectivement des mises à jour de version mineure et de version corrective, ou uniquement des mises à jour de version corrective.

Lorsque les mises à jour gérées de la plateforme sont activées, vous pouvez activer le remplacement d'instance en définissant l'option `InstanceRefreshEnabled` sur `true`. Lorsque ce paramètre est activé, Elastic Beanstalk exécute une mise à jour immuable de l'environnement chaque semaine, qu'une nouvelle version de la plateforme soit disponible ou non.

L'exemple suivant de fichier de configuration active les mises à jour gérées de la plateforme pour les mises à jour correctives de version, avec une fenêtre de maintenance commençant à 9 h UTC chaque mardi.

Example extensions .eb/ .config managed-platform-update

```
option_settings:
  aws:elasticbeanstalk:managedactions:
    ManagedActionsEnabled: true
    PreferredStartTime: "Tue:09:00"
  aws:elasticbeanstalk:managedactions:platformupdate:
    UpdateLevel: patch
    InstanceRefreshEnabled: true
```

Migration de votre application depuis une version de plateforme héritée

Si vous avez déployé une application Elastic Beanstalk qui utilise une version de plateforme héritée, vous devez migrer votre application vers un nouvel environnement utilisant une version de plateforme non héritée afin d'accéder aux nouvelles fonctions. Si vous n'êtes pas sûr d'exécuter votre application à l'aide d'une version de plate-forme héritée, vous pouvez vérifier cette information dans la console Elastic Beanstalk. Pour obtenir des instructions, consultez Pour vérifier si vous utilisez une version de plate-forme héritée.

Quelles nouvelles fonctionnalités ne figurent pas dans les versions de plate-forme héritée ?

Les plateformes héritées ne prennent pas en charge les fonctions suivantes :

- Fichiers de configuration, comme décrit dans la rubrique Personnalisation d'environnement avancée avec fichiers de configuration (`.ebextensions`)
- Vérifications de l'état ELB, comme décrit dans la rubrique Création de rapports d'intégrité de base
- Profils d'instance, comme décrit dans la rubrique Gestion des profils d'instance Elastic Beanstalk
- VPCs, comme décrit dans la Utilisation d'Elastic Beanstalk avec Amazon VPC rubrique
- Niveaux de données, comme décrit dans la rubrique Ajout d'une base de données à votre environnement Elastic Beanstalk
- Niveaux de travail, comme décrit dans la rubrique Environnements de travail Elastic Beanstalk
- Environnements d'instance unique, comme décrit dans la rubrique Types d'environnement
- Balises, comme décrit dans la rubrique Balisage des ressources dans vos environnements Elastic Beanstalk
- Mises à jour propagées, comme décrit dans la rubrique Mises à jour propagées de la configuration de l'environnement Elastic Beanstalk

Pourquoi certaines versions de plate-forme sont-elles marquées héritées ?

Certaines anciennes versions de plateforme ne prennent pas en charge les dernières fonctions Elastic Beanstalk. Ces versions affichent la mention (legacy) [(héritée)] sur la page de présentation de l'environnement dans la console Elastic Beanstalk.

Pour vérifier si vous utilisez une version de plate-forme héritée

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Sur la page de présentation de l'environnement, affichez le nom de la Platform (Plateforme).

   Votre application utilise une version de plateforme héritée si (legacy) [(héritée)] apparaît en regard du nom de la plateforme.

Pour migrer votre application

1. Déployez votre application dans un nouvel environnement. Pour plus d'informations, consultez Création d'un environnement Elastic Beanstalk.
2. Si vous disposez d'une instance DB Amazon RDS, mettez à jour votre groupe de sécurité de base de données afin d'autoriser l'accès à votre groupe de sécurité EC2 pour votre nouvel environnement. Pour savoir comment trouver le nom de votre groupe de sécurité EC2 à l'aide de la console AWS de gestion, consultezEC2 groupes de sécurité. Pour de plus amples informations sur la configuration de votre groupe de sécurité EC2, veuillez consulter la section « Autoriser un accès réseau à un groupe de sécurité Amazon EC2 » de la page Utilisation de groupes de sécurité de base de données dans le Guide de l'utilisateur Amazon Relational Database Service.
3. Echangez votre URL d'environnement. Pour plus d'informations, consultez Déploiements bleu/vert avec Elastic Beanstalk.
4. Suspendez votre ancien environnement. Pour plus d'informations, consultez Arrêt d'un environnement Elastic Beanstalk.

> **Note**
>
> Si vous utilisez Gestion des identités et des accès AWS (IAM), vous devrez mettre à jour vos politiques pour inclure CloudFormation Amazon RDS (le cas échéant). Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Gestion des identités et des accès AWS.

Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2

Cette section décrit comment migrer votre application à l'aide de l'un des chemins de migration suivants.

- Migrez depuis une branche de plateforme Amazon Linux 2 vers une branche de plateforme Amazon Linux 2023.
- Migrez d'une branche de plate-forme Amazon Linux AMI (AL1) vers une branche de plate-forme Amazon Linux 2023 (recommandé) ou vers une branche de plate-forme Amazon Linux 2.

Rubriques

- Migration depuis Amazon Linux 2 vers Amazon Linux 2023
- Migration depuis Amazon Linux AMI (AL1) vers AL2 ou AL2 023

Migration depuis Amazon Linux 2 vers Amazon Linux 2023

Cette rubrique fournit des conseils pour migrer votre application depuis une branche de plateforme Amazon Linux 2 vers une branche de plateforme Amazon Linux 2023.

Différences et compatibilité

Entre les plateformes Elastic Beanstalk AL2 et AL2 023

Il existe un degré élevé de compatibilité entre les plateformes Elastic Beanstalk Amazon Linux 2 et Amazon Linux 2023. Il y a cependant quelques différences à noter :

- Version 1 du service de métadonnées d'instance (IMDSv1) : le paramètre de l'IMDSv1option Disable est défini par défaut `true` sur les plateformes AL2 023. Par défaut, c'est `false` sur AL2 les plateformes.
- outil d'instance pkg-repo — L'pkg-repooutil n'est pas disponible pour les environnements exécutés sur AL2 les plateformes 023. Toutefois, vous pouvez toujours appliquer manuellement les mises à jour du package et du système d'exploitation à une instance AL2 023. Pour plus d'informations, consultez Gestion des packages et des mises à jour du système d'exploitation (français non garanti) dans le Guide de l'utilisateur Amazon Linux 2023.
- HTTPd Configuration Apache — Le `httpd.conf` fichier Apache pour les plateformes AL2 023 comporte des paramètres de configuration différents de ceux des AL2 plateformes suivantes :
  - Interdire l'accès à l'ensemble du système de fichiers du serveur par défaut. Ces paramètres sont décrits dans la section Protection par défaut des fichiers du serveur de la page Conseils sur la sécurité du site web d'Apache.
  - Refusez l'accès à la configuration de `.htaccess` dans tous les annuaires, à l'exception de ceux spécifiquement activés. Ce paramètre est décrit dans la section Protection de la configuration du système de la page Conseils sur la sécurité du site web d'Apache. La page Tutoriel du serveur HTTP Apache : fichiers .htaccess indique que ce paramètre peut contribuer à améliorer les performances.
  - Refuser l'accès aux fichiers portant le modèle de nom `.ht*`. Ce paramètre empêche les clients web de visualiser les fichiers `.htaccess` et `.htpasswd`.

Vous pouvez modifier les paramètres de configuration ci-dessus en fonction de votre environnement. Pour de plus amples informations, veuillez consulter Configuration d'Apache HTTPD.

- Prise en charge des variables d'environnement multilignes — Les plateformes AL2 023 prennent en charge les valeurs multilignes pour les variables d'environnement et les secrets dans les configurations de service Systemd. Les plateformes Amazon Linux 2 ne prennent pas en charge les valeurs de variables d'environnement multilignes. Cette amélioration vous permet d'utiliser des secrets multilignes et des valeurs de configuration sur les plateformes AL2 023. Pour plus d'informations sur l'utilisation des variables d'environnement et des secrets, consultezthe section called “Valeurs multilignes”.
- CloudWatch transfert de journal personnalisé — L'agent CloudWatch Logs obsolète (`awslogs`package) n'est pas disponible sur les plateformes AL2 023. Si vous avez des configurations personnalisées de transfert de journal qui installent et utilisent l'`awslogs`agent obsolète, vous devez mettre à jour vos fichiers de configuration pour utiliser l' CloudWatch agent unifié lors de la migration d'Amazon Linux 2 vers 023. AL2 Pour de plus amples informations, veuillez consulter the section called “Streaming de fichiers journaux personnalisés”.

Différences spécifiques à la plateforme

Outre les différences de système d'exploitation de base, il existe des différences spécifiques à la plate-forme entre les plateformes d'exécution Amazon Linux 2 et AL2 023 :

- Branchement de plate-forme .NET — La stratégie de branchement de la plate-forme .NET diffère entre Amazon Linux 2 et AL2 023. Sur Amazon Linux 2, la plate-forme .NET Core gère une fenêtre rotative des versions principales de .NET au sein d'une seule branche de plate-forme. Le AL2 023, chaque branche de plate-forme est épinglée à une version majeure de .NET spécifique (par exemple, .NET 9, .NET 10).

  Si vous déployez des applications dépendantes du framework (applications qui reposent sur le runtime .NET installé sur la plate-forme), vous devez sélectionner une branche de plate-forme correspondant à la version .NET cible de votre application. Si vous déployez des applications autonomes (applications qui regroupent leur propre environnement d'exécution .NET), vous pouvez utiliser n'importe quelle branche de la plate-forme .NET AL2 023, quelle que soit la version .NET de votre application, car celle-ci ne dépend pas du runtime installé sur la plate-forme. Pour de plus amples informations, veuillez consulter the section called “Regroupement d'applications”.

- Sélection de la version de Node.js — La plateforme Node.js sur Amazon Linux 2 permet de spécifier une version Node.js dans le `package.json` fichier de votre application. La plateforme Node.js sur AL2 023 ne prend pas en charge cette fonctionnalité. Vous devez utiliser la version par défaut de Node.js fournie par la branche de plateforme. Pour plus d'informations sur la gestion des versions de Node.js, consultezthe section called “Configuration des dépendances”.
- Version du serveur Ruby Puma — La plateforme Ruby sur Amazon Linux 2 ignore la version Puma spécifiée dans le `Gemfile.lock` fichier de votre application et utilise la version Puma par défaut de la plateforme. La plateforme Ruby du AL2 023 respecte la version Puma spécifiée dans le `Gemfile.lock` cas où elle est présente. Si aucune version n'est spécifiée, la plateforme installe la version Puma par défaut de la plateforme.
- Disponibilité des packages PHP — Certains packages disponibles sur les plateformes PHP Amazon Linux 2 ne sont pas disponibles sur les plateformes PHP AL2 023 :
  - Packages clients MySQL — Les packages clients de ligne de `mysql-devel` commande `mysql` et de ligne de commande ne sont pas installés sur les plateformes PHP AL2 023. Si votre application nécessite une connectivité à une base de données MySQL, utilisez le PHP `mysqli` ou des `pdo_mysql` extensions, disponibles sur les deux plateformes.
  - Outils Compass et Ruby — Les `rubygems` packages `ruby-devel` et pour le support du framework CSS Compass ne sont pas installés sur les plateformes PHP AL2 023. Compass est obsolète. Envisagez d'utiliser des outils de prétraitement CSS modernes comme alternative.
- Outils de contrôle de version Go — Le système de contrôle de version Bazaar (`bzr`) n'est pas disponible sur les plateformes AL2 023 Go. Bazaar est obsolète et n'est pas inclus dans le référentiel de packages AL2 023. Utilisez plutôt Git, Mercurial ou Subversion pour le contrôle de version, qui sont tous disponibles sur les plateformes AL2 023 Go.

Entre les systèmes d'exploitation Amazon Linux

Pour plus d'informations sur les différences entre les systèmes d'exploitation Amazon Linux 2 et Amazon Linux 2023, consultez Comparing Amazon Linux 2 and Amazon Linux 2023 dans le Guide de l'utilisateur Amazon Linux 2023.

Pour plus d'informations sur Amazon Linux 2023, consultez What is Amazon Linux 2023? dans le Guide de l'utilisateur Amazon Linux 2023.

Processus général de migration

Lorsque vous êtes prêt à passer en production, Elastic Beanstalk a blue/green besoin d'un déploiement pour effectuer la mise à niveau. Voici les meilleures pratiques générales que nous recommandons pour la migration avec une procédure de blue/green déploiement.

Préparer le test de votre migration

Avant de déployer votre application et de commencer les tests, passez en revue les informations de la section précédente Différences et compatibilité. Consultez également la référence citée dans la section Comparing Amazon Linux 2 and Amazon Linux 2023 dans le Guide de l'utilisateur Amazon Linux 2023. Prenez note des informations spécifiques de ce contenu qui s'appliquent ou peuvent s'appliquer à votre application et à votre configuration.

Étapes de migration de haut niveau

1. Créez un nouvel environnement basé sur une branche de plateforme AL2 023.
2. Déployez votre application dans l'environnement AL2 023 cible.

   Votre environnement de production existant restera actif et non affecté, pendant que vous procédez à des tests et des ajustements du nouvel environnement.
3. Testez votre application de manière approfondie dans le nouvel environnement.
4. Lorsque votre environnement AL2 023 de destination est prêt à passer en production, échangez les CNAMEs deux environnements pour rediriger le trafic vers le nouvel environnement AL2 023.

Étapes de migration plus détaillées et meilleures pratiques

Pour une procédure de blue/green déploiement plus détaillée, voirDéploiements bleu/vert avec Elastic Beanstalk.

Pour des conseils plus spécifiques et des étapes détaillées des meilleures pratiques, consultez Blue/Green method.

Plus de références pour vous aider à planifier votre migration

Les références suivantes peuvent fournir des informations supplémentaires pour planifier votre migration.

- Plateformes prises en charge par Elastic Beanstalk dans AWS Elastic Beanstalk Platforms (Plateformes)

- Historique des branches de plateforme retirées
- the section called “Plateformes Linux”
- FAQ sur le retrait de la plateforme

Migration depuis Amazon Linux AMI (AL1) vers AL2 ou AL2 023

Si votre application Elastic Beanstalk est basée sur une branche de plateforme AMI Amazon Linux, utilisez cette section pour savoir comment migrer les environnements de votre application vers Amazon Linux 2 ou Amazon Linux 2023. Les branches de plateforme de génération précédente basées sur l'AMI Amazon Linux sont désormais retirées.

Nous vous recommandons vivement de migrer vers Amazon Linux 2023, car ce système d'exploitation est plus récent qu'Amazon Linux 2. Le système d'exploitation Amazon Linux 2 atteindra la fin de sa prise en charge avant Amazon Linux 2023. Vous bénéficierez donc d'une période de prise en charge plus longue si vous migrez vers Amazon Linux 2023.

Il est intéressant de noter qu'il existe un degré élevé de compatibilité entre les plateformes Elastic Beanstalk Amazon Linux 2 et Amazon Linux 2023. Certains domaines présentent toutefois des différences : l'option par défaut du service de métadonnées d'instance version 1 (IMDSv1), la prise en charge de l'outil d'instance pkg-repo et certaines configurations Apache. HTTPd Pour de plus amples informations, consultez Amazon Linux 2023.

Différences et compatibilité

Il n'est pas garanti que les branches de la plateforme AL2 basées sur AL2 023/ soient rétrocompatibles avec votre application existante. Il est également important de savoir que même si votre code d'application se déploie avec succès sur la nouvelle version de plateforme, il peut se comporter ou fonctionner différemment en raison des différences de système d'exploitation et d'exécution.

Bien que l'AMI Amazon Linux et AL2 023/ AL2 partagent le même noyau Linux, ils diffèrent sur les points suivants : leur système d'initialisation, `libc` les versions, la chaîne d'outils du compilateur et les différents packages. Pour plus d'informations, consultez Amazon Linux 2 FAQs.

Le service Elastic Beanstalk a également mis à jour des versions de l'environnement d'exécution, d'outils de génération et d'autres dépendances spécifiques à la plateforme.

Par conséquent, nous vous recommandons de prendre votre temps, de tester soigneusement votre application dans un environnement de développement et d'effectuer les ajustements nécessaires.

Processus général de migration

Lorsque vous êtes prêt à passer en production, Elastic Beanstalk a blue/green besoin d'un déploiement pour effectuer la mise à niveau. Voici les meilleures pratiques générales que nous recommandons pour la migration avec une procédure de blue/green déploiement.

Préparer le test de votre migration

Avant de déployer votre application et de commencer les tests, consultez les informations contenues dans Considérations pour toutes les plateformes Linux, qui figure plus loin dans cette rubrique. Consultez également les informations qui s'appliquent à votre plateforme dans la section Considérations spécifiques à la plateforme qui suit. Prenez note des informations spécifiques de ce contenu qui s'appliquent ou peuvent s'appliquer à votre application et à votre configuration.

Étapes de migration de haut niveau

1. Créez un nouvel environnement basé sur une branche de plateforme AL2 ou AL2 023. Nous vous recommandons de migrer vers une branche de la plateforme AL2 023.
2. Déployez votre application dans l'environnement AL2 023/ AL2 cible.

   Votre environnement de production existant restera actif et non affecté, pendant que vous procédez à des tests et des ajustements du nouvel environnement.
3. Testez votre application de manière approfondie dans le nouvel environnement.
4. Lorsque votre AL2 environnement AL2 023/ de destination est prêt à passer en production, échangez les CNAMEs deux environnements pour rediriger le trafic vers le nouvel environnement.

Étapes de migration plus détaillées et meilleures pratiques

Pour une procédure de blue/green déploiement plus détaillée, voirDéploiements bleu/vert avec Elastic Beanstalk.

Pour des conseils plus spécifiques et des étapes détaillées des meilleures pratiques, consultez Blue/Green method.

Plus de références pour vous aider à planifier votre migration

Les références suivantes peuvent fournir des informations supplémentaires pour planifier votre migration.

- Comparing Amazon Linux 2 and Amazon Linux 2023 Guide de l'utilisateur Amazon Linux 2023.
- What is Amazon Linux 2023? dans le Guide de l'utilisateur Amazon Linux 2023
- Plateformes prises en charge par Elastic Beanstalk dans AWS Elastic Beanstalk Platforms (Plateformes)
- Historique des branches de plateforme retirées
- the section called “Plateformes Linux”
- FAQ sur le retrait de la plateforme

Considérations pour toutes les plateformes Linux

Le tableau suivant décrit les points à prendre en compte lors de la planification de la migration d'une application vers AL2 023/AL2. Ces considérations s'appliquent à toutes les plateformes Linux Elastic Beanstalk, quels que soient les langages de programmation ou les serveurs d'applications spécifiques.

Area	Modifications et informations
Fichiers de configuration	Sur les AL2 plateformes AL2 023/, vous pouvez utiliser les fichiers de configuration comme avant, et toutes les sections fonctionnent de la même manière. Cependant , des paramètres spécifiques peuvent ne pas fonctionner de la même manière que sur les plateformes AMI Amazon Linux précédentes. Par exemple : • Certains packages logiciels que vous installez à l'aide d'un fichier de configuration ne sont peut-être pas disponibles le AL2 023/ AL2 ou leur nom a peut-être changé. • Certaines options de configuration spécifiques à la plateforme ont vu leur espace de noms spécifique à la plate-forme être changé en un espace de noms différent et indépendant de la plateforme. • Les fichiers de configuration de proxy fournis dans le répertoire `.ebextensions/nginx` doivent être déplacés vers le répertoire des hooks de plateforme `.platform/nginx` . Pour en savoir plus, consultez Configuration du proxy inverse. Nous vous recommandons d'utiliser des hooks de plateforme pour exécuter du code personnalisé sur vos instances d'environnement. Vous pouvez toujours utiliser des

Area	Modifications et informations
	commandes et des commandes de conteneur dans les fichiers de configuration `.ebextensions` , mais elles ne sont pas aussi simples à utiliser. Par exemple, écrire des scripts de commande dans un fichier YAML peut être lourd et difficile à tester. Vous devez toujours utiliser des fichiers de configuration `.ebextensions` pour tout script nécessitant une référence à une ressource AWS CloudFormation .
Hooks de plateforme	AL2 les plateformes ont introduit une nouvelle façon d'étendre la plateforme de votre environnement en ajoutant des fichiers exécutables pour connecter des répertoires aux instances de l'environnement. Avec les versions précédentes de la plateforme Linux, vous avez peut-être utilisé des hooks de plateforme personnalisée. Ces hooks n'étaient pas conçus pour les plateformes gérées et n'étaient pas pris en charge, mais pouvaient fonctionner de manière utile dans certains cas. Avec les versions de AL2 plate-forme AL2 023/, les crochets de plate-forme personnalisés ne fonctionnent pas. Vous devez migrer tous les hooks vers les nouveaux hooks de plateforme. Pour plus d'informations, consultez Hooks de plateforme.

Area	Modifications et informations
Serveurs proxy pris en charge	AL2Les versions de AL2 plate-forme 023/ prennent en charge les mêmes serveurs proxy inverses que chaque plate-forme prise en charge dans ses versions de plate-forme AMI Amazon Linux. Toutes les versions de la plateforme AL2 023/ AL2 ; utilisent nginx comme serveur proxy inverse par défaut, à l'exception des plateform es ECS et Docker. Les plateformes Tomcat, Node.js, PHP et Python prennent également en charge Apache HTTPD comme alternative. Toutes les plateformes permettent la configuration du serveur proxy de manière uniforme, comme décrit dans cette section. Toutefois, la configuration du serveur proxy est légèrement différente de celle de l'AMI Amazon Linux. Voici les différences pour toutes les plateformes : • La valeur par défaut est nginx — Le serveur proxy par défaut sur AL2 toutes les versions de la plateforme AL2 023/ est nginx. Sur les versions de plateforme AMI Amazon Linux de Tomcat, PHP et Python, le serveur proxy par défaut était Apache HTTPD. • Espace de noms cohérent — Toutes les versions de la AL2 plate-forme AL2 023/ utilisent l'espace de `aws:elasticbeanstalk:environment:pro xy` noms pour configurer le serveur proxy. Sur les versions de plateforme AMI Amazon Linux, il s'agissait d'une décision par plateforme, et Node.js utilisait un espace de noms différent. • Emplacement du fichier de configuration — Vous devez placer les fichiers de configuration du proxy dans les `.platform/httpd` répertoires `.platform/ nginx` et sur toutes les versions de la AL2 plateforme AL2 023/. Sur les versions de la plateforme AMI Amazon Linux, ces emplacements étaient `.ebextens ions/nginx` et `.ebextensions/httpd` , respectivement. Pour les modifications de configuration de proxy spécifiques à la plateforme, veuillez consulter the section called “Considérations spécifiques à la plateforme”. Pour plus d'informations sur la configuration du proxy sur les AL2 plateformes AL2 023/, consultez. Configuration du proxy inverse

Area	Modifications et informations
Modifications de la configuration du proxy	Il y a des modifications de configuration de proxy qui s'appliquent uniformément à toutes les plateformes, en plus des modifications de configuration de proxy spécifiques à chaque plateforme. Il est important de se référer aux deux pour configurer avec précision vos environnements. • Toutes les plateformes — voir Configuration du proxy inverse • Spécifique aux plateformes – consultez the section called “Considérations spécifiques à la plateforme”.
Profil d’instance	AL2Les AL2 plateformes 023/ nécessitent la configuration d'un profil d'instance. La création de l'environnement peut aboutir temporairement sans profil d'instance, mais l'environnement peut afficher des erreurs rapidement après sa création lorsque des actions nécessitant un profil d'instance commencent à échouer. Pour en savoir plus, consultez the section called “Profils d’instance”.
Intégrité améliorée	AL2Les versions de AL2 plate-forme 023/ permettent d'améliorer l'état de santé par défaut. Il s'agit d'un changement si vous n'utilisez pas la console Elastic Beanstalk pour créer vos environnements. La console active l'intégrité améliorée par défaut chaque fois que cela est possible, quelle que soit la version de la plateforme. Pour en savoir plus, consultez the section called “Rapports d'intégrité améliorés et surveillance”.
AMI personnalisée	Si votre environnement utilise une AMI personnalisée, créez une nouvelle AMI basée sur AL2 023/ AL2 pour votre nouvel environnement à l'aide d'une plateforme Elastic Beanstalk 023/. AL2 AL2
Plateformes personnalisées	Les versions gérées AMIs de la AL2 plateforme AL2 023/ ne prennent pas en charge les plateformes personnalisées.

Considérations spécifiques à la plateforme

Cette section traite des considérations de migration spécifiques à certaines plateformes Linux Elastic Beanstalk.

Docker

La famille de branches de la plateforme Docker basée sur l'AMI Amazon Linux (AL1) comprend trois branches de plate-forme. Nous recommandons un chemin de migration différent pour chacun d'entre eux.

AL1 Branche de la plateforme	Trajectoire de migration vers AL2 023/ AL2
Docker multi-conteneurs géré par Amazon ECS s'exécutant sur l'AMI Amazon Linux () AL1	Docker AL2 AL2 023/branches de la plateforme basé sur ECS Les branches de AL2 plate-forme Docker AL2 023/ basées sur ECS offrent un chemin de migration simple pour les environnements exécutés sur la branche de plate-forme Docker multi-conteneurs. AL1 • Comme la précédente AL1 branche Docker multi-conteneurs, les AL2 branches de la plateforme AL2 023/ utilisent Amazon ECS pour coordonner le déploiement de plusieurs conteneurs Docker sur un cluster Amazon ECS dans un environnement Elastic Beanstalk. • Les branches de la AL2 plateforme AL2 023/ prennent en charge toutes les fonctionnalités de la précédente branche Docker multi-conteneurs. AL1 • Les branches de la AL2 plateforme AL2 023/ supportent également le même fichier `Dockerrun.aws.json` v2. Pour plus d'informations sur la migration de vos applications exécutées sur la branche multi-conteneurs de la plateforme Docker Amazon Linux vers une branche Amazon ECS exécutée AL2 sur la branche AL2 023/ de la plateforme, consultez. ???
Docker exécuté sur l'AMI Amazon Linux () AL1	Docker exécuté sur la branche AL2 023/ plateforme AL2 Nous vous recommandons de migrer vos applications exécutées sur des environnements basés sur Docker préconfiguré (Glassfish 5.0) ou Docker exécuté sur Amazon Linux AMI (AL1) vers des environnements basés sur le Docker exécuté sur Amazon Linux 2 ou Docker exécuté sur 023 branches de plate-forme. AL2

<table>
<tr><td>AL1 Branche de la plateforme</td><td>Trajectoire de migration vers AL2 023/ AL2</td></tr>
<tr><td>Docker préconfiguré (Glassfish 5.0) exécutant Amazon Linux AMI () AL1</td><td>
Si votre environnement repose sur la branche de plateforme Preconfigured Docker (Glassfish 5.0) (Docker préconfiguré [Glassfish 5.0]), consultez the section called “Tutoriel - GlassFish sur Docker : chemin vers Amazon Linux 2023”.

Le tableau suivant répertorie les informations de migration spécifiques à la branche de plate-forme Docker Running on AL2 AL2 023/.

<table>
<tr><td>Area</td><td>Modifications et informations</td></tr>
<tr><td>Stockage</td><td>Elastic Beanstalk configure Docker pour qu'il utilise des pilotes de stockage afin de stocker des images Docker et des données de conteneur. Sur l'AMI Amazon Linux, Elastic Beanstalk a utilisé le pilote de stockage Device Mapper. Pour améliorer les performances, Elastic Beanstalk a provisionné un volume Amazon EBS supplémentaire. Dans les versions AL2 023/ de la plateforme AL2 Docker, Elastic Beanstalk utilise le pilote de stockage OverlayFS et atteint des performances encore meilleures sans avoir besoin d'un volume séparé.

Avec l'AMI Amazon Linux, si vous avez utilisé l'option <code>BlockDeviceMappings</code> de l'espace de noms <code>aws:autoscaling:launchconfiguration</code> pour ajouter des volumes de stockage personnalisés à un environnement Docker, nous vous conseillons d'ajouter également le volume <code>/dev/xvdcz</code> Amazon EBS prévu par Elastic Beanstalk. Elastic Beanstalk ne provisionne plus ce volume, vous devriez donc le supprimer de vos fichiers de configuration. Pour en savoir plus, consultez the section called “Configuration Docker sur l'AMI Amazon Linux (antérieure à Amazon Linux 2)”.</td></tr>
<tr><td>Authentification du</td><td>Lorsque vous fournissez un fichier d'authentification généré par Docker pour vous connecter à un référentiel privé, vous n'avez plus besoin de le convertir dans l'ancien format requis par les versions de la plateforme Amazon Linux AMI Docker. AL2023/ Les versions de la</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td>AL1 Branche de la plateforme</td><td>Trajectoire de migration vers AL2 023/ AL2</td></tr>
<tr><td></td><td>
<table>
<tr><th>Area</th><th>Modifications et informations</th></tr>
<tr><td>référentiel privé</td><td>plateforme AL2 Docker prennent en charge le nouveau format. Pour en savoir plus, consultez the section called “Authentification avec des référentiels d'images”.</td></tr>
<tr><td>Serveur proxy</td><td>AL2023/ Les versions de la plateforme AL2 Docker ne prennent pas en charge les conteneurs autonomes qui ne s'exécutent pas derrière un serveur proxy. Sur les versions de la plateforme AMI Docker Amazon Linux, cela était possible grâce à la valeur none de l'option <code>ProxyServer</code> dans l'espace de noms <code>aws:elasticbeanstalk:environment:proxy</code> .</td></tr>
</table>
</td></tr>
</table>

Go

Le tableau suivant répertorie les informations de migration pour les versions de AL2 plate-forme AL2 023/ de la plate-forme Go.

Area	Modifications et informations
Transmission de port	Sur les AL2 plateformes AL2 023/, Elastic Beanstalk ne transmet pas de valeur de port à votre processus d'application par le biais de la variable d'environnement. <code>PORT</code> Vous pouvez simuler ce comportement pour votre processus en configurant vous-même une propriété d'environnement <code>PORT</code>. Cependant, si vous avez plusieurs processus et que vous comptez sur Elastic Beanstalk pour transmettre des valeurs de port incrémentielles à vos processus (5000, 5100, 5200, etc.), vous devez modifier votre implémentation. Pour plus d’informations, consultez Configuration du proxy inverse.

Amazon Corretto

Le tableau ci-dessous répertorie les informations de migration pour les branches de plateforme Corretto dans la plateforme Java SE.

Area	Modifications et informations
Corretto et OpenJDK	Pour implémenter la plate-forme Java, édition standard (Java SE), les branches de la AL2 plateforme AL2 023/ utilisent Amazon Corretto, une AWS distribution du kit de développement Open Java (OpenJDK). Les branches précédentes de la plateforme Elastic Beanstalk Java SE utilisent les packages OpenJDK inclus avec l'AMI Amazon Linux.
Outils de génération	AL2Les AL2 plateformes 023/ ont des versions plus récentes des outils de construction :`gradle`,`maven`, et. `ant`
Gestion des fichiers JAR	Sur les AL2 plateformes AL2 023/, si votre bundle source (fichier ZIP) contient un seul fichier JAR et aucun autre fichier, Elastic Beanstalk ne renomme plus le fichier JAR en. `application.jar` Le changement de nom se produit uniquement si vous soumettez un fichier JAR seul et non inclus dans un fichier ZIP.
Transmission de port	Sur les AL2 plateformes AL2 023/, Elastic Beanstalk ne transmet pas de valeur de port à votre processus d'application par le biais de la variable d'environnement. `PORT` Vous pouvez simuler ce comportement pour votre processus en configurant vous-même une propriété d'environnement `PORT`. Cependant, si vous avez plusieurs processus et que vous comptez sur Elastic Beanstalk pour transmettre des valeurs de port incrémentielles à vos processus (5000, 5100, 5200, etc.), vous devez modifier votre implémentation. Pour plus d'informations, consultez Configuration du proxy inverse.
Java 7	Elastic Beanstalk ne prend pas en AL2 charge AL2 une branche de plate-forme 023/Java 7. Si vous avez une application Java 7, migrez vers Corretto 8 ou Corretto 11.

Tomcat

Le tableau suivant répertorie les informations de migration pour les versions de plate-forme AL2 023/ de la AL2 plate-forme Tomcat.

<table>
<tr><th>Area</th><th>Modifications et informations</th></tr>
<tr><td>Options de configuration</td><td>Sur les versions de AL2 plate-forme AL2 023/, Elastic Beanstalk ne prend en charge qu'un sous-ensemble des options de configuration et des valeurs d'options de l'espace de noms. <code>aws:elasticbeanstalk:environment:proxy</code> Voici les informations de migration pour chaque option.
<table>
<tr><th>Option</th><th>Informations sur la migration</th></tr>
<tr><td><code>GzipCompression</code></td><td>Non pris en charge sur les versions de AL2 plate-forme 023/AL2 .</td></tr>
<tr><td><code>ProxyServer</code></td><td>AL2023/ Les versions de la plateforme AL2 Tomcat supportent à la fois les serveurs proxy nginx et Apache HTTTD version 2.4. Cependant, Apache version 2.2 n'est pas prise en charge.

Sur les versions de la plateforme AMI Amazon Linux, le proxy par défaut était Apache 2.4. Si vous avez utilisé le paramètre de proxy par défaut et ajouté des fichiers de configuration de proxy personnalisés, votre configuration de proxy devrait toujours fonctionner le AL2 023/AL2. Cependant, si vous avez utilisé la valeur de l'option <code>apache/2.2</code> , vous devez maintenant migrer votre configuration proxy vers Apache version 2.4.</td></tr>
</table>
L'<code>XX:MaxPermSize</code> option dans l'espace de <code>aws:elasticbeanstalk:container:tomcat:jvmoptions</code> noms n'est pas prise en charge sur les versions de AL2 plate-forme 023/AL2 . Le paramètre JVM permettant de modifier la taille de la génération permanente s'applique uniquement à Java 7 et aux versions antérieures, et ne s'applique donc pas aux versions de plate-forme AL2 023/AL2 .</td></tr>
<tr><td>Chemin d’accès de l’application</td><td>Sur les AL2 plateformes AL2 023/, le chemin d'accès au répertoire de l'application sur les EC2 instances Amazon de votre environnement est. <code>/var/app/current</code> C'était <code>/var/lib/tomcat8/webapps</code> sur les plateformes AMI Amazon Linux.</td></tr>
</table>

Node.js

Le tableau suivant répertorie les informations de migration pour les versions de AL2 plate-forme AL2 023/ de la plate-forme Node.js.

<table>
<tr><th>Area</th><th>Modifications et informations</th></tr>
<tr><td>Versions de Node.js installées</td><td>Sur les AL2 plateformes AL2 023/, Elastic Beanstalk gère plusieurs branches de plate-forme Node.js et n'installe que la dernière version de la version majeure de Node.js correspondant à la branche de plate-forme de chaque version de plate-forme. Par exemple, chaque version de plateforme dans la branche de plateforme Node.js 12 n'a que Node.js 12.x.y installé par défaut. Sur les versions de plateforme AMI Amazon Linux, nous avons installé les versions multiples de plusieurs versions de Node.js sur chaque version de plateforme, et nous n'avons maintenu qu'une seule branche de plateforme.

Choisissez la branche de plateforme Node.js qui correspond à la version majeure de Node.js dont votre application a besoin.</td></tr>
<tr><td>Noms des fichiers journaux HTTPD Apache</td><td>Sur les AL2 plateformes AL2 023/, si vous utilisez le serveur proxy HTTTD Apache, les noms des fichiers journaux HTTTD sont « <code>access_log</code> et »<code>error_log</code> , ce qui est cohérent avec toutes les autres plateformes supportant Apache HTTTD. Sur les versions de la plateforme d'AMI Amazon Linux, ces fichiers journaux étaient nommés <code>access.log</code> et <code>error.log</code> , respectivement.

Pour de plus amples informations sur les noms et les emplacements des fichiers journaux de toutes les plateformes, veuillez consulter the section called “Comment Elastic Beanstalk configure les journaux CloudWatch”.</td></tr>
<tr><td>Options de configuration</td><td>Sur les AL2 plateformes AL2 023/, Elastic Beanstalk ne prend pas en charge les options de configuration dans l'espace de noms. <code>aws:elasticbeanstalk:container:nodejs</code> Certaines options ont des solutions de rechange. Voici les informations de migration pour chaque option.

<table>
<tr><th>Option</th><th>Informations sur la migration</th></tr>
<tr><td><code>NodeCommand</code></td><td>Utilisez un <code>Procfile</code> ou le mot-clé <code>scripts</code> dans un fichier <code>package.json</code> pour spécifier le script de démarrage.</td></tr>
</table>
</td></tr>
</table>

Area	Modifications et informations

Option	Informations sur la migration
`NodeVersion`	Utilisez le mot-clé `engines` dans un fichier `package.json` pour spécifier la version de Node.js. Sachez que vous ne pouvez spécifier qu'une version de Node.js qui correspond à votre branche de plateforme. Par exemple, si vous utilisez la branche de plateforme Node.js 12, vous ne pouvez spécifier qu'une version 12.x.y de Node.js. Pour en savoir plus, consultez the section called “Spécification des dépendances Node.js avec un fichier package.json”.
`GzipCompression`	Non pris en charge sur les versions de AL2 plate-forme 023/AL2 .

`ProxyServer`	Dans les versions de plate-forme AL2 023/ AL2 Node.js, cette option a été déplacée vers l'`aws:elasticbeanstalk:environment:proxy` espace de noms. Vous pouvez choisir entre `nginx` (par défaut) et `apache`.
	
	AL2Les versions de la plateforme 023/ AL2 Node.js ne prennent pas en charge les applications autonomes qui ne s'exécutent pas derrière un serveur proxy. Sur les versions de la plateforme Node.js d'AMI Amazon Linux, cela était possible via la valeur `none` de l'option `ProxyServer` dans l'espace de noms `aws:elasticbeanstalk:container:nodejs` . Si votre environnement exécute une application autonome, mettez à jour votre code pour écouter le port vers lequel le serveur proxy (nginx ou Apache) transfère le trafic.
	
	`var port = process.env.PORT || 5000;`
	
	`app.listen(port, function() {`
	`  console.log('Server running at http://127.0.0.1:%s',`
	` port);`
	`});`

PHP

Le tableau suivant répertorie les informations de migration pour les versions de la AL2 plate-forme AL2 023/ dans la plate-forme PHP.

Area	Modifications et informations
Traitement des fichiers PHP	Sur les AL2 plateformes AL2 023/, les fichiers PHP sont traités à l'aide de PHP-FPM (un gestionnaire de processus CGI). Sur les plateformes AMI Amazon Linux, nous avons utilisé mod_php (un module Apache).
Serveur proxy	AL2023/ Les versions de la plateforme AL2 PHP supportent à la fois les serveurs proxy nginx et Apache HTTTD. La valeur par défaut est nginx. Les versions de la plateforme PHP AMI Amazon Linux prenaient uniquement en charge Apache HTTPD. Si vous avez ajouté des fichiers de configuration Apache personnalisés, vous pouvez définir l'option `ProxyServer` dans l'espace de noms `aws:elasticbeanstalk:environment:proxy` sur `apache`.

Python

Le tableau suivant répertorie les informations de migration pour les versions de la AL2 plate-forme AL2 023/ dans la plate-forme Python.

Area	Modifications et informations
Serveur WSGI	Sur les AL2 plateformes AL2 023/, Gunicorn est le serveur WSGI par défaut. Par défaut, Gunicorn écoute sur le port 8000. Le port peut être différent de celui utilisé par votre application sur la plateforme d'AMI Amazon Linux. Si vous définissez l'option `WSGIPath` de l'espace de noms `aws:elasticbeanstalk:container:python`, remplacez la valeur par la syntaxe de Gunicorn. Pour en savoir plus, consultez the section called “Espaces de noms de la configuration Python”. Vous pouvez également utiliser un `Procfile` pour spécifier et configurer le serveur WSGI. Pour en savoir plus, consultez the section called “Procfile”.

Area	Modifications et informations
Chemin d'accès de l'application	Sur les AL2 plateformes AL2 023/, le chemin d'accès au répertoire de l'application sur les EC2 instances Amazon de votre environnement est. `/var/app/current` C'était `/opt/python/current/app` sur les plateformes AMI Amazon Linux.
Serveur proxy	AL2023/ Les versions de la plateforme AL2 Python supportent à la fois les serveurs proxy nginx et Apache HTTTD. La valeur par défaut est nginx. Les versions de plateforme AMI Python Amazon Linux prennent uniquement en charge Apache HTTPD. Si vous avez ajouté des fichiers de configuration Apache personnalisés, vous pouvez définir l'option `ProxyServer` dans l'espace de noms `aws:elasticbeanstalk:environment:proxy` sur `apache`.

Ruby

Le tableau suivant répertorie les informations de migration pour les versions de la plate-forme AL2 023/ dans la AL2 plate-forme Ruby.

Area	Modifications et informations
Versions de Ruby installées	Sur les AL2 plateformes AL2 023/, Elastic Beanstalk installe uniquement la dernière version d'une seule version de Ruby, correspondant à la branche de plate-forme, sur chaque version de plate-forme. Par exemple, chaque version de plateforme de la branche de plateforme Ruby 2.6 dispose uniquement de Ruby 2.6.x installé. Sur les versions de la plateforme AMI Amazon Linux, nous avons installé les dernières versions de plusieurs versions de Ruby, par exemple 2.4.x, 2.5.x et 2.6.x. Si votre application utilise une version de Ruby qui ne correspond pas à la branche de plateforme que vous utilisez, nous vous recommandons de basculer vers une branche de plateforme qui dispose de la version de Ruby correcte pour votre application.
Serveur d'application	Sur les AL2 plateformes AL2 023/, Elastic Beanstalk installe le serveur d'applications Puma uniquement sur toutes les versions de la plateforme Ruby. Vous pouvez utiliser un `Procfile` pour démarrer un autre serveur d'applications et un `Gemfile` pour l'installer.

Area	Modifications et informations
	Sur la plateforme AMI Amazon Linux, nous avons pris en charge deux types de branches de plateforme pour chaque version Ruby : l'une avec le serveur d'applications Puma et l'autre avec le serveur d'applications Passenger. Si votre application utilise Passenger, vous pouvez configurer votre environnement Ruby pour installer et utiliser Passenger. Pour plus d'informations et d'exemples, consultez the section called "La plateforme Ruby".

FAQ sur le retrait de la plateforme

> **Note**
>
> Elastic Beanstalk a retiré toutes les branches de la plateforme basées sur l'AL1AMI Amazon Linux () le 18 juillet 2022.

Les réponses de cette FAQ font référence aux sujets suivants :

- Stratégie de prise en charge de la plateforme Elastic Beanstalk
- Historique des branches de plateforme retirées
- Plateformes prises en charge par Elastic Beanstalk dans AWS Elastic Beanstalk Platforms (Plateformes)
- Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2
- Amazon Linux 2 FAQs.

1. Que signifie le retrait d'une branche de plateforme ?

Après la date de retrait annoncée d'une branche de plateforme, vous ne pourrez plus créer de nouvel environnement basé sur la branche de plateforme retirée, sauf si vous disposez déjà d'un environnement actif basé sur cette branche de plateforme. Pour en savoir plus, consultez la FAQ n° 11. Elastic Beanstalk cessera de fournir de nouvelles mises à jour de maintenance pour ces branches de plateforme. Une branche de plateforme retirée n'est pas recommandée pour une utilisation dans des environnements de production. Pour en savoir plus, consultez la FAQ n° 5.

2. Pourquoi les succursales AL1 basées sur les plateformes ont-elles été AWS supprimées ?

Elastic Beanstalk retire les branches de plateforme lorsque les composants de la plateforme sont obsolètes ou retirés par leurs fournisseurs. Dans ce cas, l'AMI Amazon Linux (AL1) a mis fin au support standard le 31 décembre 2020. Alors qu'Elastic Beanstalk a continué AL1 à proposer des plateformes basées jusqu'en 2022, nous avons AL2 depuis AL2023 publié et basé des plateformes dotées des fonctionnalités les plus récentes. Pour que les clients puissent continuer à bénéficier des dernières fonctionnalités et fonctionnalités de sécurité à l'avenir, il est essentiel qu'ils migrent vers nos plateformes AL2 ou vers des plateformes AL2023 basées sur nous.

3. Quelles sont les branches de plateforme retirées ?

Pour obtenir la liste des composants de plateforme et des branches de plateforme qui ont été retirés, consultez Historique des branches de plateforme retirées.

4. Quelles plateformes sont actuellement prises en charge ?

Consultez Plateformes prises en charge par Elastic Beanstalk dans Plateformes AWS Elastic Beanstalk .

5. Elastic Beanstalk supprimera-t-il ou mettra-t-il fin à des composants de mon environnement après le retrait ?

Notre politique concernant les succursales de plateforme retirées ne supprime pas l'accès aux environnements ni les ressources. Toutefois, un environnement basé sur une branche de plateforme retirée peut se retrouver dans une situation imprévisible, car Elastic Beanstalk n'est pas en mesure de fournir des mises à jour de sécurité, un support technique ou des correctifs logiciels pour les branches de plateforme retirées en raison du fait que le fournisseur a marqué leur composant comme étant en fin de vie (EOL). Par exemple, une vulnérabilité de sécurité néfaste et critique peut apparaître dans un environnement exécuté sur une branche de plateforme retirée. Ou encore, une action de l'API EB peut cesser de fonctionner pour l'environnement si elle devient incompatible avec le service Elastic Beanstalk au fil du temps. La possibilité de ce type de risques augmente avec la durée d'activité d'un environnement basé sur une branche de plateforme retirée.

Si votre application rencontre des problèmes lors de son exécution sur une branche de plate-forme abandonnée et que vous ne parvenez pas à la migrer vers une plate-forme prise en charge, vous devrez envisager d'autres solutions. Les solutions de contournement consistent à encapsuler

l'application dans une image Docker pour l'exécuter en tant que conteneur Docker. Cela permettrait à un client d'utiliser n'importe laquelle de nos solutions Docker, telles que nos plateformes AL2023 Elastic AL2 Beanstalk/Docker, ou d'autres services basés sur Docker tels qu'Amazon ECS ou Amazon EKS. Les alternatives autres que Docker incluent notre AWS CodeDeploy service, qui permet une personnalisation complète des environnements d'exécution que vous souhaitez.

6. Puis-je soumettre une demande de report de la date de retrait ?

Non. Après la date de retrait, les environnements existants continueront à fonctionner. Toutefois, Elastic Beanstalk ne fournira plus de mise à jour de sécurité et de maintenance de la plateforme. Par conséquent, il est essentiel de migrer vers une plate-forme basée AL2 ou AL2023 si vous exécutez toujours des applications sur une plate-forme AL1 basée. Pour plus d'informations sur les risques et les solutions de contournement, consultez FAQ n° 5.

7. Quelles sont les solutions si je ne parviens pas à terminer ma AL2023 migration AL2 ou ma migration à temps ?

Les clients peuvent continuer à exécuter l'environnement, bien que nous vous encouragions fortement à planifier la migration de tous vos environnements Elastic Beanstalk vers une version de plateforme prise en charge. Cela permet de minimiser les risques et de continuer à profiter des améliorations importantes en matière de sécurité, de performances et de fonctionnalités offertes par les versions les plus récentes. Pour plus d'informations sur les risques et les solutions de contournement, consultez FAQ n° 5.

8. Quel est le processus recommandé pour migrer vers AL2 nos AL2023 plateformes ?

Pour AL1 obtenir des instructions complètes sur AL2 la migration vers AL2023//, consultezMigration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2. Cette rubrique explique qu'Elastic Beanstalk blue/green nécessite un déploiement pour effectuer la mise à niveau.

9. Si j'ai un environnement qui s'exécute sur une plateforme retirée, quel est l'impact ?

Un environnement basé sur une branche de plateforme retirée peut se retrouver dans une situation imprévisible, car Elastic Beanstalk n'est pas en mesure de fournir des mises à jour de sécurité, un support technique ou des correctifs logiciels pour les branches de plateforme retirées en raison du fait que le fournisseur a marqué leur composant comme étant en fin de vie (EOL). Par exemple, une vulnérabilité de sécurité néfaste et critique peut apparaître dans un environnement exécuté sur une

branche de plateforme retirée. Ou encore, une action de l'API EB peut cesser de fonctionner pour l'environnement si elle devient incompatible avec le service Elastic Beanstalk au fil du temps. La possibilité de ce type de risques augmente avec la durée d'activité d'un environnement basé sur une branche de plateforme retirée. Pour en savoir plus, consultez la FAQ n° 5.

10. Que se passe-t-il 90 jours après la date du retrait ?

Notre politique concernant les succursales de plateforme retirées ne supprime pas l'accès aux environnements ni les ressources. Toutefois, un environnement basé sur une branche de plateforme retirée peut se retrouver dans une situation imprévisible, car Elastic Beanstalk n'est pas en mesure de fournir des mises à jour de sécurité, un support technique ou des correctifs logiciels pour les branches de plateforme retirées en raison du fait que le fournisseur a marqué leur composant comme étant en fin de vie (EOL). Par exemple, une vulnérabilité de sécurité néfaste et critique peut apparaître dans un environnement exécuté sur une branche de plateforme retirée. Ou encore, une action de l'API EB peut cesser de fonctionner pour l'environnement si elle devient incompatible avec le service Elastic Beanstalk au fil du temps. La possibilité de ce type de risques augmente avec la durée d'activité d'un environnement basé sur une branche de plateforme retirée. Pour en savoir plus, consultez la FAQ n° 5.

11. Puis-je créer un nouvel environnement basé sur une plateforme retirée ?

Vous pouvez créer un nouvel environnement basé sur une branche de plateforme retirée, si vous avez déjà utilisé cette branche de plateforme pour créer un environnement existant à l'aide du même compte et dans la même région. La branche de plateforme retirée ne sera pas disponible dans la console Elastic Beanstalk. Toutefois, pour les clients qui disposent d'environnements existants basés sur une branche de plateforme retirée, elle sera disponible via l'interface de ligne de commande EB, l'API EB et AWS CLI. De même, les clients existants peuvent utiliser les consoles Clone environment (Cloner un environnement) et Rebuild environment (Reconstruire un environnement). Toutefois, un environnement basé sur une branche de plateforme retirée peut se retrouver dans une situation imprévisible. Pour en savoir plus, consultez la FAQ n° 5.

12. Si j'ai un environnement existant qui fonctionne sur une branche de plate-forme retirée, jusqu'à quand puis-je créer un nouvel environnement basé sur la branche de plate-forme supprimée ? Puis-je le faire à l'aide de la console, de l'interface de ligne de commande ou de l'API ?

Vous pouvez créer l'environnement après la date de départ à la retraite. Toutefois, il faut garder à l'esprit qu'une branche de plateforme retirée peut se retrouver dans une situation imprévisible. Plus

un tel environnement est créé ou actif dans le temps, plus le risque qu'il rencontre des problèmes inattendus est élevé. Pour plus d'informations sur la création d'un environnement de développement, consultez la FAQ n° 11.

13. Puis-je cloner ou reconstruire mon environnement basé sur une plateforme retirée ?

Oui. Vous pouvez le faire en utilisant les consoles Clone environment (Cloner un environnement) Rebuild environment (Reconstruire l'environnement) consoles. Vous pouvez également utiliser l'interface de ligne de commande EB, l'API EB et AWS CLI. Pour plus d'informations sur la création d'un environnement de développement, consultez la FAQ n° 11.

Cependant, nous vous encourageons fortement à planifier la migration de tous vos environnements Elastic Beanstalk vers une version de plateforme prise en charge. Cela permet de minimiser les risques et de continuer à profiter des améliorations importantes en matière de sécurité, de performances et de fonctionnalités offertes par les versions les plus récentes. Pour plus d'informations sur les risques et les solutions de contournement, consultez FAQ n° 5.

14. Après la date de mise hors service, qu'adviendrait-il des AWS ressources de mon environnement Elastic Beanstalk basé sur une branche de plateforme abandonnée ? Par exemple, si l'instance EC2 en cours d'exécution est supprimée, Elastic Beanstalk sera-t-il capable de lancer une nouvelle instance EC2 basée sur AL1 pour maintenir la capacité ?

Les ressources de l'environnement resteront actives et continueront de fonctionner. Et oui, Elastic Beanstalk s'adaptera aux instances EC2 AL1 dans l'environnement. Toutefois, Elastic Beanstalk cessera de fournir de nouvelles mises à jour de maintenance de la plateforme pour l'environnement, ce qui peut conduire à ce que l'environnement se retrouve dans une situation imprévisible au fil du temps. Pour en savoir plus, consultez la FAQ n° 5.

15. Quelles sont les principales différences entre les systèmes d'exploitation AL2023/AL2 et Amazon Linux AMI (AL1) ? Comment les branches d' AL2023Elastic AL2 Beanstalk/de la plateforme sont-elles affectées ?

Bien que l'AMI Amazon Linux et AL2023/AL2 partagent le même noyau Linux, ils diffèrent en termes de système d'initialisation, de `libc` versions, de chaîne d'outils de compilation et de différents packages. Pour plus d'informations, consultez Amazon Linux 2 FAQs.

Le service Elastic Beanstalk a également mis à jour des versions de l'environnement d'exécution, d'outils de génération et d'autres dépendances spécifiques à la plateforme. Il n'est pas garanti que les branches de la plateforme AL2 basées sur AL2023/soient rétrocompatibles avec votre application existante. En outre, même si votre code d'application se déploie avec succès sur la nouvelle version de plateforme, il peut se comporter ou fonctionner différemment en raison des différences de système d'exploitation et d'exécution. Pour une liste et une description des configurations et des personnalisations que vous devrez examiner et tester, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Annulation de mises à jour de configuration d'environnement et de déploiements d'application

Vous pouvez annuler des mises à jour en cours qui sont déclenchées par des changements de configuration d'environnement. Vous pouvez également annuler le déploiement d'une nouvelle version de l'application en cours. Par exemple, vous pouvez vouloir annuler une mise à jour si vous décidez que vous souhaitez continuer à utiliser la configuration de l'environnement existant au lieu d'appliquer les nouveaux paramètres de configuration d'environnement. Ou bien, vous pouvez réaliser que la nouvelle version de l'application que vous déployez a des problèmes susceptibles de ne pas démarrer ou de ne pas fonctionner correctement. En annulant une mise à jour d'environnement ou de version d'application, vous pouvez éviter d'attendre que le processus de mise à jour ou de déploiement soit effectué avant de lancer une nouvelle tentative pour mettre à jour la version de l'environnement ou de l'application.

> **Note**
>
> Pendant la phase de nettoyage au cours de laquelle les anciennes ressources qui ne sont plus nécessaires sont supprimées, après que le dernier lot d'instances a été mis à jour, vous ne pouvez pas plus annuler la mise à jour.

Elastic Beanstalk effectue la restauration de la même manière qu'il a réalisé la dernière mise à jour réussie. Par exemple, si des mises à jour propagées à durée définie sont activées dans votre environnement, alors Elastic Beanstalk attend le temps de pause spécifié entre la restauration des modifications sur un lot d'instances avant de restaurer les modifications sur le lot suivant. Ou bien, si vous avez récemment activé des mises à jour propagées, mais que la dernière fois que vous avez réussi une mise à jour des paramètres de configuration de votre environnement elle n'incluait pas

de mises à jour propagées, Elastic Beanstalk procède à la restauration sur toutes les instances en même temps.

Une fois qu'il commence à annuler la mise à jour, vous ne pouvez pas arrêter la restauration d'Elastic Beanstalk vers la configuration de l'environnement précédent. Le processus de restauration continue jusqu'à ce que toutes les instances de l'environnement aient la configuration de l'environnement précédent ou jusqu'à ce que le processus de restauration échoue. Pour les déploiements de version d'application, l'annulation du déploiement arrête tout simplement le déploiement ; certaines instances auront la nouvelle version de l'application et d'autres continueront à exécuter la version d'application existante. Vous pouvez déployer la même ou une autre version de l'application ultérieurement.

Pour plus d'informations sur les mises à jour propagées, consultez Mises à jour propagées de la configuration de l'environnement Elastic Beanstalk. Pour plus d'informations sur les déploiements de version d'application par lots, consultez Paramètres et stratégies de déploiement.

Pour annuler une mise à jour

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans la page de présentation de l'environnement, choisissez Actions (Actions) puis Abort current operation (Abandonner l’opération en cours).

Reconstruction d'environnements Elastic Beanstalk

Votre AWS Elastic Beanstalk environnement peut devenir inutilisable si vous n'utilisez pas les fonctionnalités d'Elastic Beanstalk pour modifier ou supprimer les ressources sous-jacentes de l'environnement. AWS Si cela se produit, vous pouvez reconstruire l'environnement pour tenter de le restaurer à un état de fonctionnement. La reconstruction d'un environnement met hors service toutes les ressources de celui-ci et les remplace par de nouvelles ressources avec la même configuration.

Vous pouvez également reconstruire des environnements suspendus dans les six semaines (42 jours) suivant leur suspension. Lors de la reconstruction, Elastic Beanstalk tente de créer un nouvel environnement avec le même nom, le même ID et la même configuration.

Reconstruction d'un environnement en cours d'exécution

Vous pouvez reconstruire un environnement via la console Elastic Beanstalk ou à l'aide de l'API `RebuildEnvironment`.

> **Warning**
>
> Si votre environnement possède une base de données couplée, elle sera supprimée lors de la reconstruction, et la nouvelle base de données de l'environnement reconstruit ne contiendra pas les données précédentes. Si vous souhaitez conserver la base de données ou prendre un instantané, assurez-vous que la politique de suppression de la base de données est correctement configurée pour obtenir les résultats souhaités après sa reconstruction. Pour de plus amples informations, veuillez consulter Cycle de vie de base de données.

Pour reconstruire un environnement en cours d'exécution (console)

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions, puis Rebuild environment (Reconstruire l'environnement).
4. Choisissez Reconstruire.

Pour reconstruire un environnement d'exécution avec l'API Elastic Beanstalk, `RebuildEnvironment`utilisez l'action AWS CLI avec le ou le SDK. AWS

```
$ aws elasticbeanstalk rebuild-environment --environment-id e-vdnftxubwq
```

Reconstruction d'un environnement suspendu

Vous pouvez reconstruire et restaurer un environnement arrêté à l'aide de la console Elastic Beanstalk, de l'interface de ligne de commande EB ou de l'API `RebuildEnvironment`.

> **Note**
>
> Si vous n'utilisez pas votre propre nom de domaine personnalisé avec votre environnement suspendu, l'environnement utilise un sous-domaine d'elasticbeanstalk.com. Ces sous-

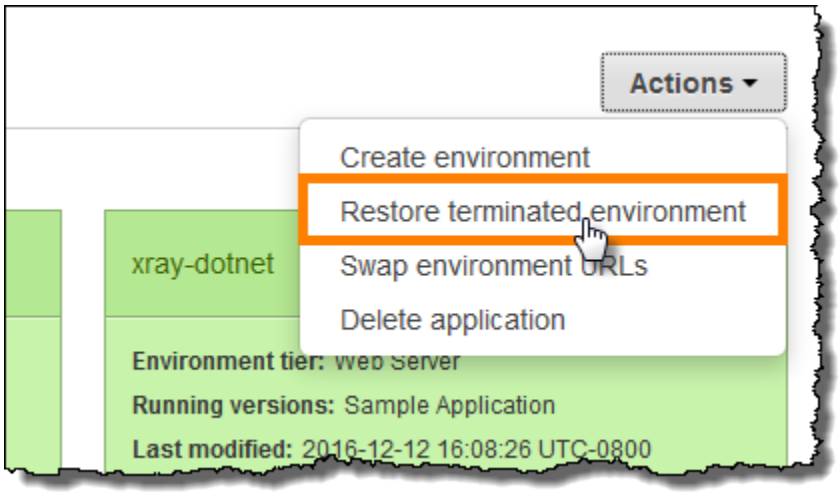
domaines sont partagés au sein d'une région Elastic Beanstalk. Ils peuvent donc être utilisés par un environnement créé par un client dans la même région. Si votre environnement a été suspendu, un autre environnement peut utiliser son sous-domaine. Dans ce cas, la reconstruction échoue.
Vous pouvez éviter cette erreur en utilisant un domaine personnalisé. Consultez Nom de domaine de votre environnement Elastic Beanstalk pour plus de détails.

Les environnements récemment fermés apparaissent dans l'aperçu des applications pendant une heure maximum. Au cours de cette période, vous pouvez afficher les événements de l'environnement dans le tableau de bord de celui-ci et utiliser l'action Restore environment (Restaurer l'environnement) pour le reconstruire.

Pour reconstruire un environnement qui n'est plus visible, utilisez l'option Restore terminated environment (Restaurer l'environnement résilié) depuis la page d'application.

Pour reconstruire un environnement suspendu (console)

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.
3. Choisissez Actions, puis Restore terminated environment (Restaurer l'environnement résilié).



4. Choisissez un environnement suspendu.
5. Choisissez Restore (Restaurer).

Elastic Beanstalk tente de créer un nouvel environnement avec le même nom, le même ID et la même configuration. S'il existe un environnement avec le même nom ou la même URL lorsque vous tentez de reconstruire, la reconstruction échoue. La suppression de la version d'application qui avait été déployée dans l'environnement entraîne également l'échec de la reconstruction.

Si vous utilisez l'interface de ligne de commande EB pour gérer votre environnement, exécutez la commande eb restore pour reconstruire un environnement suspendu.

```
$ eb restore e-vdnftxubwq
```

Pour plus d'informations, consultez eb restore.

Pour reconstruire un environnement terminé avec l'API Elastic Beanstalk, `RebuildEnvironment`utilisez l'action AWS CLI avec le ou le SDK. AWS

```
$ aws elasticbeanstalk rebuild-environment --environment-id e-vdnftxubwq
```

Types d'environnement

Dans AWS Elastic Beanstalk, vous pouvez créer un environnement évolutif et équilibré en charge ou un environnement à instance unique. Le type d'environnement dont vous avez besoin dépend de l'application que vous déployez. Par exemple, vous pouvez développer et tester une application dans un environnement instance unique pour économiser de l'argent, mettre à niveau cet environnement vers un environnement évolutif, d'équilibrage de charge lorsque l'application est prête pour la production.

> **Note**
>
> Un niveau d'environnement de travail pour une application web qui traite des tâches en arrière-plan n'inclut pas un équilibreur de charge. Cependant, un environnement de travail monte efficacement en puissance en ajoutant des instances au groupe Auto Scaling pour traiter les données à partir de la file d'attente Amazon SQS lorsque la charge le nécessite.

Environnement évolutif et équilibré en charge

Un environnement évolutif et équilibré en charge utilise les services Elastic Load Balancing et Amazon EC2 Auto Scaling pour fournir les EC2 instances Amazon requises pour votre application déployée. Amazon EC2 Auto Scaling démarre automatiquement des instances supplémentaires pour s'adapter à la charge croissante de votre application. Si la charge de votre application diminue, Amazon EC2 Auto Scaling arrête les instances mais laisse toujours fonctionner le nombre minimum d'instances que vous avez spécifié. Si votre application a besoin d'une évolutivité avec l'option d'exécution dans plusieurs zones de disponibilité, utilisez un environnement évolutif et d'équilibrage de charge. Si vous ne savez pas quel type d'environnement sélectionner, vous pouvez en choisir un et, si nécessaire, changer le type d'environnement plus tard.

Environnement à instance unique

Un environnement à instance unique contient une EC2 instance Amazon dotée d'une adresse IP élastique. Un environnement instance unique n'a pas d'équilibreur de charge, ce qui peut vous aider à réduire les coûts par rapport à un environnement évolutif et d'équilibrage de charge. Bien qu'un environnement à instance unique utilise le service Amazon EC2 Auto Scaling, les paramètres relatifs au nombre minimum d'instances, au nombre maximum d'instances et à la capacité souhaitée sont tous définis sur 1. Par conséquent, de nouvelles instances ne sont pas démarrées pour adapter la charge croissante sur votre application.

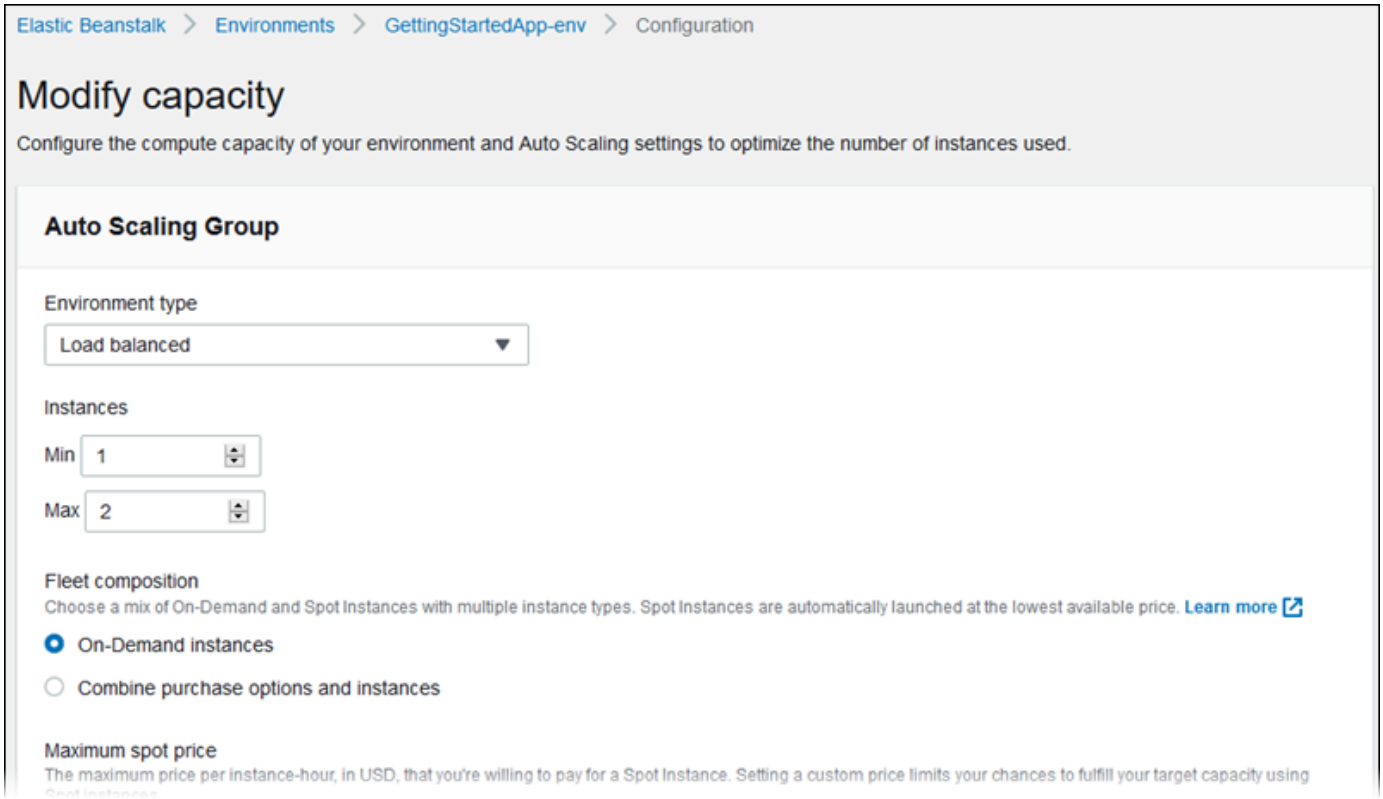
Utilisez un environnement instance unique si vous attendez un faible trafic de votre application de production ou si vous effectuez un développement à distance. Si vous ne savez pas quel type d'environnement sélectionner, vous pouvez en choisir un et, si nécessaire, changer le type d'environnement plus tard. Pour de plus amples informations, veuillez consulter Changement de type d'environnement.

Changement de type d'environnement

Vous pouvez remplacer votre type d'environnement par un environnement instance unique ou un environnement évolutif, d'équilibrage de charge, en modifiant la configuration de votre environnement. Dans certains cas, vous souhaiterez peut-être modifier votre type d'environnement d'un type à un autre. Imaginons par exemple que vous avez développé et testé une application dans un environnement instance unique afin d'économiser de l'argent. Lorsque votre application est prête pour la production, vous pouvez remplacer le type d'environnement par un environnement évolutif, d'équilibrage de charge afin qu'il puisse évoluer pour répondre aux exigences de vos clients.

Pour modifier le type d'un environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la liste Type d'environnement, sélectionnez un type d'environnement.



6. Choisissez Enregistrer.

   La mise à jour de l'environnement peut prendre plusieurs minutes pendant qu'Elastic Beanstalk fournit des ressources AWS .

Si votre environnement se trouve dans un VPC, sélectionnez les sous-réseaux dans lesquels placer les instances Elastic Load Balancing et Amazon EC2 . Chaque zone de disponibilité dans laquelle votre application s'exécute doit avoir les deux. Consultez Utilisation d'Elastic Beanstalk avec Amazon VPC pour plus de détails.

Environnements de travail Elastic Beanstalk

Si votre AWS Elastic Beanstalk application exécute des opérations ou des flux de travail longs, vous pouvez décharger ces tâches vers un environnement de travail dédié. Le découplage de l'élément frontal de votre application web d'un processus qui effectue des opérations de blocage est une façon courante de garantir que votre application reste réactive sous charge.

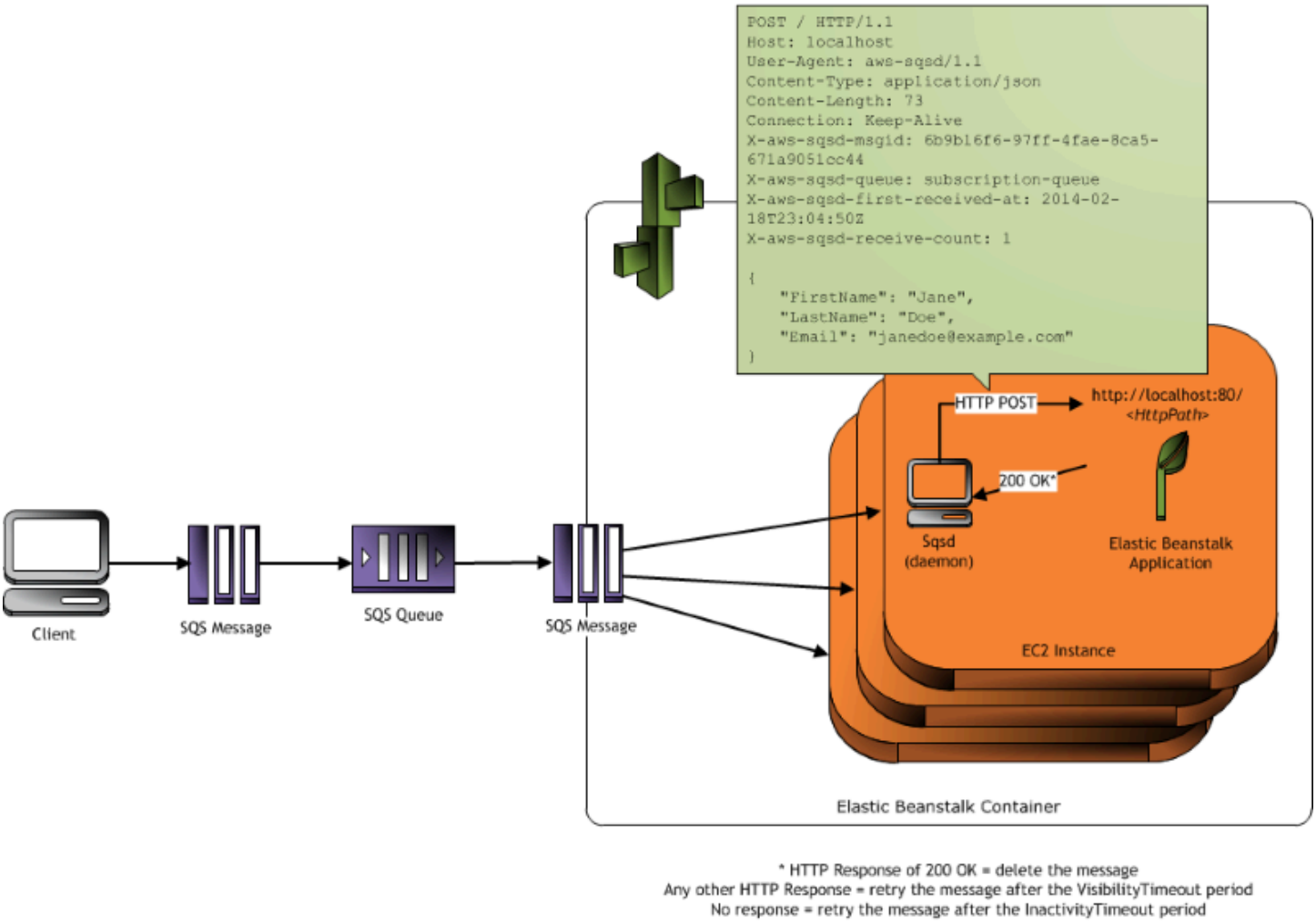
Une tâche à long terme est tout ce qui accroît considérablement le temps nécessaire pour effectuer une demande, tel que le traitement d'images ou de vidéos, l'envoi d'e-mails ou la génération d'une archive ZIP. Ces opérations peuvent ne prendre qu'une ou deux secondes, mais un délai de quelques secondes est beaucoup pour une requête web qui aboutirait sinon en moins de 500 ms.

Une option consiste à générer dynamiquement un processus de travail localement, à renvoyer une réussite et à traiter la tâche de manière asynchrone. Cela fonctionne si votre instance peut faire face à toutes les tâches qui lui sont envoyées. Sous une charge élevée, toutefois, une instance peut être submergée par des tâches en arrière-plan et arrêter de répondre à des demandes de priorité plus élevée. Si des utilisateurs peuvent générer plusieurs tâches, l'augmentation de la charge peut ne pas correspondre à une augmentation des utilisateurs, ce qui complique l'augmentation de la taille de votre niveau de serveur web efficacement.

Pour éviter d'exécuter des tâches de longue durée localement, vous pouvez utiliser le AWS SDK de votre langage de programmation pour les envoyer vers une file d'attente Amazon Simple Queue Service (Amazon SQS) et exécuter le processus qui les exécute sur un ensemble d'instances distinct. Les instances de travail prennent des éléments de la file d'attente uniquement lorsqu'elles ont la capacité nécessaire pour les exécuter, ce qui leur évite d'être submergées.

Les environnements de travail Elastic Beanstalk simplifient ce processus en gérant la file d'attente Amazon SQS et en exécutant un processus démon sur chaque instance qui lit automatiquement

depuis la file d'attente. Lorsque le démon extrait un élément de la file d'attente, il envoie une demande HTTP POST localement à `http://localhost/` sur le port 80 avec le contenu du message de file d'attente dans le corps. Tout ce que votre application doit faire consiste à effectuer la tâche longue durée en réponse au POST. Vous pouvez configurer le démon pour publier vers un chemin d'accès différent, utiliser un type MIME autre qu'application/JSON, vous connecter à une file d'attente existante ou personnaliser des connexions (nombre maximal de demandes simultanées), des expirations et des nouvelles tentatives de connexion.



Avec des tâches périodiques, vous pouvez également configurer le démon de travail pour mettre en attente les messages en fonction d'une planification CRON. Chaque tâche périodique peut PUBLIER vers un chemin différent. Activez des tâches périodiques en incluant un fichier YAML dans votre code source qui définit le calendrier et le chemin d'accès pour chaque tâche.

> **Note**
>
> La plateforme .NET sous Windows Server ne prend pas en charge les environnements de travail.

Sections

- Démon SQS d'environnement de travail
- Files d’attente de lettres mortes
- Tâches périodiques
- Utilisez Amazon CloudWatch pour le dimensionnement automatique dans les différents niveaux de l'environnement de travail
- Configuration des environnements de travail

Démon SQS d'environnement de travail

Les environnements de travail exécutent un processus démon fourni par Elastic Beanstalk. Ce démon est mis à jour régulièrement pour ajouter des fonctionnalités et résoudre des bogues. Pour obtenir la dernière version du démon, effectuez la mise à jour vers la version de la plateforme la plus récente.

Lorsque l'application dans l'environnement de travail renvoie une réponse `200 OK` pour reconnaître qu'elle a reçu et traité avec succès la demande, le démon envoie un appel `DeleteMessage` à la file d'attente Amazon SQS afin que le message soit supprimé de la file d'attente. Si l'application renvoie une réponse autre que `200 OK`, Elastic Beanstalk attend l'expiration de la période `ErrorVisibilityTimeout` configurée pour remettre le message dans la file d'attente. S'il n'y a pas de réponse, Elastic Beanstalk attend pour remettre le message dans la file d'attente après la période `InactivityTimeout` afin que le message soit disponible pour une autre tentative au moment du traitement.

> **Note**
>
> Les propriétés des files d'attente Amazon SQS (ordre des messages, at-least-once livraison et échantillonnage des messages) peuvent affecter la façon dont vous concevez une application Web pour un environnement de travail. Pour plus amples informations, veuillez

consulter à Propriétés des files d'attente distribuées dans le Guide du développeur Amazon Simple Queue Service.

Amazon SQS supprime automatiquement les messages qui ont été dans une file d'attente pendant une durée supérieure à la `RetentionPeriod` configurée.

Le démon définit les en-têtes HTTP suivants.

En-têtes HTTP

Nom	Valeur

`User-Agent`	`aws-sqsd`
	
	`aws-sqsd/1.1 1`

`X-Aws-Sqsd-Msgid`	ID de message SQS, utilisé pour détecter des tempête de messages (un nombre anormalem ent élevé de nouveaux messages).
`X-Aws-Sqsd-Queue`	Nom de la file d'attente SQS.
`X-Aws-Sqsd-First-Received-At`	Le temps UTC, au format ISO 8601, lorsque le message a été reçu pour la première fois.
`X-Aws-Sqsd-Receive-Count`	Nombre de message SQS reçus.
`X-Aws-Sqsd-Attr-` *`message-attribute-name`*	Attributs de message personnalisé attribués au message en cours de traitement. Le `message-attribute-name` est le nom d'attribut de message réel. Tous les attributs de message de numéro et de chaîne sont ajoutés à l'en-tête. Les attributs binaires sont ignorés et ils ne sont pas inclus dans l'en-tête.
`Content-Type`	Configuration de type MIME ; par défaut, `application/json` .

Files d’attente de lettres mortes

Les environnements de travail Elastic Beanstalk prennent en charge les files d'attente de lettres mortes Amazon Simple Queue Service (Amazon SQS). Une file d'attente de lettres mortes est une file d'attente où les autres files d'attente (source) peuvent envoyer des messages qui, pour une raison quelconque, n'ont pas pu être traités avec succès. Un avantage principal de l'utilisation d'une file d'attente de lettres mortes est la possibilité d'isoler les messages dont le traitement a échoué. Vous pouvez ensuite analyser les messages envoyés à la file d'attente de lettres mortes pour tenter de déterminer pourquoi leur traitement a échoué.

Une file d'attente de lettres mortes est activée par défaut pour un environnement de travail si vous spécifiez une file d'attente Amazon SQS générée automatiquement au moment de la création de votre couche d'environnement de travail. Si vous sélectionnez une file d'attente SQS existante pour votre environnement de travail, vous devez utiliser SQS pour configurer une file d'attente de lettres mortes de façon indépendante. Pour de plus amples informations sur la manière d'utiliser SQS pour configurer une file d'attente de lettres mortes, veuillez consulter Utilisation des files d'attente de lettres mortes Amazon SQS.

Vous ne pouvez pas désactiver les files d'attente de lettres mortes. Les messages qui ne peuvent pas être diffusés sont toujours envoyés à terme vers une file d'attente de lettres mortes. Vous pouvez, toutefois, désactivez effectivement cette fonctionnalité en définissant l'option `MaxRetries` à la valeur valide maximale de 100.

Si une file d'attente de lettres mortes n'est pas configurée pour la file d'attente Amazon SQS de votre environnement de travail, Amazon SQS conserve les messages dans la file d'attente jusqu'à l'expiration de la période de rétention. Pour de plus amples informations sur la configuration de la période de rétention, veuillez consulter the section called “Configuration des environnements de travail”.

> **Note**
>
> L'option Elastic Beanstalk `MaxRetries` est équivalente à l'option SQS `MaxReceiveCount`. Si votre environnement de travail n'utilise pas une file d'attente SQS générée automatiquement, utilisez l'option `MaxReceiveCount` dans SQS pour désactiver efficacement votre file d'attente de lettres mortes. Pour de plus amples informations, veuillez consulter Utilisation des files d'attente de lettres mortes Amazon SQS.

Pour de plus amples informations sur le cycle de vie d'un message SQS, veuillez consulter Cycle de vie des messages.

Tâches périodiques

Vous pouvez définir des tâches périodiques dans un fichier nommé `cron.yaml` dans votre groupe source pour ajouter des tâches à la file d'attente de votre environnement de travail automatiquement à intervalles réguliers.

Par exemple, le fichier `cron.yaml` suivant crée deux tâches périodiques. La première s'exécute toutes les 12 heures et la seconde à 23 h UTC tous les jours.

Example cron.yaml

```
version: 1
cron:
 - name: "backup-job"
   url: "/backup"
   schedule: "0 */12 * * *"
 - name: "audit"
   url: "/audit"
   schedule: "0 23 * * *"
```

Le **`name`** doit être unique pour chaque tâche. L'URL est le chemin d'accès auquel la demande POST est envoyée pour déclencher la tâche. Le programme est une expression CRON qui détermine le moment d'exécution de la tâche.

Lorsqu'une tâche s'exécute, le démon publie un message dans la file d'attente SQS de l'environnement avec un en-tête indiquant la tâche qui doit être effectuée. Toute instance de l'environnement peut prélever le message et traiter la tâche.

> **Note**
>
> Si vous configurez votre environnement de travail avec une file d'attente SQS et que vous choisissez une file d'attente FIFO Amazon SQS, les tâches périodiques ne sont pas pris en charge.

Elastic Beanstalk utilise le choix principal pour déterminer quelle instance dans votre environnement de travail met en attente la tâche périodique. Chaque instance tente de devenir leader en écrivant

dans une table Amazon DynamoDB. La première instance qui réussit est la principale et elle doit continuer à écrire dans la table pour conserver l'état de principale. Si la principale devient hors service, une autre instance prend rapidement sa place.

Pour les tâches périodiques, le démon de travail définit les en-têtes supplémentaires suivants.

En-têtes HTTP

Nom	Valeur
`X-Aws-Sqsd-Taskname`	Pour les tâches périodiques, le nom de la tâche à effectuer.
`X-Aws-Sqsd-Scheduled-At`	Heure à laquelle la tâche périodique a été planifiée
`X-Aws-Sqsd-Sender-Id`	AWS numéro de compte de l'expéditeur du message

Utilisez Amazon CloudWatch pour le dimensionnement automatique dans les différents niveaux de l'environnement de travail

Ensemble, Amazon EC2 Auto Scaling CloudWatch surveille l'utilisation du processeur des instances en cours d'exécution dans l'environnement de travail. La manière dont vous configurez la limite de mise à l'échelle automatique pour la capacité de l'UC détermine le nombre d'instances que le groupe Auto Scaling exécute pour gérer adéquatement le débit de messages dans la file d'attente Amazon SQS. Chaque instance EC2 publie ses mesures d'utilisation du processeur sur CloudWatch. Amazon EC2 Auto Scaling extrait l'utilisation moyenne CloudWatch du processeur sur toutes les instances de l'environnement de travail. Vous configurez les seuils supérieur et inférieur, ainsi que le nombre d'instances à ajouter ou à résilier, selon la capacité de l'UC. Quand Amazon EC2 Auto Scaling détecte que vous avez atteint le seuil supérieur spécifié sur la capacité de l'UC, Elastic Beanstalk crée les instances dans l'environnement de travail. Les instances sont supprimées quand la charge de l'UC est inférieure au seuil.

> **Note**
>
> Les messages qui n'ont pas été traités au moment de la résiliation d'une instance sont renvoyés à la file d'attente où ils peuvent être traités par un autre démon sur une instance qui est encore en cours d'exécution.

Vous pouvez également définir d'autres CloudWatch alarmes, selon vos besoins, à l'aide de la console Elastic Beanstalk, de la CLI ou du fichier d'options. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon CloudWatch et Création d'un groupe Auto Scaling avec des stratégies de mise à l'échelle par étape.

Configuration des environnements de travail

Vous pouvez gérer la configuration d'un environnement de travail en modifiant la catégorie Worker (Environnement de travail) sur la page Configuration dans la console de gestion de l'environnement.

Elastic Beanstalk > Environments > GettingStartedApp-env > Configuration

Modify worker

You can create a new Amazon SQS queue for your worker application or pull work items from an existing queue. The worker daemon on the instances in your environment pulls an item from the queue and relays it in the body of a POST request to a local HTTP path relative to localhost.

Queue

Worker queue

Autogenerated queue

SQS queue from which to read work items.

Messages

HTTP path

/

The daemon pulls items from the Amazon SQS queue and posts them locally to this path.

MIME type

application/json

Change the MIME type of the POST requests that the worker daemon sends to your application.

HTTP connections

50

Maximum number of concurrent connections to the application.

Visibility timeout

300 seconds

The amount of time to lock an incoming message for processing before returning it to the queue.

Error visibility timeout

seconds

The amount of time to wait before resending a message after an error response from the application.

Advanced options

The following settings control advanced behavior of the worker tier daemon. Learn more

Max retries

10

Maximum number of retries after which the message is discarded.

Connection timeout

5

Inactivity timeout

> **Note**
>
> Vous pouvez configurer le chemin d'URL pour la publication des messages de file d'attente de travail, mais vous ne pouvez pas configurer le port IP. Elastic Beanstalk publie toujours les messages de file d'attente de travail sur le port 80. L'application de l'environnement de travail ou son proxy doit être à l'écoute sur le port 80.

Pour configurer le démon de travail

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration de l'Worker (Environnement de travail, choisissez Edit (Modifier).

La page de configuration Modify worker (Modifier l'environnement de travail) propose les options suivantes.

Dans la section Queue (File d'attente) :

- Worker queue (File d'attente de travail) : spécifiez la file d'attente Amazon SQS à partir de laquelle le démon lit. Vous pouvez choisir une file d'attente existante, si vous en avez une. Si vous choisissez Autogenerated queue (File d'attente générée automatiquement), Elastic Beanstalk crée une nouvelle file d'attente Amazon SQS et une Worker queue URL (URL de file d'attente de travail)correspondante.

  > **Note**
  >
  > Lorsque vous choisissez File d'attente générée automatiquement, la file d'attente créée par Elastic Beanstalk est une file d'attente Amazon SQS standard . Lorsque vous choisissez une file d'attente existante, vous pouvez fournir une file standard ou une file d'attente Amazon SQS FIFO. Sachez que si vous spécifiez une file d'attente FIFO, les tâches périodiques ne sont pas prises en charge.

- Worker queue URL (URL de la file d'attente de travail) : si vous choisissez une Worker queue (File d'attente de travail) existante, ce paramètre affiche l'URL associée à cette file d'attente Amazon SQS.

Dans la section Messages :

- HTTP path (Chemin HTTP) : spécifiez le chemin d'accès relatif à l'application qui reçoit les données à partir de la file d'attente Amazon SQS. Les données sont insérées dans le corps du message d'un message HTTP POST. La valeur par défaut est `/`.
- MIME type (Type MIME) : indiquez le type MIME que le message HTTP POST utilise. La valeur par défaut est `application/json`. Cependant, n'importe quelle valeur est valide car vous pouvez créer, puis spécifier votre propre type MIME.
- HTTP Connections (Connexions HTTP) : spécifiez le nombre maximal de connexions simultanées que le démon peut réaliser sur toute application au sein d'une instance Amazon EC2. La valeur par défaut est **`50`**. Vous pouvez spécifier une valeur de **`1`** à **`100`**.
- Visibility timeout (Délai de visibilité) : indiquez la durée, en secondes, pendant laquelle un message entrant à partir de la file d'attente Amazon SQS est verrouillé pour traitement. Une fois que la durée configurée est écoulée, le message est à nouveau rendu visible dans la file d'attente permettant à un autre démon de le lire. Choisissez une valeur supérieure au délai estimé nécessaire à votre application pour traiter des messages, jusqu'à **`43200`** secondes.
- Error visibility timeout (Délai de visibilité de l'erreur) : indiquez la durée, en secondes, qui s'écoule avant qu'Elastic Beanstalk renvoie un message à la file d'attente Amazon SQS après qu'une tentative de le traiter a échoué avec une erreur explicite. Vous pouvez spécifier une valeur de **`0`** à **`43200`** secondes.

Dans la section Advanced options (Options avancées) :

- Max retries (Nombre maximal de nouvelles tentatives) : spécifiez le nombre maximal de fois qu'Elastic Beanstalk tente d'envoyer le message à la file d'attente Amazon SQS avant de déplacer le message dans la file d'attente des lettres mortes. La valeur par défaut est **`10`**. Vous pouvez spécifier une valeur de **`1`** à **`100`**.

> **Note**
>
> Le Max retries (Nombre maximum de nouvelles tentatives) ne s'applique qu'aux files d'attente Amazon SQS configurées avec une file d'attente de lettres mortes. Pour toutes les

files d'attente Amazon SQS qui ne sont pas configurées avec une file d'attente de lettres mortes, Amazon SQS conserve les messages dans la file d'attente et les traite jusqu'à l'expiration de la période spécifiée par l'option Retention period (Période de conservation).

- Connection timeout (Délai de connexion) : indiquez la durée, en secondes, d'attente de connexions réussies à une application. La valeur par défaut est **5**. Vous pouvez spécifier une valeur de **1** à **60** secondes.
- Inactivity timeout (Délai d'inactivité) : indiquez la durée, en secondes, d'attente d'une réponse sur une connexion existante à une application. La valeur par défaut est **180**. Vous pouvez spécifier une valeur de **1** à **36000** secondes.
- Retention period (Période de conservation) : indiquez la durée, en secondes, de validité et de traitement actif d'un message. La valeur par défaut est **345600**. Vous pouvez spécifier une valeur de **60** à **1209600** secondes.

Si vous utilisez une file d'attente Amazon SQS existante, les paramètres que vous configurez lorsque vous créez un environnement de travail peuvent entrent en conflit avec les paramètres que vous avez configurés directement dans Amazon SQS. Par exemple, si vous configurez un environnement de travail avec une valeur `RetentionPeriod` qui est supérieure à la valeur `MessageRetentionPeriod` que vous avez définie dans Amazon SQS, Amazon SQS supprime le message quand il dépasse la valeur `MessageRetentionPeriod`.

À l'inverse, si la valeur `RetentionPeriod` que vous configurez dans les paramètres d'environnement de travail est inférieure à la valeur `MessageRetentionPeriod` que vous définissez dans Amazon SQS, alors le démon supprime le message avant qu'Amazon SQS puisse le faire. Pour `VisibilityTimeout`, la valeur que vous configurez pour le démon dans les paramètres d'environnement de travail remplace la valeur Amazon SQS `VisibilityTimeout`. Assurez-vous que les messages sont supprimés correctement en comparant vos paramètres Elastic Beanstalk à vos paramètres Amazon SQS.

Création de liens entre les environnements Elastic Beanstalk

A mesure que votre application se développe en taille et en complexité, vous pourrez souhaiter la diviser en composants ayant différents cycle de vie opérationnels et de développement. En exécutant des services plus petits qui interagissent les uns avec les autres via une interface bien définie, les équipes peuvent travailler de manière indépendante et réduire les risques lors des déploiements.

AWS Elastic Beanstalk vous permet de lier vos environnements pour partager des informations entre les composants qui dépendent les uns des autres.

> **Note**
>
> Elastic Beanstalk prend en charge actuellement des liens d'environnement pour toutes les plateformes, sauf Multicontainer Docker.

Les liens d'environnement vous permettent de spécifier les connexions entre les environnements de composants de votre application en tant que références désignées. Lorsque vous créez un environnement qui définit un lien, Elastic Beanstalk définit une variable d'environnement portant le même nom que le lien. La valeur de la variable est le point de terminaison que vous pouvez utiliser pour vous connecter à l'autre composant qui peut être un environnement de travail ou de serveur web.

Par exemple, si votre application est constituée d'un serveur frontal qui collecte des adresses e-mail et d'un serveur de travail qui envoie un e-mail de bienvenue aux adresses de messagerie collectées par le serveur frontal, vous pouvez créer un lien vers le serveur de travail dans votre serveur frontal et permettre à ce dernier de découvrir automatiquement le point de terminaison (URL de file d'attente) pour votre serveur de travail.

Définissez les liens vers d'autres environnements dans un manifeste d'environnement, un fichier formaté YAML nommé `env.yaml` à la racine de la source de votre application. Le manifeste suivant définit un lien vers un serveur de travail nommé de l'environnement :

`~/workspace/my-app/frontend/env.yaml`

```
AWSConfigurationTemplateVersion: 1.1.0.0
EnvironmentLinks:
  "WORKERQUEUE": "worker"
```

Lorsque vous créez un environnement avec une version de l'application qui inclut le manifeste d'environnement ci-dessus, Elastic Beanstalk recherche un environnement nommé `worker` qui appartient à la même application. Si cet environnement existe, Elastic Beanstalk crée une propriété d'environnement nommée `WORKERQUEUE`. La valeur de `WORKERQUEUE` est l'URL de file d'attente Amazon SQS. L'application frontale peut lire cette propriété de la même manière qu'une variable d'environnement. Consultez Manifeste d'environnement (`env.yaml`) pour plus de détails.

Pour utiliser les liens d'environnement, ajoutez un manifeste d'environnement à la source de votre application et chargez-le à l'aide de l'interface de ligne de commande EB AWS CLI ou d'un SDK. Si vous utilisez le AWS CLI ou un SDK, définissez l'`process`indicateur lorsque vous appelez `CreateApplicationVersion` :

```
$ aws elasticbeanstalk create-application-version --process --application-name
 my-app --version-label frontend-v1 --source-bundle S3Bucket="amzn-s3-demo-
bucket",S3Key="front-v1.zip"
```

Cette option indique à Elastic Beanstalk de valider les fichiers de configuration et le manifeste d'environnement de votre groupe source lorsque vous créez la version d'application. Les ensembles de l'interface de ligne de commande EB définissent automatiquement cet indicateur lorsque vous avez un manifeste d'environnement dans le répertoire de votre projet.

Créez vos environnements normalement à l'aide de n'importe quel client. Lorsque vous avez besoin de mettre fin à des environnements, résiliez l'environnement avec le lien en premier. Si un environnement est lié à un autre environnement, Elastic Beanstalk empêche l'arrêt de l'environnement lié. Pour remplacer cette protection, utilisez l'indicateur `ForceTerminate`. Ce paramètre est disponible dans l' AWS CLI en tant que `--force-terminate` :

```
$ aws elasticbeanstalk terminate-environment --force-terminate --environment-name
 worker
```

Restaurer votre environnement Elastic Beanstalk à partir d'un état non valide

Cette rubrique fournit des informations générales et des ressources qui expliquent comment résoudre les problèmes liés à un environnement Elastic Beanstalk dont l'état n'est pas valide.

Corriger l'erreur

Les opérations standard sur un environnement dont l'état n'est pas valide ne se termineront pas correctement. L'opération échouée renverra un message d'erreur contenant le texte suivant :

```
The stack stack_id associated with environment environment-ID is in stack-status state.
```

Pour résoudre cette erreur, consultez l'article du centre de connaissances Pourquoi mon environnement Elastic Beanstalk n'est-il pas valide ? .

> **Note**
>
> Avant le 16 décembre 2024, l'opération défaillante renvoyait l'erreur suivante à la place : `Environment is in an invalid state for this operation. Must be ready.` dans ce cas, vous deviez contacter le AWS Support pour réinitialiser l'état de l'environnement après avoir effectué les actions correctives.
> Aujourd'hui, vous devez toujours résoudre les problèmes de stack en suivant les instructions de l'article référencé du Knowledge Center. Cependant, une fois que vous avez terminé les actions correctives, Elastic Beanstalk met automatiquement à jour le statut de l'environnement de non valide à disponible, et vous pouvez reprendre les opérations standard sur votre environnement sans plus attendre.

Pourquoi l'erreur se produit

Lorsque vous déployez une application dans Elastic Beanstalk, le service crée une pile sous-jacente. AWS CloudFormation Elastic Beanstalk CloudFormation appelle le service pour lancer les ressources dans votre environnement et propager les modifications de configuration.

Si Elastic Beanstalk effectue une opération sur un environnement sans avoir accès à une ressource requise, la pile sous-jacente de l'environnement peut entrer dans un CloudFormation état défaillant. D'autres problèmes peuvent également être à l'origine de cet état, bien que les problèmes d'autorisation en soient la cause principale. En raison de l'état défaillant de la pile, les demandes d'opérations CloudFormation Elastic Beanstalk ne peuvent pas effectuer de nouvelles mises à jour de la pile, ce qui entraîne l'échec des opérations Elastic Beanstalk, telles que et. UpdateEnvironment RetrieveEnvironmentInfo

À ce stade, vous devez d'abord corriger la cause première du problème sous-jacent pour remédier à la CloudFormation situation. Le service Elastic Beanstalk CloudFormation détecte ensuite le changement d'état de la pile et effectue un suivi pour rétablir l'état disponible de votre environnement. À ce stade, d'autres opérations peuvent être effectuées avec succès.

Les problèmes d'autorisation sont généralement à l'origine de cet effet sur la CloudFormation pile et sur l'environnement Elastic Beanstalk out-of-band, bien que les modifications puissent également entraîner des problèmes.

Important

Pour éviter toute perturbation de votre environnement, nous vous recommandons vivement de lancer des opérations de gestion et de configuration de votre environnement uniquement à partir du service Elastic Beanstalk. La modification des ressources à l'aide de la console, des commandes CLI ou du SDK d'un service autre qu'Elastic Beanstalk out-of-band est une modification qui entraîne une dérive des ressources. La dérive des ressources affecte l'état de la CloudFormation pile, ce qui entraîne à son tour un état non valide pour l'environnement Elastic Beanstalk.

Pour plus d'informations sur la dérive des ressources, voir Qu'est-ce que la dérive ? dans le guide de AWS CloudFormation l'utilisateur.

Configuration d'environnements Elastic Beanstalk

Cette rubrique se concentre sur les options de configuration disponibles dans la console Elastic Beanstalk. AWS Elastic Beanstalk propose un large éventail d'options pour personnaliser les ressources de votre environnement, ainsi que le comportement et les paramètres de plateforme d'Elastic Beanstalk.

Les rubriques suivantes montrent comment configurer votre environnement dans la console. Elles décrivent également les espaces de noms sous-jacents qui correspondent aux options de la console à utiliser avec les fichiers de configuration ou les options de configuration d'API. Pour en savoir plus sur les méthodes de configuration avancées, consultez Configuration d'environnements (niveau avancé).

Rubriques

- Ressources provisionnées
- Configuration d'un environnement avec la console Elastic Beanstalk.
- Les EC2 instances Amazon pour votre environnement Elastic Beanstalk
- Auto Scaling de vos instances d'environnement Elastic Beanstalk
- Équilibreur de charge pour votre environnement Elastic Beanstalk
- Ajout d'une base de données à votre environnement Elastic Beanstalk
- La sécurité de votre AWS Elastic Beanstalk environnement
- Balisage des ressources dans vos environnements Elastic Beanstalk
- Variables d'environnement et autres paramètres du logiciel
- Notifications d'environnement Elastic Beanstalk avec Amazon SNS
- Configuration d'Amazon Virtual Private Cloud (Amazon VPC) avec Elastic Beanstalk
- Nom de domaine de votre environnement Elastic Beanstalk

Ressources provisionnées

Lorsque vous créez un environnement de serveur Web, Elastic Beanstalk crée plusieurs ressources pour soutenir le fonctionnement de votre application. Ce chapitre explique comment personnaliser ces ressources pour votre environnement Elastic Beanstalk.

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.
- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Équilibreur de charge – Équilibreur de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibreur de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibreur de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et qui sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *`subdomain`*.*`region`*.elasticbeanstalk.com.

> **Sécurité du domaine**
>
> Pour renforcer la sécurité de vos applications Elastic Beanstalk, le domaine elasticbeanstalk.com est enregistré dans la liste des suffixes publics (PSL).

Si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Elastic Beanstalk, nous vous recommandons d'utiliser des cookies `__Host-` avec un préfixe pour une sécurité accrue. Cette pratique protège votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Configuration d'un environnement avec la console Elastic Beanstalk.

Cette rubrique décrit les options de configuration disponibles via la console Elastic Beanstalk et explique comment naviguer dans les pages de configuration.

Pour afficher un récapitulatif de la configuration de votre environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.

Page de configuration

La page Configuration overview (Présentation de la configuration) affiche un ensemble de catégories de configuration. Chaque catégorie de configuration regroupe un ensemble d'options associées.

Accès au service

Les options de cette catégorie sélectionnent le rôle de service et le profil d'EC2 instance qu'Elastic Beanstalk utilise pour gérer votre environnement. Choisissez éventuellement une paire de EC2 clés pour vous connecter en toute sécurité à vos EC2 instances.

Mise en réseau et base de données

Les options de cette catégorie configurent les paramètres VPC et les sous-réseaux pour les EC2 instances et l'équilibreur de charge de l'environnement. Ils offrent également la possibilité de configurer une base de données Amazon RDS intégrée à votre environnement.

Trafic et dimensionnement des instances

Ces options personnalisent la capacité, le dimensionnement et l'équilibrage de charge pour les EC2 instances de l'environnement. Vous pouvez également configurer Elastic Load Balancing pour capturer des journaux contenant des informations détaillées sur les demandes envoyées à l'équilibreur de charge.

Les options suivantes pour vos EC2 instances sont également disponibles pour la configuration :

- Type, taille, taux d' input/output exploitation (IOPS) et débit du volume racine.
- Activation du service de métadonnées d'instance (IMDS).
- Sélection de groupes de EC2 sécurité pour contrôler le trafic des instances.
- CloudWatch intervalle de surveillance des métriques.
- Intervalle de temps pour l'enregistrement des métriques.

Mises à jour, surveillance et journalisation

Cette catégorie configure les options suivantes :

- Rapports sur la santé environnementale, y compris la possibilité de sélectionner des rapports sanitaires améliorés.
- Mises à jour de plateforme gérées qui définissent quand et comment Elastic Beanstalk déploie les modifications apportées à l'environnement.
- Activation du service X-Ray pour collecter des données sur le comportement de votre application afin d'identifier les problèmes et les opportunités d'optimisation.
- Options spécifiques à la plate-forme, notamment les propriétés du serveur proxy et de l'environnement du système d'exploitation.

Navigation dans la page de configuration

Choisissez Modifier dans une catégorie de configuration pour lancer la page de configuration associée, où vous pouvez voir les valeurs complètes des options et apporter des modifications.

Navigation dans une catégorie de configuration

Naviguez dans une page de catégorie de configuration en effectuant l'une des actions suivantes :

- Annuler — Revenez à la page d'aperçu de la configuration sans appliquer vos modifications de configuration. Lorsque vous choisissez Annuler, toutes les modifications en attente apportées sur les catégories de configuration sont perdues.

  Vous pouvez également annuler vos modifications de configuration en choisissant un autre élément sur la page de navigation de gauche, tel que Events ou Logs.
- Continue (Continuer) : cette option vous redirige vers la page Configuration overview (Présentation de la configuration). Vous pouvez alors continuer à apporter des modifications ou appliquer celles qui sont en attente.
- Appliquer : appliquez les modifications que vous avez apportées dans l'une des catégories de configuration à votre environnement. Dans certains cas, vous êtes invité à confirmer une conséquence de l'une de vos décisions de configuration.

Navigation sur la page d'aperçu de la configuration

Choisissez Modifier dans une catégorie de configuration pour ouvrir une page de configuration associée, où vous pouvez voir les valeurs complètes des options et apporter des modifications. Lorsque vous avez terminé de consulter et de modifier les options, vous pouvez choisir l'une des actions suivantes sur la page d'aperçu de la configuration :

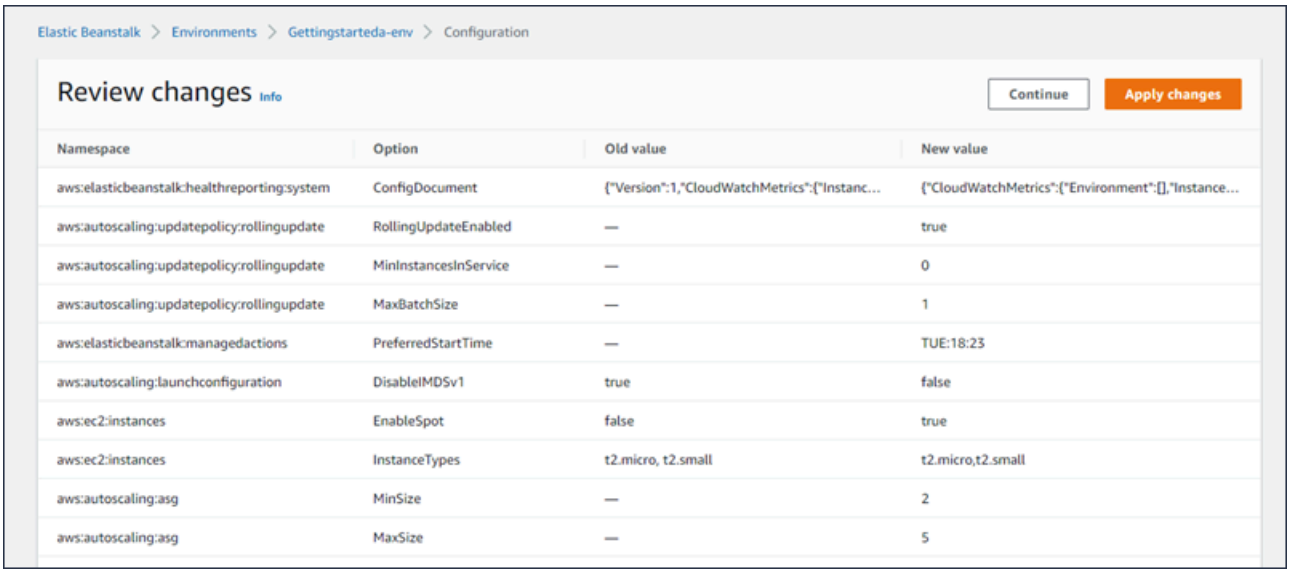
- Cancel (Annuler) : cette option vous redirige vers le tableau de bord de l'environnement sans appliquer les modifications de configuration. Lorsque vous choisissez Annuler, toutes les modifications en attente apportées sur les catégories de configuration sont perdues.

  Vous pouvez également annuler vos modifications de configuration en choisissant un autre élément sur la page de navigation de gauche, tel que Events ou Logs.
- Review changes (Vérifier les modifications) : cette option vous permet d'obtenir un résumé de toutes les modifications en attente que vous avez effectuées dans les catégories de configuration. Pour en savoir plus, consultez Page de vérification des modifications.
- Apply changes(Appliquer les modifications) : cette option vous permet d'appliquer à votre environnement les modifications que vous avez effectuées dans les catégories de configuration. Dans certains cas, vous êtes invité à confirmer une conséquence de l'une de vos décisions de configuration.

Page de vérification des modifications

La page Review Changes (Vérifier les modifications) affiche un tableau répertoriant toutes les modifications d'options que vous avez effectuées dans les catégories de configuration et que vous n'avez pas encore appliquées à l'environnement.

Les tableaux répertorient chaque option en combinant les valeurs Namespace (Espace de noms) et Option Name (Nom de l'option) qu'Elastic Beanstalk utilise pour identifier les options. Pour en savoir plus, consultez Options de configuration.



Une fois que vous avez vérifié les modifications, vous pouvez choisir l'une des actions suivantes :

- Continue (Continuer) : cette option vous redirige vers la page Configuration overview (Présentation de la configuration). Vous pouvez alors continuer à apporter des modifications ou appliquer celles qui sont en attente.
- Apply configuration (Appliquer une configuration) : cette option vous permet d'appliquer à votre environnement les modifications que vous avez effectuées dans les catégories de configuration. Dans certains cas, vous êtes invité à confirmer une conséquence de l'une de vos décisions de configuration.

Les EC2 instances Amazon pour votre environnement Elastic Beanstalk

Cette rubrique décrit les EC2 instances Amazon et les options de configuration qui affectent votre environnement Elastic Beanstalk.

Lorsque vous créez un environnement de serveur Web, vous AWS Elastic Beanstalk créez une ou plusieurs machines virtuelles Amazon Elastic Compute Cloud (Amazon EC2), appelées instances.

Les instances de votre environnement sont configurées pour exécuter des applications web sur la plateforme que vous choisissez. Vous pouvez apporter des modifications aux propriétés et aux comportements des instances de votre environnement lorsque vous créez votre environnement, ou même pendant son exécution. Vous pouvez également déjà effectuer ces modifications en modifiant le code source que vous déployez dans l'environnement. Pour plus d'informations, consultez the section called “Options de configuration”.

> **Note**
>
> Le groupe Auto Scaling de votre environnement gère les EC2 instances Amazon qui exécutent votre application. Lorsque vous apportez des modifications de configuration décrites dans cette rubrique, la configuration de lancement change également. La configuration de lancement est soit un modèle de EC2 lancement Amazon, soit une ressource de configuration de lancement de groupe Auto Scaling. Ce changement nécessite le remplacement de toutes les instances. Il déclenche également soit une mise à jour propagée, soit une mise à jour immuable, en fonction du type de mise à jour configuré.

EC2 options d'achat d'instances

Elastic Beanstalk propose EC2 plusieurs options d'achat d'instances Amazon :

- À la demande : une instance à la demande est une pay-as-you-go ressource. Aucun engagement à long terme n'est requis lorsque vous l'utilisez.
- Réservée — Une instance réservée est un discount de facturation préacheté appliqué automatiquement aux instances à la demande correspondantes dans votre environnement.
- Spot — Une instance Spot est une EC2 instance Amazon inutilisée qui est disponible à un prix inférieur au prix à la demande. Vous pouvez activer et configurer l'allocation d'instances Spot dans votre environnement. Pour de plus amples informations, veuillez consulter Auto Scaling de vos instances d'environnement Elastic Beanstalk.

Rubriques

- Types d' EC2 instances Amazon
- Configuration des EC2 instances Amazon à l'aide de la console Elastic Beanstalk

- Gestion des groupes de sécurité EC2
- Configuration des groupes de sécurité et des types d'instances Amazon EC2 à l'aide du AWS CLI
- Configuration des EC2 instances Amazon avec des options d'espace de noms
- Configuration de l'IMDS sur les instances de votre environnement Elastic Beanstalk

Types d' EC2 instances Amazon

Cette rubrique explique le terme type d'instance. Lorsque vous créez un nouvel environnement, Elastic Beanstalk fournit des instances EC2 Amazon en fonction des types d'instances Amazon EC2 que vous choisissez. Les types d'instances que vous choisissez déterminent le matériel hôte qui exécute vos instances. EC2 les types d'instances peuvent être classés en fonction de l'architecture de processeur sur laquelle chacun est basé. Elastic Beanstalk prend en charge les types d'instances basés sur les AWS architectures de processeur suivantes : architecture Arm 64 bits de Graviton (arm64), architecture 64 bits (x86) et architecture 32 bits (i386). Elastic Beanstalk sélectionne l'architecture du processeur x86 par défaut lorsque vous créez un nouvel environnement.

> **Note**
>
> L'architecture 32 bits i386 n'est plus prise en charge par la majorité des plateformes Elastic Beanstalk. Nous vous recommandons de choisir plutôt les types d'architecture x86 ou arm64. Elastic Beanstalk fournit des options de configuration pour les types d'instance de processeur i386 dans l'espace de noms `aws:ec2:instances`.

Tous les types d'instance de la configuration d'un environnement Elastic Beanstalk donné doivent avoir le même type d'architecture de processeur. Supposons que vous ajoutiez un nouveau type d'instance à un environnement existant qui possède déjà un type d'instance t2.medium, basé sur l'architecture x86. Vous ne pouvez ajouter qu'un autre type d'instance de la même architecture, tel que t2.small. Si vous souhaitez remplacer les types d'instance existants par ceux issus d'une architecture différente, vous pouvez le faire. Mais assurez-vous que tous les types d'instances de la commande sont basés sur le même type d'architecture.

Elastic Beanstalk ajoute régulièrement la prise en charge de nouveaux types d'instances compatibles après leur introduction par Amazon. EC2 Pour plus d'informations sur les types d'instances disponibles, consultez la section Types d'instances dans le guide de EC2 l'utilisateur Amazon.

> **Note**
>
> Elastic Beanstalk prend désormais en charge Graviton sur toutes les dernières plateformes Amazon Linux 2 dans toutes les régions prises en charge par Graviton. AWS Pour plus d'informations sur la création d'un environnement Elastic Beanstalk avec des types d'instance basés sur arm64, consultez Configuration des EC2 instances Amazon à l'aide de la console Elastic Beanstalk.
> Créez de nouveaux environnements qui exécutent des EC2 instances Amazon sur l'architecture arm64 et migrez vos applications existantes vers ces environnements grâce aux options de déploiement d'Elastic Beanstalk.
> Pour en savoir plus sur les processeurs basés sur Graviton arm64, consultez les ressources suivantes AWS :
>
> - Avantages — Le processeur AWS Graviton
> - Mise en route et autres sujets, tels que les considérations spécifiques à la langue — Article sur les premiers pas avec Graviton AWS GitHub

Configuration des EC2 instances Amazon à l'aide de la console Elastic Beanstalk

Vous pouvez créer ou modifier la configuration de l'instance Amazon de votre environnement Elastic Beanstalk dans EC2 la console Elastic Beanstalk.

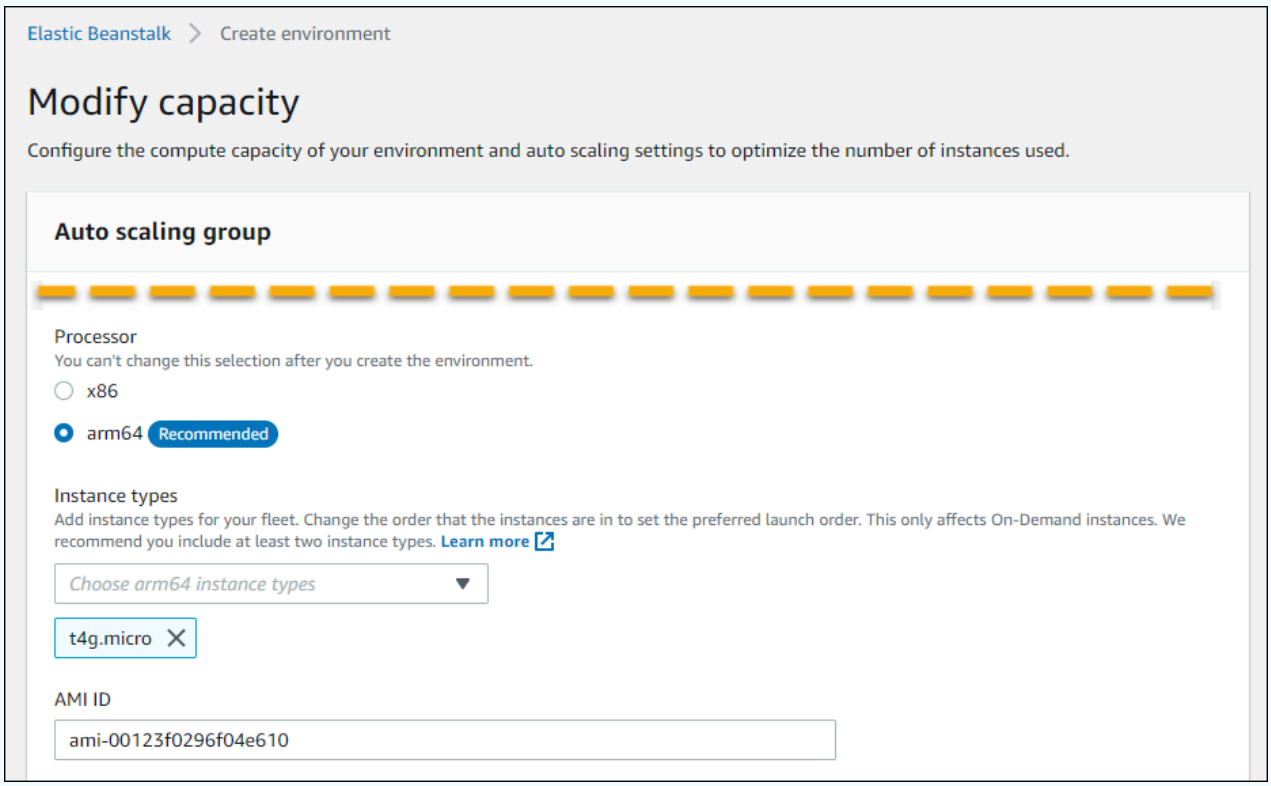
> **Note**
>
> Bien que la console Elastic Beanstalk ne propose pas la possibilité de modifier l'architecture du processeur d'un environnement existant, vous pouvez le faire avec le. AWS CLI Pour des exemples de commandes, voirConfiguration des groupes de sécurité et des types d'instances Amazon EC2 à l'aide du AWS CLI.

Pour configurer les EC2 instances Amazon dans la console Elastic Beanstalk lors de la création de l'environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS

2. Dans le panneau de navigation, choisissez Environments (Environnements).
3. Choisissez Create a new environment (Créer un nouvel environnement) pour commencer à créer votre environnement.
4. Sur la page principale de l'assistant, avant de choisir Créer un environnement, choisissez Configurer plus d'options.
5. Dans la catégorie de configuration Instances, choisissez Edit (Modifier). Modifiez les paramètres de cette catégorie, puis choisissez Apply (Appliquer). Pour les descriptions de paramètre, consultez la section the section called “Paramètres de catégorie d'instances” de cette page.
6. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier). Modifiez les paramètres de cette catégorie, puis choisissez Continue (Continuer). Pour les descriptions de paramètre, consultez la section the section called “Paramètres de catégorie de capacité” de cette page.

> **Sélection de l'architecture du processeur**
>
> Faites défiler la page jusqu'à Processeur pour sélectionner une architecture de processeur pour vos EC2 instances. La console répertorie les architectures de processeur prises en charge par la plateforme que vous avez choisie précédemment dans le panneau Créer un environnement.
> Si vous ne voyez pas l'architecture de processeur dont vous avez besoin, revenez à la liste des catégories de configuration pour sélectionner une plateforme qui la prend en charge. Dans le panneau Modify Capacity (Modifier la capacité), choisissez Annuler. Ensuite, choisissez Change platform version (Modifier la version de la plateforme) pour choisir de nouveaux paramètres de plateforme. Ensuite, dans la catégorie de configuration Capacity (Capacité), choisissez Edit(Modifier) pour ne plus voir les choix d'architecture du processeur.



7. Choisissez Enregistrer, puis effectuez toutes les autres modifications de configuration exigées par votre environnement.
8. Choisissez Create environment.

Pour configurer les EC2 instances Amazon d'un environnement d'exécution dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Instances, choisissez Edit (Modifier). Modifiez les paramètres de cette catégorie, puis choisissez Apply (Appliquer). Pour les descriptions de paramètre, consultez la section the section called “Paramètres de catégorie d'instances” de cette page.

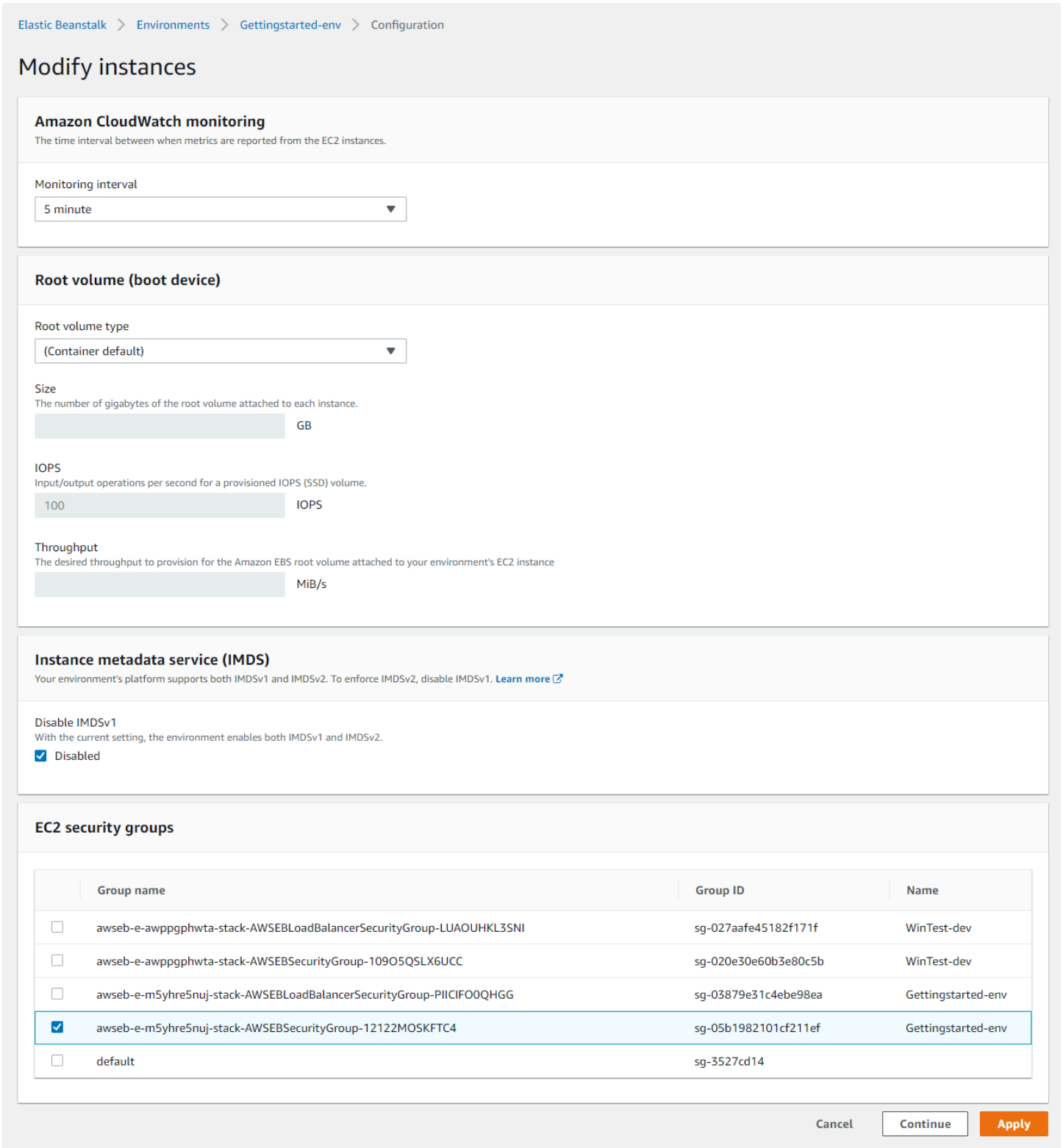
5. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier). Modifiez les paramètres de cette catégorie, puis choisissez Continue (Continuer). Pour les descriptions de paramètre, consultez la section the section called “Paramètres de catégorie de capacité” de cette page.

Paramètres de catégorie d'instances

Les paramètres suivants relatifs aux EC2 instances Amazon sont disponibles dans la catégorie Configuration des instances.

Options

- Intervalle de surveillance
- Volume racine (périphérique de démarrage)
- Service des métadonnées d'instance
- EC2 groupes de sécurité


Elastic Beanstalk > Environments > Gettingstarted-env > Configuration
Modify instances
Amazon CloudWatch monitoring
The time interval between when metrics are reported from the EC2 instances.
Monitoring interval
5 minute
Root volume (boot device)
Root volume type
(Container default)
Size
The number of gigabytes of the root volume attached to each instance.
GB
IOPS
Input/output operations per second for a provisioned IOPS (SSD) volume.
100
IOPS
Throughput
The desired throughput to provision for the Amazon EBS root volume attached to your environment's EC2 instance
MiB/s
Instance metadata service (IMDS)
Your environment's platform supports both IMDSv1 and IMDSv2. To enforce IMDSv2, disable IMDSv1. Learn more
Disable IMDSv1
With the current setting, the environment enables both IMDSv1 and IMDSv2.
Disabled
EC2 security groups
Group name
Group ID
Name
awseb-e-awppgphwta-stack-AWSEBLoadBalancerSecurityGroup-LUAOUHKL3SNI
sg-027aafe45182f171f
WinTest-dev
awseb-e-awppgphwta-stack-AWSEBSecurityGroup-109O5QSLX6UCC
sg-020e30e60b3e80c5b
WinTest-dev
awseb-e-m5yhre5nuj-stack-AWSEBLoadBalancerSecurityGroup-PIICIFO0QHGG
sg-03879e31c4ebe98ea
Gettingstarted-env
awseb-e-m5yhre5nuj-stack-AWSEBSecurityGroup-12122MOSKFTC4
sg-05b1982101cf211ef
Gettingstarted-env
default
sg-3527cd14
Cancel
Continue
Apply

Intervalle de surveillance

Par défaut, les instances de votre environnement publient des indicateurs de santé de base CloudWatch sur Amazon toutes les cinq minutes, sans frais supplémentaires.

Pour des rapports plus détaillés, vous pouvez définir l'intervalle de surveillance à 1 minute afin d'augmenter la fréquence à laquelle les ressources de votre environnement publient les indicateurs de santé de base CloudWatch à. CloudWatch des frais de service s'appliquent pour les mesures d'intervalle d'une minute. Pour plus d'informations, consultez Amazon CloudWatch.

Volume racine (périphérique de démarrage)

Chaque instance dans votre environnement est configurée avec un volume racine. Le volume racine est le périphérique de stockage en mode bloc Amazon EBS attaché à l'instance pour stocker le système d'exploitation, les bibliothèques, les scripts et le code source de votre application. Par défaut, toutes les plateformes utilisent des périphériques de stockage en mode bloc à usage général (SSD) pour le stockage.

Vous pouvez modifier Type de volume racine pour utiliser les types de volumes d'IOPS provisionnées (SSD) ou de stockage magnétique et, si nécessaire, accroître la taille du volume. Pour les volumes d'IOPS provisionnées, vous devez également sélectionner le nombre d'IOPS à approvisionner. Débit est uniquement applicable aux types de volumes SSD gp3. Vous pouvez saisir le débit souhaité à allouer. Il peut varier entre 125 et 1 000 megioctets par seconde (Mio/s). Sélectionnez le type de volume qui répond à vos exigences de prix et de performance.

> **Important**
>
> Le paramètre d'`RootVolumeType`option peut amener Elastic Beanstalk à migrer un environnement existant avec des configurations de lancement vers des modèles de lancement. Pour ce faire, vous devez disposer des autorisations nécessaires pour gérer les modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si vous utilisez des politiques personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre environnement. Pour plus d'informations et d'autres considérations, consultezMigration de votre environnement Elastic Beanstalk pour lancer des modèles.

Pour plus d'informations, consultez les types de volumes Amazon EBS dans le guide de l' EC2 utilisateur Amazon et les détails du produit Amazon EBS.

Service des métadonnées d'instance

Le service de métadonnées d'instance (IMDS) est un composant sur instance utilisé par le code sur l'instance pour accéder en toute sécurité aux métadonnées d'instance. Le code peut accéder aux métadonnées d'instance à partir d'une instance en cours d'exécution en utilisant l'une des deux méthodes suivantes : Il s'agit du service de métadonnées d'instance version 1 (IMDSv1) ou du service de métadonnées d'instance version 2 (IMDSv2). IMDSv2 est plus sûr. Désactivez IMDSv1 pour appliquer IMDSv2. Pour de plus amples informations, veuillez consulter the section called “IMDS”.

> **Note**
>
> La section IMDS de cette page de configuration apparaît uniquement pour les versions de plate-forme compatibles IMDSv2.

EC2 groupes de sécurité

Les groupes de sécurité attachés à vos instances déterminent le trafic autorisé à atteindre les instances et à en sortir.

Le groupe EC2 de sécurité par défaut créé par Elastic Beanstalk autorise tout le trafic entrant en provenance d'Internet ou des équilibreurs de charge sur les ports standard pour HTTP (80) et SSH (22). Vous pouvez également définir vos propres groupes de sécurité personnalisés afin de définir des règles de pare-feu pour les EC2 instances. Les groupes de sécurité peuvent autoriser le trafic sur d'autres ports ou en provenance d'autres sources. Par exemple, vous pouvez créer un groupe de sécurité pour l'accès SSH qui autorise le trafic entrant sur le port 22 à partir d'une plage d'adresses IP limitée. Pour plus de sécurité, vous pouvez également en créer un qui autorise le trafic provenant d'un hôte bastion auquel vous seul pouvez accéder.

Vous pouvez choisir de désactiver votre environnement du groupe de EC2 sécurité par défaut en définissant l'`DisableDefaultEC2SecurityGroup`option dans l'espace de aws:autoscaling:launchconfiguration noms sur`true`. Cette option n'est pas disponible dans la console, mais vous pouvez la définir avec le AWS CLI. Pour de plus amples informations, veuillez consulter Gestion des groupes de sécurité EC2.

Pour plus d'informations sur les groupes EC2 de sécurité Amazon, consultez Amazon EC2 Security Groups dans le guide de EC2 l'utilisateur Amazon.

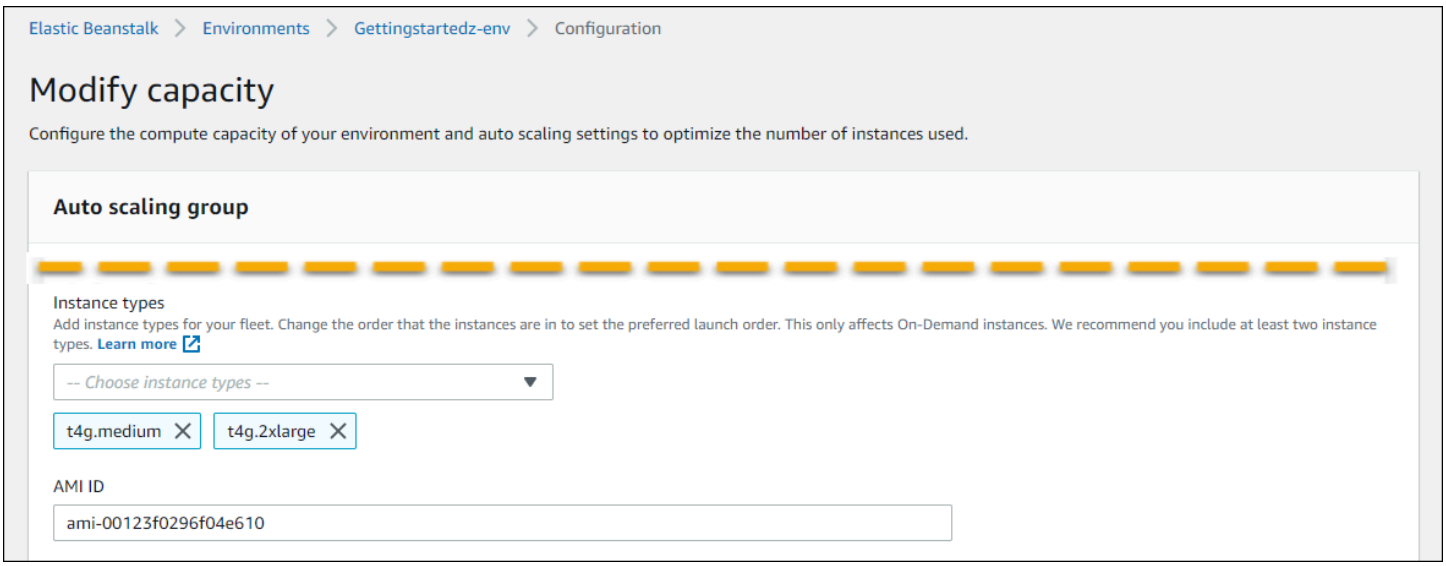
> **Note**
>
> Pour autoriser le trafic entre les instances de l'environnement A et celles de l'environnement B, vous pouvez ajouter une règle au groupe de sécurité qu'Elastic Beanstalk a attaché à l'environnement B. Puis, vous pouvez spécifier le groupe de sécurité qu'Elastic Beanstalk a attaché à l'environnement A. Ceci permet le trafic entrant dans/le trafic sortant depuis les instances de l'environnement A. Toutefois, cette opération crée une dépendance entre les deux groupes de sécurité. Si vous essayez de résilier ultérieurement l'environnement A, Elastic Beanstalk ne pourra pas supprimer le groupe de sécurité de l'environnement, car le groupe de sécurité de l'environnement B en est dépendant.
> Par conséquent, nous vous recommandons de créer un groupe de sécurité distinct. Ensuite, attachez-le à l'environnement A et spécifiez-le dans une règle du groupe de sécurité de l'environnement B.

Paramètres de catégorie de capacité

Les paramètres suivants relatifs aux EC2 instances Amazon sont disponibles dans la catégorie Configuration de la capacité.

Options

- Types d'instances
- ID d'AMI



Types d’instances

Le paramètre Types d'instances détermine le type d' EC2 instance Amazon lancée pour exécuter votre application. Cette page de configuration affiche une liste de types d'instance. Vous pouvez sélectionner un ou plusieurs types d'instance. La console Elastic Beanstalk affiche uniquement les types d'instance en fonction de l'architecture de processeur configurée pour votre environnement. Par conséquent, vous ne pouvez ajouter que des types d'instance de la même architecture de processeur.

> **Note**
>
> Bien que la console Elastic Beanstalk ne propose pas la possibilité de modifier l'architecture du processeur d'un environnement existant, vous pouvez le faire avec le. AWS CLI Pour des exemples de commandes, voirConfiguration des groupes de sécurité et des types d'instances Amazon EC2 à l'aide du AWS CLI.

Choisissez une instance qui soit suffisamment puissante pour exécuter votre application en charge, mais pas puissante au point d'être inactive la plupart du temps. A des fins de développement, la famille t2 d'instances fournit une quantité modérée de puissance avec la possibilité de rafales sur de courtes périodes. À grande échelle, les applications à haut niveau de disponibilité utilisent un groupe d'instances afin de garantir que cette capacité ne soit pas considérablement affectée si une seule instance s'arrête. Commencez par un type d'instance qui vous permet d'exécuter cinq instances sous des charges modérées pendant les heures normales. En cas de défaillance d'une instance, les instances restantes peuvent absorber le reste du trafic. Le tampon de capacité laisse également du temps pour que l'environnement s'adapte à mesure que le trafic commence à augmenter pendant les heures de pointe.

Pour plus d'informations sur les familles et les types d' EC2 instances Amazon, consultez la section Types d'instances dans le guide de EC2 l'utilisateur Amazon. Pour déterminer quels types d'instances répondent à vos besoins et quelles sont les régions prises en charge, consultez la section Types d'instances disponibles dans le guide de EC2 l'utilisateur Amazon.

ID d’AMI

L'Amazon Machine Image (AMI) est l'image de machine Amazon Linux ou Windows Server qu'Elastic Beanstalk utilise pour EC2 lancer des instances Amazon dans votre environnement. Elastic Beanstalk fournit des images de machine contenant les outils et les ressources requis pour exécuter votre application.

Elastic Beanstalk sélectionne une valeur AMI par défaut pour votre environnement en fonction de la Région, de la version de plateforme et de l'architecture de processeur que vous choisissez. Si vous avez créé une AMI personnalisée, remplacez l'ID AMI par défaut par votre ID personnalisé.

Gestion des groupes de sécurité EC2

Lorsqu'Elastic Beanstalk crée un environnement, il attribue un groupe de sécurité par défaut aux instances EC2 lancées avec celui-ci. Les groupes de sécurité attachés à vos instances déterminent le trafic autorisé à atteindre les instances et à en sortir.

Le groupe de sécurité EC2 par défaut créé par Elastic Beanstalk autorise tout le trafic entrant en provenance d'Internet ou des équilibreurs de charge sur les ports standard pour HTTP (80) et SSH (22). Vous pouvez également définir vos propres groupes de sécurité personnalisés afin de définir des règles de pare-feu pour les instances EC2. Les groupes de sécurité peuvent autoriser le trafic sur d'autres ports ou en provenance d'autres sources. Par exemple, vous pouvez créer un groupe de sécurité pour l'accès SSH qui autorise le trafic entrant sur le port 22 à partir d'une plage d'adresses IP limitée. Pour plus de sécurité, vous pouvez également en créer un qui autorise le trafic provenant d'un hôte bastion auquel vous seul pouvez accéder.

Vous pouvez choisir de désactiver votre environnement du groupe de sécurité EC2 par défaut en définissant l'`DisableDefaultEC2SecurityGroup`option dans l'espace de aws:autoscaling:launchconfiguration noms sur. `true` Utilisez les fichiers de configuration AWS CLIou pour appliquer cette option à votre environnement et pour associer des groupes de sécurité personnalisés aux instances EC2.

Gestion des groupes de sécurité EC2 dans des environnements multi-instances

Si vous créez un groupe de sécurité EC2 personnalisé dans un environnement multi-instances, vous devez également tenir compte de la manière dont les équilibreurs de charge et les règles de trafic entrant garantissent la sécurité et l'accessibilité de vos instances.

Le trafic entrant vers un environnement comportant plusieurs instances EC2 est géré par l'équilibreur de charge, qui dirige le trafic entrant entre toutes les instances EC2. Lorsqu'Elastic Beanstalk crée un groupe de sécurité EC2 par défaut, il définit également des règles entrantes qui autorisent le trafic entrant depuis l'équilibreur de charge. Sans cette règle entrante dans le groupe de sécurité, le trafic entrant ne sera pas autorisé à entrer dans les instances. Cette condition empêcherait essentiellement les instances de recevoir des demandes externes.

Si vous désactivez le groupe de sécurité EC2 par défaut pour un environnement d'équilibrage de charge, Elastic Beanstalk valide certaines règles de configuration. Si la configuration ne répond

pas aux contrôles de validation, il émet des messages vous demandant de fournir la configuration requise. Les contrôles de validation sont les suivants :

- Au moins un groupe de sécurité doit être attribué à l'équilibreur de charge à l'aide de l'`SecurityGroups`option aws:elbv2:loadbalancer ouaws:elb:loadbalancer, selon qu'il s'agit d'un équilibreur de charge d'application ou d'un équilibreur de charge classique, respectivement. Pour des AWS CLI exemples, voirConfiguration à l'aide du AWS CLI.
- Des règles de trafic entrant doivent exister pour permettre à vos instances EC2 de recevoir du trafic en provenance de l'équilibreur de charge. Vos groupes de sécurité EC2 et vos groupes de sécurité d'équilibreur de charge doivent faire référence à ces règles entrantes. Pour plus d'informations, consultez Règles relatives au trafic entrantla section suivante.

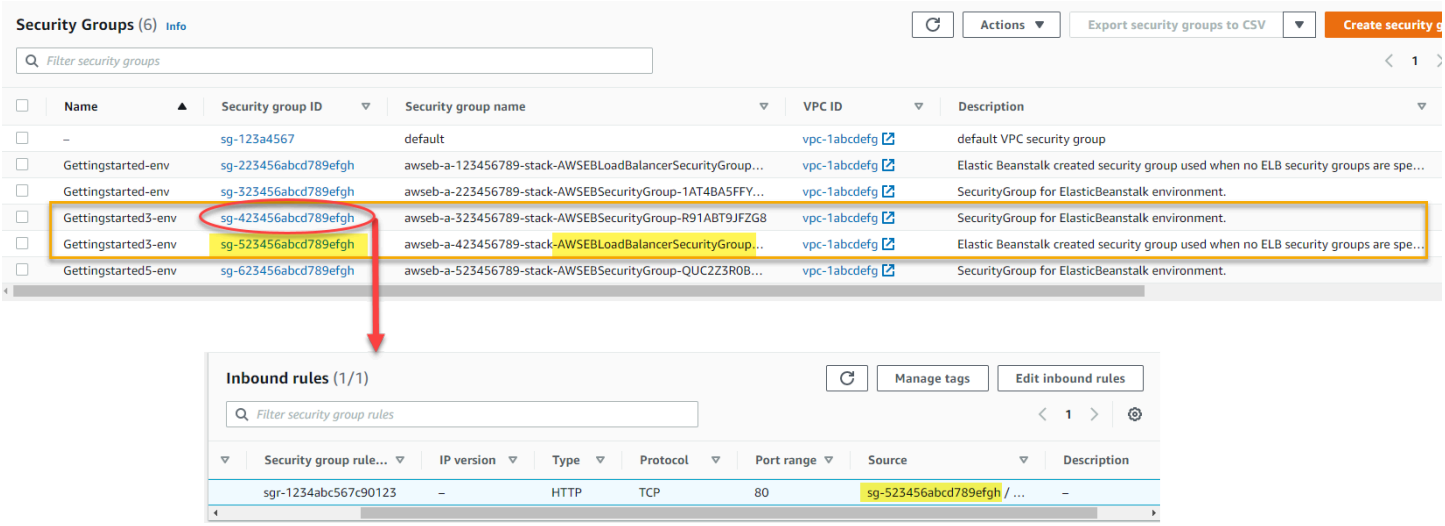
Règles relatives au trafic entrant

Le ou les groupes de sécurité EC2 pour un environnement multi-instances doivent inclure une règle entrante qui fait référence au groupe de sécurité de l'équilibreur de charge. Cela s'applique aux environnements dotés de tout type d'équilibreur de charge, dédié ou partagé, et dotés de groupes de sécurité d'équilibreurs de charge personnalisés ou par défaut.

Vous pouvez consulter tous les groupes de sécurité attachés aux composants de votre environnement dans la console EC2. L'image suivante montre la liste des groupes de sécurité créés par défaut par Elastic Beanstalk lors de l'opération de création d'un environnement sur la console EC2.

L'écran Groupes de sécurité affiche les environnements et les groupes de sécurité associés. GettingStarted-env et GettingStarted3-env sont des environnements multi-instances dotés d'équilibreurs de charge dédiés. Chacun de ces environnements possède deux groupes de sécurité répertoriés, l'un pour les instances EC2 et l'autre pour l'équilibreur de charge. Elastic Beanstalk crée ces groupes de sécurité lorsqu'il crée les environnements. GettingStarted5-env ne possède pas de groupe de sécurité d'équilibreur de charge, car il ne possède qu'une seule instance EC2, et donc aucun équilibreur de charge.

L'écran des règles entrantes explore le groupe de sécurité EC2 pour les instances de 3-env. GettingStarted Cet exemple définit les règles entrantes pour le groupe de sécurité EC2. Notez que la colonne Source des règles entrantes répertorie l'identifiant du groupe de sécurité du groupe de sécurité de l'équilibreur de charge répertorié dans l'image précédente. Cette règle permet aux instances EC2 de GettingStarted3-env de recevoir du trafic entrant depuis cet équilibreur de charge spécifique sur le port 80.



Pour plus d'informations, consultez Modifier les groupes de sécurité pour votre instance et les règles d'Elastic Load Balancing dans le guide de l'utilisateur Amazon EC2.

Configuration des groupes de sécurité et des types d'instances Amazon EC2 à l'aide du AWS CLI

Vous pouvez utiliser l'interface de ligne de AWS commande (AWS CLI) pour configurer les instances Amazon EC2 dans vos environnements Elastic Beanstalk.

Configuration des groupes de sécurité EC2 à l'aide du AWS CLI

Cette rubrique fournit des exemples de différentes configurations de groupes de sécurité EC2 pour les environnements à instance unique et à charge équilibrée (multi-instances). Pour plus d'informations sur les options de ces exemples, consultez aws:autoscaling:launchconfiguration.

> **Remarques**
>
> L'opération de création d'environnement fournit un groupe de sécurité EC2 par défaut. Il crée également un environnement avec un équilibreur de charge d'application par défaut. L'opération de mise à jour de l'environnement peut être utilisée pour désactiver ou activer le groupe de sécurité EC2 par défaut pour votre environnement à l'aide de l'option booléenne. `DisableDefaultEC2SecurityGroup` L'exemple 5 montre comment rétablir la configuration de sécurité par défaut de votre environnement si vous l'avez déjà modifiée.

Les exemples suivants montrent une commande create-environment désactivant le groupe de sécurité EC2 par défaut et fournissant des groupes de sécurité personnalisés à la place. L'`DisableDefaultEC2SecurityGroup`option étant définie sur`true`, le groupe de sécurité EC2 par défaut qu'Elastic Beanstalk associe normalement aux instances EC2 n'est pas créé. Par conséquent, vous devez fournir `SecurityGroups` cette option à d'autres groupes de sécurité.

Notez que l'aws:elasticbeanstalk:environment`EnvironmentType`option est définie sur`SingleInstance`. Pour créer un environnement à instance unique, vous devez spécifier cette option, car il s'`LoadBalancer`agit de la valeur par défaut`EnvironmentType`. Étant donné que cet environnement n'inclut pas d'équilibreur de charge, il n'est pas nécessaire de spécifier un groupe de sécurité d'équilibreur de charge.

Example 1 — Nouvel environnement à instance unique avec groupes de sécurité EC2 personnalisés (options d'espace de noms intégrées)

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v6.5.0 applrunning Node.js 22" \
--option-settings \
Namespace=aws:elasticbeanstalk:environment,OptionName=EnvironmentType,Value=SingleInstance
 \
Namespace=aws:autoscaling:launchconfiguration,OptionName=IamInstanceProfile,Value=aws-
elasticbeanstalk-ec2-role \
Namespace=aws:autoscaling:launchconfiguration,OptionName=DisableDefaultEC2SecurityGroup,Value=t
 \
Namespace=aws:autoscaling:launchconfiguration,OptionName=SecurityGroups,Value=sg-
abcdef01, sg-abcdef02 \
Namespace=aws:autoscaling:launchconfiguration,OptionName=EC2KeyName,Value=my-keypair
```

À titre de solution de rechange, utilisez un fichier `options.json` pour spécifier les options de l'espace de noms au lieu de les inclure en ligne.

Example 2 — Nouvel environnement à instance unique avec groupes de sécurité EC2 personnalisés (options d'espace de noms dans le fichier) **`options.json`**

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
```

```
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v6.5.0 running Node.js 22" \
--option-settings file://options.json
```

Example

```
### example options.json ###
[
  { "Namespace" : "aws:elasticbeanstalk:environment",
    "OptionName" : "EnvironmentType",
    "Value" : "SingleInstance"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "IamInstanceProfile",
    "Value": "aws-elasticbeanstalk-ec2-role"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "DisableDefaultEC2SecurityGroup",
    "Value": "true"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "SecurityGroups",
    "Value": "sg-abcdef01, sg-abcdef02"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "EC2KeyName",
    "Value": "my-keypair"
  }
]
```

L'exemple suivant crée un environnement d'équilibrage de charge. Elle spécifie l'option d'aws:elasticbeanstalk:environmentespace de noms `LoadBalancerType` définie sur. `application` Comme nous désactivons le groupe de sécurité EC2 par défaut avec `DisableDefaultEC2SecurityGroup` cette option, nous devons à nouveau fournir nos propres groupes de sécurité personnalisés pour les instances EC2, avec aws:autoscaling:launchconfiguration `SecurityGroups` cette option, comme dans l'exemple précédent. Étant donné que cet environnement dispose d'un équilibreur de charge pour acheminer le trafic, nous devons également fournir des groupes de sécurité pour l'équilibreur de charge.

Pour créer un environnement avec un équilibreur de charge classique, mais avec la même configuration, mettez à jour la configuration de l'option `LoadBalancerType` d'espace de aws:elasticbeanstalk:environment noms sur. `classic`

Les différents types d'équilibreurs de charge ont des espaces de noms différents qui contiennent les options permettant de spécifier les groupes de sécurité :

- équilibreur de charge d'application — option aws:elbv2:loadbalancer `SecurityGroups`
- équilibreur de charge classique — option aws:elb:loadbalancer `SecurityGroups`
- équilibreur de charge réseau : les équilibreurs de charge réseau n'ayant pas de groupes de sécurité, configurez les groupes de sécurité EC2 avec des identifiants VPC. Pour plus d'informations, voir Mettre à jour les groupes de sécurité pour votre Network Load Balancer dans le Guide de l'utilisateur pour les Network Load Balancers.

Example 3 — Nouvel environnement multi-instance avec groupes de sécurité EC2 personnalisés (options d'espace de noms dans le fichier) **`options.json`**

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v6.5.0 running Node.js 22" \
--option-settings file://options.json
```

Example

```
### example options.json ###
[
  {
    "Namespace" : "aws:elasticbeanstalk:environment",
    "OptionName" : "EnvironmentType",
    "Value" : "LoadBalanced"
  },
  {
  "Namespace" : "aws:elasticbeanstalk:environment",
    "OptionName" : "LoadBalancerType",
    "Value" : "application"
  },
  {
```

```
    "Namespace" : "aws:elbv2:loadbalancer",
    "OptionName" : "SecurityGroups",
    "Value" : "sg-abcdefghikl012345"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "IamInstanceProfile",
    "Value": "aws-elasticbeanstalk-ec2-role"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "DisableDefaultEC2SecurityGroup",
    "Value": "true"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "SecurityGroups",
    "Value": "sg-abcdef01, sg-abcdef02"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "EC2KeyName",
    "Value": "my-keypair"
  }
]
```

Vous pouvez désactiver le groupe de sécurité EC2 par défaut pour un environnement existant à l'aide de la commande update-environment. L'exemple de commande suivant désactive le groupe de sécurité EC2 par défaut et attribue des groupes de sécurité EC2 personnalisés aux instances EC2 de l'environnement.

Utilisez les `options.jason` fichiers d'exemple des exemples 4 (a), 4 (b) ou 4 (c), selon que l'environnement est équilibré en charge et selon le type d'équilibreur de charge. Le fichier de configuration 4 (a) spécifie les groupes de sécurité pour un environnement à instance unique. Comme il ne nécessite pas d'équilibreur de charge, nous fournissons uniquement le groupe de sécurité pour les instances EC2. Les fichiers de configuration 4 (b) et 4 (c) spécifient les groupes de sécurité pour un équilibreur de charge d'application et un équilibreur de charge classique. Dans ces cas, nous devons également spécifier des groupes de sécurité pour l'équilibreur de charge.

Example 4 — Mettre à jour un environnement existant pour désactiver le groupe de sécurité EC2 par défaut (options d'espace de noms dans le **`options.json`** fichier)

```
aws elasticbeanstalk update-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v6.5.0 running Node.js 22" \
--option-settings file://options.json
```

Example 4 (a) — Fichier de configuration pour un environnement à instance unique (sans équilibreur de charge)

```
### example options.json ###
[
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "DisableDefaultEC2SecurityGroup",
    "Value": "true"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "SecurityGroups",
    "Value": "sg-abcdef01, sg-abcdef02"
  }
]
```

Pour mettre à jour un environnement qui utilise un équilibreur de charge d'application, utilisez l'espace de `aws:elbv2:loadbalancer` noms pour spécifier les groupes de sécurité pour l'équilibreur de charge.

Example 4 (b) — Fichier de configuration pour un environnement doté d'un équilibreur de charge d'application

```
### example options.json ###
[
  {
    "Namespace" : "aws:elbv2:loadbalancer",
    "OptionName" : "SecurityGroups",
    "Value" : "sg-abcdefghikl012345"
  },
  {
```

```
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "DisableDefaultEC2SecurityGroup",
    "Value": "true"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "SecurityGroups",
    "Value": "sg-abcdef01, sg-abcdef02"
  }
]
```

Pour mettre à jour un environnement qui utilise un équilibreur de charge classique, utilisez l'espace de `aws:elb:loadbalancer` noms pour spécifier les groupes de sécurité pour l'équilibreur de charge.

Example 4 (c) — Fichier de configuration pour un environnement avec un équilibreur de charge classique

```
### example options.json ###
[
  {
    "Namespace" : "aws:elb:loadbalancer",
    "OptionName" : "SecurityGroups",
    "Value" : "sg-abcdefghikl012345"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "DisableDefaultEC2SecurityGroup",
    "Value": "true"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",n
    "OptionName": "SecurityGroups",
    "Value": "sg-abcdef01, sg-abcdef02"
  }
]
```

Pour rétablir le comportement et la configuration par défaut de votre environnement avec le groupe de sécurité par défaut attribué par Elastic Beanstalk, utilisez la commande update-environment pour définir la valeur sur. `DisableDefaultEC2SecurityGroup` `false` Pour un environnement multi-instances, Elastic Beanstalk gère également les groupes de sécurité et les règles de trafic réseau pour l'équilibreur de charge de votre environnement.

L'exemple suivant s'applique à un environnement à instance unique ou à instances multiples (équilibrage de charge) :

Example 5 — Mettre à jour un environnement pour qu'il utilise à nouveau le groupe de sécurité par défaut (options d'espace de noms dans **`options.json`** le fichier)

```
aws elasticbeanstalk update-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v6.5.0 running Node.js 22" \
--option-settings file://options.json
```

Example

```
### example options.json ###
[
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "DisableDefaultEC2SecurityGroup",
    "Value": "false"
  }
]
```

Configuration d'EC2 avec des types d'instances à l'aide du AWS CLI

Cette rubrique fournit des exemples de configuration des types d'instances EC2 dans votre environnement.

Les deux premiers exemples créent un environnement. La commande indique un type d'instance Amazon EC2, t4g.small, basé sur l'architecture du processeur arm64. Elastic Beanstalk définit par défaut l'ID d'image (AMI) des instances EC2 en fonction de la région, de la version de la plateforme et du type d'instance. Le type d'instance correspond à une architecture de processeur. Le paramètre `solution-stack-name` s'applique à la version de la plateforme.

Example 1 : créer un environnement basé sur arm64 (options d'espace de noms en ligne)

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
```

```
--solution-stack-name "64bit Amazon Linux 2 v3.4.7 running Docker" \
--option-settings \
Namespace=aws:autoscaling:launchconfiguration,OptionName=IamInstanceProfile,Value=aws-elasticbeanstalk-ec2-role \
Namespace=aws:ec2:instances,OptionName=InstanceTypes,Value=t4g.small
```

À titre de solution de rechange, utilisez un fichier `options.json` pour spécifier les options de l'espace de noms au lieu de les inclure en ligne.

Example 2 : créer un environnement basé sur arm64 (options d'espace de noms en ligne dans le fichier **`options.json`**)

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2 v3.4.7 running Docker" \
--option-settings file://options.json
```

Example

```
### example options.json ###
[
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "IamInstanceProfile",
    "Value": "aws-elasticbeanstalk-ec2-role"
  },
  {
    "Namespace": "aws:ec2:instances",
    "OptionName": "InstanceTypes",
    "Value": "t4g.small"
  }
]
```

Les deux exemples suivants mettent à jour la configuration d'un environnement existant avec la commande update-environment. Dans cet exemple, nous ajoutons un autre type d'instance basé également sur l'architecture du processeur arm64. Pour les environnements existants, tous les types d'instance ajoutés doivent avoir la même architecture de processeur. Si vous souhaitez remplacer

les types d'instance existants par ceux issus d'une architecture différente, vous pouvez le faire. Mais assurez-vous que tous les types d'instances de la commande ont le même type d'architecture.

Example 3 : mettre à jour un environnement basé sur arm64 existant (options d'espace de noms en ligne)

```
aws elasticbeanstalk update-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2 v3.4.7 running Docker" \
--option-settings \
Namespace=aws:autoscaling:launchconfiguration,OptionName=IamInstanceProfile,Value=aws-elasticbeanstalk-ec2-role \
Namespace=aws:ec2:instances,OptionName=InstanceTypes,Value=t4g.small,t4g.micro
```

À titre de solution de rechange, utilisez un fichier `options.json` pour spécifier les options de l'espace de noms au lieu de les inclure en ligne.

Example 4 : mettre à jour un environnement basé sur arm64 existant (options d'espace de noms dans le fichier **`options.json`**)

```
aws elasticbeanstalk update-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2 v3.4.7 running Docker" \
--option-settings file://options.json
```

Example

```
### example options.json ###
[
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "IamInstanceProfile",
    "Value": "aws-elasticbeanstalk-ec2-role"
  },
  {
    "Namespace": "aws:ec2:instances",
    "OptionName": "InstanceTypes",
```

```
    "Value": "t4g.small, t4g.micro"
  }
]
```

Les deux exemples suivants montrent des commandes create-environment supplémentaires. Ces exemples ne fournissent pas de valeurs pour `InstanceTypes`. Quand les valeurs `InstanceTypes` ne sont pas spécifiées, Elastic Beanstalk utilise par défaut une architecture de processeur basée sur x86. L'ID d'image (AMI) des instances EC2 de l'environnement sera défini par défaut en fonction de la région, de la version de la plateforme et du type d'instance par défaut. Le type d'instance correspond à une architecture de processeur.

Example 5 : créer un environnement basé sur x86 (options d'espace de noms en ligne)

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2 v3.4.7 running Docker" \
--option-settings \
Namespace=aws:autoscaling:launchconfiguration,OptionName=IamInstanceProfile,Value=aws-elasticbeanstalk-ec2-role
```

À titre de solution de rechange, utilisez un fichier `options.json` pour spécifier les options de l'espace de noms au lieu de les inclure en ligne.

Example 6 : créer un environnement basé sur x86 (options d'espace de noms en ligne dans le fichier **`options.json`**)

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2 v3.4.7 running Docker" \
--option-settings file://options.json
```

Example

```
### example options.json ###
```

```
[
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "IamInstanceProfile",
    "Value": "aws-elasticbeanstalk-ec2-role"
  }
]
```

Configuration des EC2 instances Amazon avec des options d'espace de noms

Vous pouvez utiliser les options de configuration dans l'espace de noms `aws:autoscaling:launchconfiguration` afin de configurer les instances de votre environnement, y compris les options supplémentaires qui ne sont pas disponibles dans la console.

> **Important**
>
> Le paramètre d'`BlockDeviceMappings`option `DisableIMDSv1RootVolumeType`, ou peut amener Elastic Beanstalk à migrer un environnement existant avec des configurations de lancement vers des modèles de lancement. Pour ce faire, vous devez disposer des autorisations nécessaires pour gérer les modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si vous utilisez des politiques personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre environnement. Pour plus d'informations et d'autres considérations, consultezMigration de votre environnement Elastic Beanstalk pour lancer des modèles.

L'exemple de fichier de configuration suivant utilise les options de configuration de base expliquées dans cette rubrique. Pour voir des exemples d'options de configuration supplémentaires lorsque vous devez spécifier des groupes de sécurité pour les équilibreurs de charge, consultezConfiguration à l'aide du AWS CLI.

```
option_settings:
  aws:autoscaling:launchconfiguration:
    SecurityGroups: my-securitygroup
    MonitoringInterval: "1 minute"
    DisableIMDSv1: false
    DisableDefaultEC2SecurityGroup: true
```

```
    SecurityGroups: "sg-abcdef01, sg-abcdef02"
    EC2KeyName: my-keypair
    IamInstanceProfile: "aws-elasticbeanstalk-ec2-role"
    BlockDeviceMappings: "/dev/sdj=:100,/dev/sdh=snap-51eef269,/dev/sdb=ephemeral0"
  aws:elasticbeanstalk:environment:
    EnvironmentType: SingleInstance
```

Les `DisableDefaultEC2SecurityGroup` et ne `BlockDeviceMappings` sont pas disponibles dans la console.

Vous pouvez utiliser `BlockDeviceMappings` pour configurer des périphériques de stockage en mode bloc supplémentaires pour vos instances. Pour plus d'informations, consultez la section Block Device Mapping dans le guide de EC2 l'utilisateur Amazon.

L'interface de ligne de commande (CLI) EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Consultez Valeurs recommandées pour plus de détails.

Configuration de l'IMDS sur les instances de votre environnement Elastic Beanstalk

Cette rubrique décrit le service de métadonnées d'instance (IMDS).

Les métadonnées d'instance sont des données liées à une instance Amazon Elastic Compute Cloud (Amazon EC2) que les applications peuvent utiliser pour configurer ou gérer l'instance en cours d'exécution. Le service de métadonnées d'instance (IMDS) est un composant sur instance utilisé par le code sur l'instance pour accéder en toute sécurité aux métadonnées d'instance. Ce code peut être le code de la plateforme Elastic Beanstalk sur les instances de votre environnement AWS , le SDK que votre application est susceptible d'utiliser, ou même le propre code de votre application. Pour plus d'informations, consultez la section Métadonnées de l'instance et données utilisateur dans le guide de EC2 l'utilisateur Amazon.

Le code peut accéder aux métadonnées d'une instance en cours d'exécution à l'aide de l'une des deux méthodes suivantes : service de métadonnées d'instance version 1 (IMDSv1) ou service de métadonnées d'instance version 2 (IMDSv2). IMDSv2 utilise des requêtes orientées session et atténue plusieurs types de vulnérabilités susceptibles d'être utilisées pour tenter d'accéder à l'IMDS. Pour plus d'informations sur ces deux méthodes, consultez la section Configuration du service de métadonnées d'instance dans le guide de EC2 l'utilisateur Amazon.

Rubriques

- Prise en charge de la plateforme pour IMDS
- Choisir des méthodes IMDS
- Configuration d'IMDS à l'aide de la console Elastic Beanstalk
- Espace de noms aws:autoscaling:launchconfiguration

Prise en charge de la plateforme pour IMDS

Les plateformes Elastic Beanstalk exécutées sur Amazon Linux 2 et Amazon Linux 2023, ainsi que sur un serveur Windows, prennent toutes en charge les deux et. IMDSv1 IMDSv2 Pour de plus amples informations, consultez Configuration d'IMDS à l'aide de la console Elastic Beanstalk.

Choisir des méthodes IMDS

Lorsque vous prenez une décision concernant les méthodes IMDS que votre environnement doit prendre en charge, tenez compte des cas d'utilisation suivants :

- AWS SDK — Si votre application utilise un AWS SDK, assurez-vous d'utiliser la dernière version du SDK. Ils AWS SDKs font des appels IMDS et les nouvelles versions du SDK les utilisent dans la IMDSv2 mesure du possible. Si vous le désactivez IMDSv1 ou si votre application utilise une ancienne version du SDK, les appels IMDS risquent d'échouer.
- Le code de votre application : si votre application effectue des appels IMDS, pensez à utiliser le AWS SDK afin de pouvoir effectuer les appels au lieu de faire des requêtes HTTP directes. De cette façon, vous n'avez pas besoin de modifier le code pour basculer entre les méthodes IMDS. Le AWS SDK est utilisé dans la mesure du IMDSv2 possible.
- Code de la plateforme Elastic Beanstalk : notre code effectue des appels IMDS AWS via le SDK et IMDSv2 est donc utilisé sur toutes les versions de plateforme compatibles. Si votre code utilise un up-to-date AWS SDK et effectue tous les appels IMDS via le SDK, vous pouvez le désactiver en toute sécurité. IMDSv1

Configuration d'IMDS à l'aide de la console Elastic Beanstalk

Vous pouvez modifier la configuration de l'instance Amazon de votre environnement Elastic Beanstalk dans EC2 la console Elastic Beanstalk.

> **Important**
>
> Le paramètre d'`DisableIMDSv1`option peut amener Elastic Beanstalk à migrer un environnement existant avec des configurations de lancement vers des modèles de lancement. Pour ce faire, vous devez disposer des autorisations nécessaires pour gérer les modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si vous utilisez des politiques personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre environnement. Pour plus d'informations et d'autres considérations, consultezMigration de votre environnement Elastic Beanstalk pour lancer des modèles.

Pour configurer l'IMDS sur vos EC2 instances Amazon dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Instances (Instances), choisissez Edit (Modifier).
5. Définissez Disable IMDSv1 pour appliquer IMDSv2. Désélectionnez Désactiver IMDSv1 pour activer à la fois IMDSv1 et IMDSv2.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Espace de noms aws:autoscaling:launchconfiguration

Vous pouvez utiliser une option de configuration dans l'espace de noms `aws:autoscaling:launchconfiguration` pour configurer IMDS sur les instances de votre environnement.

> **Important**
>
> Le paramètre d'`DisableIMDSv1`option peut amener Elastic Beanstalk à migrer un environnement existant avec des configurations de lancement vers des modèles de lancement. Pour ce faire, vous devez disposer des autorisations nécessaires pour gérer les modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si

vous utilisez des politiques personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre environnement. Pour plus d'informations et d'autres considérations, consultezMigration de votre environnement Elastic Beanstalk pour lancer des modèles.

L'exemple de fichier de configuration suivant désactive IMDSv1 l'utilisation de l'`DisableIMDSv1`option.

```
option_settings:
  aws:autoscaling:launchconfiguration:
    DisableIMDSv1: true
```

Définissez Disable `true` sur IMDSv1 pour désactiver IMDSv1 et appliquer IMDSv2.

Définissez Disable IMDSv1 sur `false` pour activer à la fois IMDSv1 et IMDSv2.

Auto Scaling de vos instances d'environnement Elastic Beanstalk

Cette rubrique décrit comment personnaliser les fonctionnalités d'Auto Scaling pour gérer la charge de travail de votre environnement Elastic Beanstalk. Vous pouvez configurer Auto Scaling pour votre environnement à l'aide de la console Elastic Beanstalk, des options de configuration de l'espace de noms ou de AWS CLIla CLI EB.

Environnements à charge équilibrée ou à instance unique

Votre AWS Elastic Beanstalk environnement inclut un groupe Auto Scaling qui gère les instances Amazon EC2 de votre environnement. Dans un environnement à instance unique, le groupe Auto Scaling s'assure qu'il y a toujours une instance en cours d'exécution. Dans un environnement à charge équilibrée, vous configurez le groupe avec une gamme d'instances à exécuter et l'Auto Scaling ajoute ou supprime les instances en fonction de la charge requise par vos besoins.

Configuration de l'instance EC2

Le groupe Auto Scaling applique également vos choix de configuration pour provisionner et gérer les instances EC2 de votre environnement. Vous pouvez modifier la configuration EC2 pour changer le type d'instance, la paire de clés, le stockage Amazon Elastic Block Store (Amazon EBS) et d'autres paramètres qui ne peuvent être configurés que lorsque vous lancez une instance.

Instances ponctuelles et à la demande

En option, Elastic Beanstalk peut inclure des instances ponctuelles dans votre environnement et les gérer en combinaison avec des instances à la demande. Vous pouvez configurer Amazon EC2 Auto Scaling pour surveiller et réagir automatiquement aux modifications qui affectent la disponibilité de vos instances Spot en activant Rééquilibrage de capacité. Vous pouvez également configurer les instances Stratégie d'allocation au comptant utilisées par le service Auto Scaling pour fournir des instances Spot à votre environnement.

Autorisations requises lors de l'activation des instances Spot

L'activation des requêtes d'instance Spot nécessite l'utilisation de modèles de lancement Amazon EC2. Lorsque vous configurez cette fonction au cours de la création ou des mises à jour de l'environnement, Elastic Beanstalk tente de configurer votre environnement de sorte qu'il utilise des modèles de lancement Amazon EC2 (si l'environnement ne les utilise pas déjà). Dans ce cas, si votre politique d'utilisateur ne dispose pas des autorisations nécessaires, la création ou les mises à jour de l'environnement peuvent échouer. Par conséquent, nous vous recommandons d'utiliser notre politique d'utilisateur gérée ou d'ajouter les autorisations requises à vos politiques personnalisées. Pour plus d'informations sur les autorisations requises, consultez Autorisations requises pour les modèles de lancement.

Déclencheurs de mise à l'échelle automatique

Le groupe Auto Scaling utilise deux CloudWatch alarmes Amazon pour déclencher des opérations de dimensionnement. Les déclencheurs par défaut évoluent quand le trafic réseau sortant moyen de chaque instance est supérieur à 6 Mio ou inférieur à 2 Mio sur une période de cinq minutes. Pour utiliser Auto Scaling de façon efficace, configurez des déclencheurs adaptés à votre application, au type d'instance et aux exigences du service. Vous pouvez mettre à l'échelle en fonction de plusieurs statistiques, y compris la latence, les I/O disque, l'utilisation de l'UC et le nombre de demandes.

Planifier des actions Auto Scaling

Pour optimiser l'utilisation des instances Amazon EC2 de votre environnement à travers les périodes prévisibles de trafic de pointe, configurez votre groupe Auto Scaling pour modifier son nombre d'instances sur un calendrier. Vous pouvez planifier les modifications apportées à la configuration de votre groupe qui se produisent de manière quotidienne ou hebdomadaire, ou planifier des modifications exceptionnelles pour préparer les événements marketing qui génèreront un trafic important sur votre site.

Bilan de santé d'Auto Scaling

La scalabilité automatique surveille l'état d'intégrité de chaque instance Amazon EC2 qu'il lance. Si une instance est résiliée de façon inattendue, Auto Scaling détecte cette résiliation et lance une instance de remplacement. Pour configurer le groupe afin d'utiliser le mécanisme de vérification de l'état d'intégrité de l'équilibreur de charge, consultez Paramètre de vérification de l'état Auto Scaling pour votre environnement Elastic Beanstalk.

Rubriques

- Migration de votre environnement Elastic Beanstalk pour lancer des modèles
- Support des instances Spot pour votre environnement Elastic Beanstalk
- Déclencheurs Auto Scaling pour votre environnement Elastic Beanstalk
- Actions Auto Scaling planifiées pour vos environnements Elastic Beanstalk
- Paramètre de vérification de l'état Auto Scaling pour votre environnement Elastic Beanstalk

Migration de votre environnement Elastic Beanstalk pour lancer des modèles

Depuis le 1er octobre 2024, Amazon EC2 Auto Scaling ne prend plus en charge les configurations de lancement pour les nouveaux comptes. Les comptes créés avant cette date peuvent avoir des configurations de lancement.

Nous vous recommandons de migrer vers des modèles de lancement pour bénéficier des avantages suivants :

- Disponibilité améliorée pour vos applications
- Meilleure optimisation des charges de travail dans vos groupes Auto Scaling
- Accès aux dernières fonctionnalités d'EC2 et d'Auto Scaling

Pour plus d'informations, consultez la section Configurations de lancement d'Auto Scaling dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling.

Paramètres des options pour les modèles de lancement

Pour migrer votre environnement des configurations de lancement vers les modèles de lancement, définissez l'une des options de configuration suivantes :

- `RootVolumeType`option définie sur gp3. Vous pouvez définir cette option à l'aide de la console ou de l'espace de noms.
- `BlockDeviceMappings`l'option contient gp3. Vous pouvez définir cette option à l'aide de la console ou de l'espace de noms.
- `DisableIMDSv1`option définie sur true. Nous vous recommandons de définir cette option à l'aide de l'espace de noms.
- `EnableSpot`option définie sur true. Pour de plus amples informations, veuillez consulter Activation des instances Spot.

> **Important**
>
> Une fois qu'un environnement commence à utiliser des modèles de lancement, Elastic Beanstalk ne revient pas aux configurations de lancement, même si vous supprimez les options de configuration qui ont initialement déclenché l'utilisation des modèles de lancement.

Vérifiez si votre environnement possède des configurations de lancement ou des modèles de lancement

Vous pouvez vérifier si votre environnement utilise déjà des modèles de lancement, ou s'il utilise des configurations de lancement, en inspectant le modèle de CloudFormation pile.

Pour inspecter le modèle de CloudFormation stack de votre environnement

1. Ouvrez la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.
2. Dans la barre de navigation en haut de l'écran, choisissez la AWS région dans laquelle vous avez créé l'environnement.
3. Sur la page Stacks de la CloudFormation console, inspectez la colonne Description.

   Localisez et sélectionnez la pile pour l'environnement Elastic Beanstalk. CloudFormation affiche les détails de la pile pour l'environnement.
4. Dans Détails de la pile, sélectionnez l'onglet Modèle.

   À l'aide de la recherche de page de votre navigateur, vous pouvez rechercher dans le texte du modèle launchtemplate ou launchconfiguration.

Pour plus d'informations, consultez la section Afficher les informations relatives à la pile dans le guide de AWS CloudFormation l'utilisateur.

Autorisations requises pour les modèles de lancement

La politique de AWSElasticBeanstalkManagedUpdatesCustomerRolePolicyrôle de service géré par défaut d'Elastic Beanstalk fournit les autorisations requises pour créer et gérer des modèles de lancement. Elastic Beanstalk doit gérer des modèles de lancement pour effectuer de nombreuses opérations liées à l'environnement, notamment la création d'environnements.

Si vous associez des politiques personnalisées à un rôle de service Elastic Beanstalk, vérifiez que le rôle de service inclut les autorisations suivantes pour créer des modèles de lancement. Ces autorisations permettent à Elastic Beanstalk de créer et de mettre à jour avec succès des environnements dans votre compte :

Autorisations requises pour les modèles de lancement Amazon EC2

- `ec2:RunInstances`
- `ec2:CreateLaunchTemplate`
- `ec2:CreateLaunchTemplateVersions`
- `ec2:DeleteLaunchTemplate`
- `ec2:DeleteLaunchTemplateVersions`
- `ec2:DescribeLaunchTemplate`
- `ec2:DescribeLaunchTemplateVersions`

L'exemple de politique IAM suivant inclut ces autorisations.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:RunInstances",
        "ec2:CreateLaunchTemplate",
        "ec2:CreateLaunchTemplateVersions",
        "ec2:DeleteLaunchTemplate",
        "ec2:DeleteLaunchTemplateVersions",
        "ec2:DescribeLaunchTemplate",
        "ec2:DescribeLaunchTemplateVersions"
```

```
      ],
      "Resource": [
                "*"
            ]
    }
  ]
}
```

Pour plus d'informations, consultez Gestion des rôles de service Elastic Beanstalk et Gestion des stratégies utilisateur Elastic Beanstalk.

En savoir plus sur les modèles de lancement

Pour en savoir plus sur les modèles de lancement, consultez les modèles de lancement Auto Scaling dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling.

Pour en savoir plus sur la AWS transition vers les modèles de lancement et les avantages qu'ils offrent, consultez Amazon EC2 Auto Scaling n'ajoutera plus la prise en charge des nouvelles fonctionnalités EC2 aux configurations de lancement sur AWS le Compute Blog.

> **Important**
>
> Il n'est pas nécessaire de suivre la procédure décrite dans cet article de blog pour effectuer la transition d'un ancien environnement vers des modèles de lancement. Pour migrer un environnement Elastic Beanstalk existant vers des modèles de lancement, définissez l'une des options répertoriées dans. Paramètres des options pour les modèles de lancement

Support des instances Spot pour votre environnement Elastic Beanstalk

Cette rubrique décrit les options de configuration disponibles pour gérer la capacité et l'équilibrage de charge des instances Spot dans votre environnement Elastic Beanstalk. Il fournit également des détails et des exemples des méthodes que vous pouvez utiliser pour configurer ces options. Vous pouvez utiliser la console Elastic Beanstalk, les options de configuration de l'espace de noms, ou AWS CLIla CLI EB pour gérer les options de configuration.

Minimisez les interruptions des instances Spot grâce au rééquilibrage des capacités

Pour minimiser l'impact des interruptions des instances Spot sur votre application, vous pouvez activer l'option Capacity Rebalancing incluse dans Amazon EC2 Auto Scaling.

> **Important**
>
> La demande d'instances Spot peut varier considérablement d'un instant à l'autre, de même que la disponibilité d'instances Spot peut aussi considérablement fluctuer selon le nombre d'instances Amazon EC2 inutilisées disponibles. Par ailleurs, votre instance Spot est susceptible d'être interrompue.

Lorsque vous activez le rééquilibrage des capacités, EC2 tente automatiquement de remplacer les instances Spot dans un groupe Auto Scaling avant qu'elles ne soient interrompues. Pour activer cette fonction, configurez le groupe Auto Scaling dans la console Elastic Beanstalk Vous pouvez également définir la valeur de l'option de configuration `EnableCapacityRebalancing` Elastic Beanstalk sur `true`dans l'espace de noms aws:autoscaling:asg.

Pour plus d'informations, consultez Capacity Rebalancing dans le guide de l'utilisateur Amazon EC2 Auto Scaling et Spot Instance Interruptions dans le guide de l'utilisateur Amazon EC2.

Anciens types d'instances et support pour les instances Spot

Certains AWS comptes plus anciens peuvent fournir à Elastic Beanstalk des types d'instances par défaut qui ne sont pas compatibles avec les instances Spot. Si vous activez les demandes d'instance Spot et que le message d'erreur Aucun des types d'instance que vous avez spécifiés ne prend en charge les instances Spot, mettez à jour votre configuration avec des types d'instances compatibles avec les instances Spot. Pour choisir des types d'instance Spot, utilisez Spot Instance Advisor.

Rubriques

- Activation des instances Spot pour votre environnement
- Stratégie d'allocation d'instances Spot
- Gestion des instances à la demande et des instances Spot
- Configuration des capacités pour votre environnement Elastic Beanstalk

Activation des instances Spot pour votre environnement

Pour tirer parti des instances Spot Amazon EC2, définissez l'`EnableSpot`option pour votre environnement. Le groupe Auto Scaling de votre environnement combine ensuite des options d'achat Amazon EC2 et gère un mélange d'instances à la demande et d'instances Spot.

Vous pouvez utiliser la console Elastic Beanstalk, les options de configuration de l'espace de noms, ou AWS CLIla CLI EB pour activer les demandes d'instance Spot pour votre environnement.

Avant d'activer les instances Spot pour votre environnement, familiarisez-vous avec les options de configuration d'Auto Scaling, de capacité et d'équilibrage de charge disponibles. Les exigences de votre application liées à la charge de travail, à l'impact des interruptions d'instance et à la tarification sont toutes des considérations importantes lorsque vous planifiez l'activation des instances Spot.

Les rubriques suivantes fournissent des informations détaillées sur les options d'Auto Scaling et de gestion des capacités, ainsi que sur l'impact de leur utilisation combinée sur votre environnement. Il existe des procédures et des exemples de configurations pour vous informer et vous guider sur les différentes options et sur la façon de les configurer. Nous proposons également des outils et des fonctionnalités pour vous aider à gérer votre configuration et à réagir aux événements. Vous pouvez planifier des modifications automatisées de votre configuration en fonction de périodes de trafic prévisibles, configurer des déclencheurs pour répondre à des facteurs tels que le volume de trafic, et configurer la surveillance et les bilans de santé Auto Scaling.

Pour des informations plus détaillées sur les instances Spot, notamment une explication des concepts clés et des meilleures pratiques, consultez la section Instances Spot dans le guide de l'utilisateur Amazon EC2.

> **Important**
>
> Le paramètre d'`EnableSpot`option peut amener Elastic Beanstalk à migrer un environnement existant avec des configurations de lancement vers des modèles de lancement. Cela nécessite les autorisations nécessaires pour gérer les modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si vous utilisez des politiques personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre environnement. Pour plus d'informations et d'autres considérations, consultezMigration de votre environnement Elastic Beanstalk pour lancer des modèles.

Stratégie d'allocation d'instances Spot

Vous pouvez sélectionner l'une des stratégies d'allocation répertoriées dans cette rubrique pour votre environnement Elastic Beanstalk. Utilisez la console Elastic Beanstalk, les options de configuration de l'espace de noms, AWS CLIou le, pour définir et configurer la stratégie d'allocation des instances Spot et les attributs associés pour votre environnement.

Amazon EC2 applique une stratégie d'allocation pour gérer et approvisionner des instances Spot pour votre environnement. Chaque stratégie d'allocation optimise les instances allouées en fonction de la façon dont elle est définie pour gérer la capacité disponible, le prix et la sélection des types d'instances.

Amazon EC2 Auto Scaling propose les stratégies d'allocation suivantes pour les instances Spot.

- Capacité optimisée (par défaut)
  - Demande des instances ponctuelles au pool, avec une capacité optimale compte tenu du nombre d'instances en cours de lancement.
  - Cette stratégie fonctionne bien pour les charges de travail où la possibilité d'interruption de service doit être minimisée.
- Prix et capacité optimisés
  - Demande des instances ponctuelles aux pools présentant le plus faible risque d'interruption et le prix le plus bas possible.
  - C'est le choix préférable pour la plupart des charges de travail Spot.
- Capacité optimisée priorisée
  - Demande d'abord des instances ponctuelles en fonction de la disponibilité des capacités, tout en respectant au mieux votre choix de priorisation des types d'instances. Vous pouvez fournir une liste de types d'instances, classés par priorité, lorsque vous configurez les options d'instance Spot pour Elastic Beanstalk.
  - Cette stratégie convient aux charges de travail qui nécessitent une interruption de service minimale et où la priorisation d'un type d'instance spécifique est importante.
- Prix le plus bas
  - Demande des instances Spot auprès du pool le moins cher avec les instances disponibles.
  - Il est important de prendre des précautions lors de l'utilisation de cette stratégie, car elle ne prend en compte que le prix de l'instance et non la disponibilité de la capacité, ce qui se traduira par des taux d'interruption élevés.

Pour plus de détails sur chaque stratégie d'allocation, consultez la section Stratégies d'allocation pour plusieurs types d'instances dans le manuel Amazon EC2 Auto Scaling User Guide.

Pour vous aider à comprendre quelle stratégie d'allocation est la mieux adaptée aux exigences de votre environnement, consultez Choisir la stratégie d'allocation ponctuelle appropriée dans le guide de l'utilisateur Amazon EC2.

Gestion des instances à la demande et des instances Spot

Vous pouvez lancer et mettre automatiquement à l'échelle une flotte d'instances à la demande et d'instances Spot au sein d'un même groupe Auto Scaling. Les options suivantes peuvent être utilisées conjointement pour configurer la façon dont le service Auto Scaling gère les instances ponctuelles et les instances à la demande dans votre environnement.

Vous pouvez configurer ces options pour votre environnement à l'aide de la console Elastic Beanstalk, des options de configuration de l'espace de nommage, de la ou de AWS CLIl'EB CLI.

Ces options font partie de l'espace de noms aws:ec2:instances :

- `EnableSpot`- Lorsque ce paramètre est défini sur `true` ce paramètre, les demandes d'instance Spot sont activées pour votre environnement.
- `SpotFleetOnDemandBase`- Définit le nombre minimum d'instances à la demande que votre groupe Auto Scaling fournit avant de prendre en compte les instances Spot à mesure que votre environnement évolue.
- `SpotFleetOnDemandAboveBasePercentage`- Le pourcentage d'instances à la demande faisant partie de la capacité supplémentaire que votre groupe Auto Scaling fournit au-delà des `SpotOnDemandBase` instances.

Les options répertoriées précédemment sont en corrélation avec les options suivantes de l'espace de noms aws:autoscaling:asg :

- `MinSize`- Le nombre minimum d'instances que vous souhaitez inclure dans votre groupe Auto Scaling.
- `MaxSize`- Le nombre maximum d'instances que vous souhaitez dans votre groupe Auto Scaling.

> **Important**
>
> Le paramètre d'`EnableSpot`option peut amener Elastic Beanstalk à migrer un environnement existant avec des configurations de lancement vers des modèles de lancement. Cela nécessite les autorisations nécessaires pour gérer les modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si vous utilisez des politiques personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre

environnement. Pour plus d'informations et d'autres considérations, consultezMigration de votre environnement Elastic Beanstalk pour lancer des modèles.

Appliquer les deux ensembles d'options d'espace de noms

Les points suivants décrivent comment la combinaison de ces paramètres d'options affecte la mise à l'échelle de votre environnement.

- Seule `MinSize` détermine la capacité initiale de votre environnement, c'est-à-dire le nombre d'instances que vous souhaitez exécuter au minimum.
- `SpotFleetOnDemandBase` n'affecte pas la capacité initiale. Lorsque Spot est activé, cette option détermine le nombre d'instances à la demande mises en service avant que les instances ponctuelles ne soient prises en compte.
- Imaginez quand `SpotFleetOnDemandBase` est inférieur à `MinSize`. Vous aurez toujours exactement `MinSize` instances comme capacité initiale. Au moins `SpotFleetOnDemandBase` d'entre elles doivent être des instances à la demande.
- Imaginez que `SpotFleetOnDemandBase` est supérieur à `MinSize`. À mesure de la mise à l'échelle de votre environnement, vous êtes assuré d'obtenir au minimum une quantité d'instances supplémentaires égale à la différence entre les deux valeurs. Autrement dit, vous êtes assuré d'obtenir au moins `(SpotFleetOnDemandBase - MinSize)` instances supplémentaires qui sont à la demande avant de satisfaire à l'exigence `SpotFleetOnDemandBase`.

Environnements d'instance unique

Dans les environnements de production, les instances Spot sont particulièrement utiles dans le cadre d'un environnement scalable et à charge équilibrée. Nous ne recommandons pas d'utiliser Spot dans un environnement à instance unique. Si les instances Spot ne sont pas disponibles, vous risquez de perdre toute la capacité (une instance unique) de votre environnement. Vous pouvez toujours souhaiter utiliser une instance Spot dans un environnement à instance unique pour le développement ou le test. Dans ce cas, assurez-vous de définir l `SpotFleetOnDemandBase` et `SpotFleetOnDemandAboveBasePercentage` à zéro. Tout autre paramétrage donne lieu à une instance à la demande.

Exemples de paramètres d'options de mise à l'échelle

Les exemples suivants illustrent différents scénarios de définition des différentes options de mise à l'échelle. Tous les exemples supposent qu'un environnement à charge équilibrée avec les demandes d'instances Spot activées a été défini.

Example 1 : Capacité initiale avec instances à la demande et instances Spot

Paramètres d'option

Option	Namespace	Valeur
`MinSize`	`aws:autoscaling:asg`	`10`
`MaxSize`	`aws:autoscaling:asg`	`24`
`SpotFleetOnDemandBase`	`aws:ec2:instances`	`4`
`SpotFleetOnDemandAboveBasePercentage`	`aws:ec2:instances`	`50`

Dans cet exemple, l'environnement commence avec dix instances, dont sept instances à la demande (quatre de base et 50 % des six autres au-delà) et trois instances Spot. L'environnement peut accepter jusqu'à 24 instances. Au fur et à mesure du dimensionnement, la proportion d'instances à la demande de la partie du parc au-delà des quatre instances à la demande de base est maintenue à 50 %, jusqu'à un maximum de 24 instances au total, dont 14 sont des instances à la demande (quatre de base et 50 % des 20 autres au-delà de la base) et 10 sont des instances Spot.

Example 2 : Capacité initiale avec uniquement des instances à la demande

Paramètres d'option

Option	Namespace	Valeur
`MinSize`	`aws:autoscaling:asg`	`4`
`MaxSize`	`aws:autoscaling:asg`	`24`

Option	Namespace	Valeur
SpotFleetOnDemandBase	aws:ec2:instances	4
SpotFleetOnDemandAboveBasePercentage	aws:ec2:instances	50

Dans cet exemple, l'environnement commence avec quatre instances à la demande. L'environnement peut accepter jusqu'à 24 instances. Au fur et à mesure du dimensionnement, la proportion d'instances à la demande de la partie du parc au-delà des quatre instances à la demande de base est maintenue à 50 %, jusqu'à un maximum de 24 instances au total, dont 14 sont des instances à la demande (quatre de base et 50 % des 20 autres au-delà de la base) et 10 sont des instances Spot.

Example 3 :Instances à la demande de base supplémentaires au-delà de la capacité initiale

Paramètres d'option

Option	Namespace	Valeur
MinSize	aws:autoscaling:asg	3
MaxSize	aws:autoscaling:asg	24
SpotFleetOnDemandBase	aws:ec2:instances	4
SpotFleetOnDemandAboveBasePercentage	aws:ec2:instances	50

Dans cet exemple, l'environnement commence avec trois instances à la demande. L'environnement peut accepter jusqu'à 24 instances. La première instance supplémentaire au-delà des trois instances initiales est une instance à la demande, pour compléter les quatre instances à la demande de base. Au fur et à mesure du dimensionnement, la proportion d'instances à la demande de la partie du parc au-delà des quatre instances à la demande de base est maintenue à 50 %, jusqu'à un maximum de 24 instances au total, dont 14 sont des instances à la demande (quatre de base et 50 % des 20 autres au-delà de la base) et 10 sont des instances Spot.

Configuration des capacités pour votre environnement Elastic Beanstalk

Cette rubrique décrit les différentes approches permettant de configurer la capacité Auto Scaling pour votre environnement Elastic Beanstalk. Vous pouvez utiliser la console Elastic Beanstalk, la AWS CLI CLI EB, les options ou les options d'espace de noms.

> **Important**
>
> Le paramètre d'`EnableSpot`option peut amener Elastic Beanstalk à migrer un environnement existant avec des configurations de lancement vers des modèles de lancement. Cela nécessite les autorisations nécessaires pour gérer les modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si vous utilisez des politiques personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre environnement. Pour plus d'informations et d'autres considérations, consultezMigration de votre environnement Elastic Beanstalk pour lancer des modèles.

Configuration via la console

Vous pouvez configurer la gestion des capacités d'un groupe Auto Scaling en modifiant Capacity sur la page Configuration de l'environnement dans la console Elastic Beanstalk.

Pour configurer la capacité du groupe Auto Scaling dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Auto Scaling group (Groupe Auto Scaling), configurez les paramètres suivants.

   - Environment type (Type d'environnement) – Sélectionnez Load balanced (Charge équilibrée).
   - Min instances (Nombre minimal d'instances) – Nombre minimal d'instances EC2 que le groupe doit contenir à tout moment. Le groupe démarre avec le nombre minimal et ajoute les instances quand la condition de déclenchement d'une augmentation est remplie.

- Max instances (Nombre maximal d'instances) – Nombre maximal d'instances EC2 que le groupe doit contenir à tout moment.

  > **Note**
  >
  > Si vous utilisez des mises à jour propagées, assurez-vous que le nombre maximal d'instances est supérieur au paramètre Instances minimums en service pour les mises à jour propagées.

- Composition de la flotte — La valeur par défaut est Instances à la demande. Pour activer les demandes d'instance Spot, sélectionnez Combined purchase options and instance types (Options d'achat et types d'instance combinés).

  > **Important**
  >
  > Le paramètre d'`EnableSpot`option peut amener Elastic Beanstalk à migrer un environnement existant avec des configurations de lancement vers des modèles de lancement. Cela nécessite les autorisations nécessaires pour gérer les modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si vous utilisez des politiques personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre environnement. Pour plus d'informations et d'autres considérations, consultezMigration de votre environnement Elastic Beanstalk pour lancer des modèles.

  Les options suivantes sont activées si vous choisissez d'activer les demandes d'instance Spot :

  - Stratégie d'allocation ponctuelle : détermine la méthode utilisée pour gérer et approvisionner les instances ponctuelles dans votre environnement, en fonction de la capacité disponible, du prix et de la sélection des types d'instances. Choisissez entre Capacité optimisée (par défaut), Prix, capacité optimisée, Capacité optimisée priorisée ou Prix le plus bas. Pour une description de chaque stratégie d'allocation et de plus amples informations, voirthe section called “Stratégie d'allocation au comptant”.
  - Prix spot maximum : pour obtenir des recommandations concernant les options de prix maximum pour les instances Spot, consultez l'historique des tarifs des instances Spot dans le guide de l'utilisateur Amazon EC2.

- On-Demande base (À la demande de base) – Nombre minimal d'instances à la demande que votre groupe Auto Scaling alloue avant de considérer les instances Spot à mesure que votre environnement grandit.
- On-Demand above base (À la demande au-dessus de la base) – Pourcentage d'instances à la demande dans le cadre de la capacité supplémentaire que votre groupe Auto Scaling alloue au-delà des instances à la demande de base.

> **Note**
>
> Les options On-Demand base (Base à la demande) et On-Demand above base (Base supérieure à la demande) sont en corrélation avec les options Instances Min et Max répertoriées plus tôt. Pour de plus amples informations sur ces options, consultez the section called “Support pour les instances Spot”.

- Rééquilibrage des capacités : cette option n'est pertinente que lorsqu'il existe au moins une instance Spot dans votre groupe Auto Scaling. Lorsque cette fonctionnalité est activée, EC2 tente automatiquement de remplacer les instances Spot du groupe Auto Scaling avant qu'elles ne soient interrompues, minimisant ainsi les interruptions des instances Spot dans vos applications. Pour plus d'informations, consultez la section Rééquilibrage des capacités dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling

- Architecture : architecture du processeur de vos instances EC2. L'architecture du processeur détermine les types d'instances EC2 disponibles dans le champ suivant.
- Types d'instances : types d'instances Amazon EC2 lancées pour exécuter votre application. Pour en savoir plus, consultez the section called “Types d’instances”.
- AMI ID (ID d'AMI) – Image machine utilisée par Elastic Beanstalk pour lancer des instances Amazon EC2 dans votre environnement. Pour en savoir plus, consultez the section called “ID d’AMI”.
- Availability Zones (Zones de disponibilité) – Choisissez le nombre de zones de disponibilité entre lesquelles répartir les instances de votre environnement. Par défaut, le groupe Auto Scaling lance les instances de manière uniforme sur l'ensemble des zones utilisables. Pour concentrer vos instances en moins de zones, choisissez le nombre de zones à utiliser. Pour les environnements de production, utilisez au moins deux zones afin de vous assurer que votre application est disponible en cas de panne de l'une des zones de disponibilité.
- Placement (facultatif) – Choisissez les zones de disponibilité à utiliser. Utilisez ce paramètre si vos instances ont besoin de se connecter aux ressources de zones spécifiques ou si vous avez acheté des instances réservées, qui sont spécifiques à une zone. Si vous lancez votre

environnement dans un VPC personnalisé, vous ne pouvez pas configurer cette option. Dans un VPC personnalisé, vous choisissez les zones de disponibilité des sous-réseaux que vous affectez à votre environnement.

- Scaling cooldown (Stabilisation de la mise à l'échelle) – Délai d'attente (en secondes) avant que les instances se lancent ou se terminent après la mise à l'échelle, avant de continuer à analyser les déclencheurs. Pour plus d'informations, consultez Stabilisation de la mise à l'échelle.

6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Configuration à l'aide des options d'espace de noms

Elastic Beanstalk fournit des options de configuration pour les paramètres Auto Scaling dans deux espaces de noms : `aws:autoscaling:asg` et `aws:ec2:instances`.

Espace de noms aws:autoscaling:asg

L'espace de noms `aws:autoscaling:asg` fournit des options pour le dimensionnement et la disponibilité d'ensemble.

L'exemple de fichier de configuration suivant configure le groupe Auto Scaling de manière à ce qu'il utilise de deux à quatre instances, des zones de disponibilité spécifiques et un temps de stabilisation de 12 minutes (720 secondes). Il permet le rééquilibrage de capacité pour les instances ponctuelles. Cette `EnableCapacityRebalancing` option ne prend effet que si elle `EnableSpot` est définie sur `true` dans l'espace de `aws:ec2:instances`noms, comme indiqué dans l'exemple de fichier de configuration suivant celui-ci.

```
option_settings:
  aws:autoscaling:asg:
    Availability Zones: Any
    Cooldown: '720'
    Custom Availability Zones: 'us-west-2a,us-west-2b'
    MaxSize: '4'
    MinSize: '2'
    EnableCapacityRebalancing: true
```

Espace de noms aws:ec2:instances

> **Note**
>
> Lorsque vous mettez à jour la configuration de votre environnement et supprimez un ou plusieurs types d'instance de l'option `InstanceTypes`, Elastic Beanstalk résilie toutes les instances Amazon EC2 exécutées sur les types d'instance supprimés. Le groupe Auto Scaling de votre environnement lance ensuite de nouvelles instances, si nécessaire pour compléter la capacité souhaitée, en utilisant les types d'instance spécifiés actuels.

L'espace de noms `aws:ec2:instances` fournit des options liées aux instances de votre environnement, y compris la gestion d'instances Spot. Il complète `aws:autoscaling:launchconfiguration` et `aws:autoscaling:asg`.

L'exemple de fichier de configuration suivant configure le groupe Auto Scaling pour activer les demandes d'instances Spot pour votre environnement. Il désigne trois types d'instances possibles qui peuvent être utilisés. Au moins une instance à la demande est utilisée pour la capacité de base, et 33 % soutenus des instances à la demande sont utilisés pour toute capacité supplémentaire.

La configuration définit la stratégie d'allocation des places sur`capacity-optimized-prioritized`. Cette stratégie d'allocation particulière donne la priorité aux lancements d'instances depuis le pool en fonction de l'ordre des types d'instances spécifiés dans l'`InstanceTypes`option. Si `SpotAllocationStrategy` ce n'est pas spécifié, la valeur par défaut est. `capacity-optimized`

```
option_settings:
  aws:ec2:instances:
    EnableSpot: true
    InstanceTypes: 't2.micro,t3.micro,t3.small'
    SpotAllocationStrategy: capacity-optimized-prioritized
    SpotFleetOnDemandBase: '1'
    SpotFleetOnDemandAbovePercentage: '33'
```

Pour choisir des types d'instance Spot, utilisez Spot Instance Advisor.

> **Important**
>
> Le paramètre d'`EnableSpot`option peut amener Elastic Beanstalk à migrer un environnement existant avec des configurations de lancement vers des modèles de

lancement. Cela nécessite les autorisations nécessaires pour gérer les modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si vous utilisez des politiques personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre environnement. Pour plus d'informations et d'autres considérations, consultezMigration de votre environnement Elastic Beanstalk pour lancer des modèles.

Configuration à l'aide du AWS CLI

Cette section fournit des exemples de la manière dont vous pouvez utiliser la commande AWS CLI create-environment pour configurer votre environnement avec les options Auto Scaling et Capacity décrites dans ces sections. Vous remarquerez que les paramètres d'espace de noms pour `aws:autoscaling:asg`et `aws:ec2:instances`, comme décrit dans la section précédente sur les options de configuration des espaces de noms, sont également configurés dans cet exemple.

L'interface de ligne de AWS commande fournit des commandes pour créer et configurer des environnements Elastic Beanstalk. Avec `--option-settings` cette option, vous pouvez transmettre des options d'espace de noms prises en charge par Elastic Beanstalk. Cela signifie que les options de configuration de l'espace de noms décrites précédemment peuvent être transmises aux AWS CLI commandes applicables pour configurer votre environnement Elastic Beanstalk.

> **Note**
>
> Vous pouvez également utiliser la commande update-environment avec `--option-settings` pour ajouter ou mettre à jour des options d'espace de noms. Si vous devez supprimer des options d'espace de noms de votre environnement, utilisez la update-environment commande avec`--options-to-remove`.

L'exemple suivant crée un nouvel environnement. Reportez-vous à la rubrique précédente Options de configuration de l'espace de noms pour plus de contexte sur les options transmises.

La première option répertoriée, `IamInstanceProfile` dans l'espace de aws:autoscaling:launchconfiguration noms, est le profil d'instance Elastic Beanstalk. C'est nécessaire lorsque vous créez un nouvel environnement.

Example— créer un environnement avec les options Auto Scaling (options d'espace de noms intégrées)

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v4.3.0 running Python 3.12" \
--option-settings \
Namespace=aws:autoscaling:launchconfiguration,OptionName=IamInstanceProfile,Value=aws-elasticbeanstalk-ec2-role
Namespace=aws:autoscaling:asg,OptionName=Availability Zones,Value=Any \
Namespace=aws:autoscaling:asg,OptionName=Cooldown,Value=720 \
Namespace=aws:autoscaling:asg,OptionName=Custom Availability Zones,Value=us-west-2a,us-west-2b \
Namespace=aws:autoscaling:asg,OptionName=MaxSize,Value=4 \
Namespace=aws:autoscaling:asg,OptionName=MinSize,Value=2 \
Namespace=aws:autoscaling:asg,OptionName=EnableCapacityRebalancing,Value=true \
Namespace=aws:ec2:instances,OptionName=EnableSpot,Value=true \
Namespace=aws:ec2:instances,OptionName=InstanceTypes,Value=t2.micro,t3.micro,t3.small \
Namespace=aws:ec2:instances,OptionName=SpotAllocationStrategy,Value=capacity-optimized-prioritized \
Namespace=aws:ec2:instances,OptionName=SpotFleetOnDemandBase,Value=1 \
Namespace=aws:ec2:instances,OptionName=SpotFleetOnDemandAboveBasePercentage,Value=33
```

> **Important**
>
> Le paramètre d'`EnableSpot`option peut amener Elastic Beanstalk à migrer un environnement existant avec des configurations de lancement vers des modèles de lancement. Pour ce faire, vous devez disposer des autorisations nécessaires pour gérer les modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si vous utilisez des politiques personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre environnement. Pour plus d'informations et d'autres considérations, consultezMigration de votre environnement Elastic Beanstalk pour lancer des modèles.

À titre de solution de rechange, utilisez un fichier `options.json` pour spécifier les options de l'espace de noms au lieu de les inclure en ligne.

Example—create-environment avec les options Auto Scaling (options d'espace de noms dans le fichier) options.json

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v4.3.0 running Python 3.12"
--option-settings file://options.json
```

Example

```
### example options.json ###
[
    {
        "Namespace": "aws:autoscaling:launchconfiguration",
        "OptionName": "IamInstanceProfile",
        "Value": "aws-elasticbeanstalk-ec2-role"
    },
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "Availability Zones",
        "Value": "Any"
    },
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "Cooldown",
        "Value": "720"
    },
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "Custom Availability Zones",
        "Value": "us-west-2a,us-west-2b"
    },
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "MaxSize",
        "Value": "4"
    },
    {
```

```
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "MinSize",
        "Value": "2"
    },
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "EnableCapacityRebalancing",
        "Value": "true"
    },
    {
        "Namespace": "aws:ec2:instances",
        "OptionName": "EnableSpot",
        "Value": "true"
    },
    {
        "Namespace": "aws:ec2:instances",
        "OptionName": "InstanceTypes",
        "Value": "t2.micro,t3.micro,t3.small"
    },
    {
        "Namespace": "aws:ec2:instances",
        "OptionName": "SpotAllocationStrategy",
        "Value": "capacity-optimized-prioritized"
    },
    {
        "Namespace": "aws:ec2:instances",
        "OptionName": "SpotFleetOnDemandBase",
        "Value": "1"
    },
    {
        "Namespace": "aws:ec2:instances",
        "OptionName": "SpotFleetOnDemandAboveBasePercentage",
        "Value": "33"
    }
]
```

Configuration à l'aide de l'EB CLI

Lorsque vous créez un environnement à l'aide de la commande eb create, vous pouvez spécifier quelques options liées au groupe Auto Scaling de votre environnement. Voici quelques-unes des options qui vous aident à contrôler la capacité de votre environnement.

`--single`

Crée l'environnement avec une instance Amazon EC2 et sans équilibreur de charge. Si vous n'utilisez pas cette option, un équilibreur de charge est ajouté à l'environnement qui a été créé.

`--enable-spot`

Active les demandes d'instances Spot pour votre environnement.

> **Important**
>
> Le paramètre d'`enable-spot`option peut amener Elastic Beanstalk à migrer un environnement existant avec des configurations de lancement vers des modèles de lancement. Pour ce faire, vous devez disposer des autorisations nécessaires pour gérer les modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si vous utilisez des politiques personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre environnement. Pour plus d'informations et d'autres considérations, consultezMigration de votre environnement Elastic Beanstalk pour lancer des modèles.

Les options suivantes de la commande eb create ne peuvent être utilisées qu'avec `--enable-spot`.

`--instance-types`

Liste les types d'instances Amazon EC2 que vous souhaitez que votre environnement utilise.

`--spot-max-price`

Prix maximum par heure d'unité, en dollars américains, que vous êtes prêt à payer pour une instance Spot. Pour obtenir des recommandations concernant les options de prix maximum pour les instances Spot, consultez l'historique des tarifs des instances Spot dans le guide de l'utilisateur Amazon EC2.

`--on-demand-base-capacity`

Nombre minimal d'instances à la demande que votre groupe Auto Scaling alloue avant de considérer les instances Spot à mesure que votre environnement grandit.

`--on-demand-above-base-capacity`

Pourcentage d'instances à la demande dans le cadre de la capacité supplémentaire allouée par votre groupe Auto Scaling supérieur au nombre d'instances qui est spécifié par l'option `--on-demand-base-capacity`.

L'exemple suivant crée un environnement et configure le groupe Auto Scaling de manière à activer les demandes d'instances Spot pour le nouvel environnement. Pour cet exemple, trois types d'instances possibles peuvent être utilisés.

```
$ eb create --enable-spot --instance-types "t2.micro,t3.micro,t3.small"
```

Important

Il existe une autre option nommée de la même manière qui s'appelle `--instance-type` (pas de « s »), que l'interface de ligne de commande (CLI) EB reconnaît uniquement lors du traitement des instances à la demande. N'utilisez pas `--instance-type` (pas de « s ») avec l'option `--enable-spot`. Si vous le faites, l'interface de ligne de commande (CLI) EB l'ignore. Utilisez plutôt `--instance-types` (avec « s ») avec l'option `--enable-spot`.

Déclencheurs Auto Scaling pour votre environnement Elastic Beanstalk

Le groupe Auto Scaling de votre environnement Elastic Beanstalk utilise CloudWatch deux alarmes Amazon pour déclencher des opérations de dimensionnement. Les déclencheurs par défaut évoluent quand le trafic réseau sortant moyen de chaque instance est supérieur à 6 Mo ou inférieur à 2 Mo sur une période de cinq minutes. Pour utiliser Amazon EC2 Auto Scaling de manière efficace, configurez des déclencheurs adaptés à votre application, à votre type d'instance et à vos exigences de service. Vous pouvez mettre à l'échelle en fonction de plusieurs statistiques, y compris la latence, les I/O disque, l'utilisation de l'UC et le nombre de demandes.

Pour plus d'informations sur CloudWatch les métriques et les alarmes, consultez Amazon CloudWatch Concepts dans le guide de CloudWatch l'utilisateur Amazon.

Configuration des déclencheurs Auto Scaling

Vous pouvez configurer les déclencheurs qui ajustent le nombre d'instances du groupe Auto Scaling de votre environnement dans la console Elastic Beanstalk.

Pour configurer des déclencheurs dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Déclencheurs de dimensionnement, configurez les paramètres suivants :

   - Métrique – Mesure utilisée pour votre déclencheur Auto Scaling.
   - Statistique – Calcul de statistiques que le déclencheur doit utiliser, comme `Average`.
   - Unité – Unité pour la métrique du déclencheur, comme Octets.
   - Période — Spécifie la fréquence à laquelle Amazon CloudWatch mesure les statistiques de votre déclencheur.
   - Breach duration (Durée de la faille) – Durée, en minutes, pendant laquelle une métrique peut se trouver en dehors des seuils supérieur et inférieur avant de déclencher une opération de mise à l'échelle.
   - Seuil supérieur – Si la métrique dépasse ce nombre pendant la durée de l'utilisation hors limites, une opération de mise à l'échelle est déclenchée.
   - Incrément de mise à l'échelle : nombre d' EC2 instances Amazon à ajouter lors de l'exécution d'une activité de dimensionnement.
   - Seuil inférieur – Si la métrique tombe en-dessous de ce nombre pendant la durée de l'utilisation hors limites, une opération de mise à l'échelle est déclenchée.
   - Diminuer l'incrément : nombre d' EC2 instances Amazon à supprimer lors d'une activité de dimensionnement.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Espace de noms aws:autoscaling:trigger

Elastic Beanstalk fournit des options de configuration pour les paramètres Auto Scaling dans l'espace de noms `aws:autoscaling:trigger`. Les paramètres de cet espace de noms sont organisés par la ressource à laquelle ils s'appliquent.

```
option_settings:
```

```
  AWSEBAutoScalingScaleDownPolicy.aws:autoscaling:trigger:
    LowerBreachScaleIncrement: '-1'
  AWSEBAutoScalingScaleUpPolicy.aws:autoscaling:trigger:
    UpperBreachScaleIncrement: '1'
  AWSEBCloudwatchAlarmHigh.aws:autoscaling:trigger:
    UpperThreshold: '6000000'
  AWSEBCloudwatchAlarmLow.aws:autoscaling:trigger:
    BreachDuration: '5'
    EvaluationPeriods: '1'
    LowerThreshold: '2000000'
    MeasureName: NetworkOut
    Period: '5'
    Statistic: Average
    Unit: Bytes
```

Actions Auto Scaling planifiées pour vos environnements Elastic Beanstalk

Pour optimiser l'utilisation des EC2 instances Amazon dans votre environnement pendant les périodes prévisibles de pic de trafic, configurez votre groupe Amazon EC2 Auto Scaling pour modifier son nombre d'instances selon un calendrier. Vous pouvez configurer votre environnement avec une action récurrente pour augmenter la capacité le matin et la réduire le soir, lorsque le trafic est faible. Par exemple, si une raison telle qu'un événement marketing amène du trafic vers votre site pendant une période limitée, vous pouvez planifier un événement unique pour augmenter la capacité au début de cet événement, et un autre pour le réduire à la fin.

Vous pouvez définir jusqu'à 120 actions planifiées actives par environnement. Elastic Beanstalk conserve également jusqu'à 150 actions planifiées expirées, que vous pouvez réutiliser en mettant à jour leurs paramètres.

Configuration d'actions planifiées

Vous pouvez créer des actions planifiées pour le groupe Auto Scaling de votre environnement dans la console Elastic Beanstalk.

Pour configurer des actions planifiées dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Time-based scaling (Dimensionnement basé sur la durée), choisissez Add scheduled action (Ajouter une action planifiée).
6. Renseignez les paramètres des actions planifiées suivants :

   - Nom – Indiquez un nom unique de 255 caractères alphanumériques au maximum et sans espace.
   - Instances – Choisissez le nombre minimum et maximum d'instances à appliquer au groupe Auto Scaling.
   - Capacité souhaitée (facultatif) – Définissez la capacité souhaitée initiale pour le groupe Auto Scaling. Lorsque l'action planifiée est appliquée, des déclencheurs ajustent la capacité souhaitée en fonction de leurs paramètres.
   - Occurrence – Choisissez Récurrent pour répéter l'action de mise à l'échelle selon un calendrier spécifique.
   - Heure de début – Pour des actions ponctuelles, choisissez la date et l'heure d'exécution de l'action.

     Pour des actions récurrentes, l'heure de début est facultative. Spécifiez-le pour choisir le moment le plus proche où l'action est exécutée. Après cette période, l'action se répète selon l'expression de la Récurrence.
   - Récurrence – Utilisez une expression Cron pour spécifier la fréquence à laquelle vous souhaitez que l'action planifiée se produise. Par exemple, `30 6 * * 2` exécute l'action tous les mardis à 6h30 UTC.
   - Heure de fin (facultatif) – Facultatif pour les actions récurrentes. Si elle est spécifiée, l'action se répète selon l'expression de la Récurrence et n'est plus exécutée après cette période.

     Lorsqu'une action planifiée se termine, Auto Scaling ne revient pas automatiquement à ses paramètres précédents. Configurez une seconde action planifiée pour rétablir les paramètres d'origine d'Auto Scaling selon vos besoins.
7. Choisissez Ajouter.
8. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

> **Note**
>
> Les actions planifiées ne sont pas enregistrées tant qu'elles ne sont pas appliquées.

Espace de noms aws:autoscaling:scheduledaction

Si vous devez configurer un grand nombre d'actions planifiées, vous pouvez utiliser les fichiers de configuration ou l'API Elastic Beanstalk pour appliquer les modifications d'option de configuration à partir d'un fichier YAML ou JSON. Ces méthodes vous permettent également d'accéder à l'option `Suspend` pour désactiver temporairement une action planifiée récurrente.

> **Note**
>
> Lorsque vous utilisez des options de configuration d'action planifiée en dehors de la console, utilisez le format d'heure ISO 8601 pour spécifier les heures de début et de fin en UTC. Par exemple, 2015-04-28T04:07:02Z. Pour plus d'informations sur le format horaire ISO 8601, accédez à Date and Time Formats. Les dates doivent être uniques dans toutes les actions planifiées.

Elastic Beanstalk fournit des options de configuration pour les paramètres d'action planifiée dans l'espace de noms `aws:autoscaling:scheduledaction`. Utilisez le champ `resource_name` pour spécifier le nom de l'action planifiée.

Example S cheduled-scale-up-specific -time-long.config

Ce fichier de configuration indique à Elastic Beanstalk de monter en puissance de 5 à 10 instances à cette date : 2015-12-12T00:00:00Z.

```
option_settings:
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledScaleUpSpecificTime
    option_name: MinSize
    value: '5'
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledScaleUpSpecificTime
    option_name: MaxSize
    value: '10'
```

```
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledScaleUpSpecificTime
    option_name: DesiredCapacity
    value: '5'
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledScaleUpSpecificTime
    option_name: StartTime
    value: '2015-12-12T00:00:00Z'
```

Example S cheduled-scale-up-specific -time.config

Pour utiliser la syntaxe abrégée avec la CLI EB ou les fichiers de configuration, ajoutez le nom de la ressource au début de l'espace de noms.

```
option_settings:
  ScheduledScaleUpSpecificTime.aws:autoscaling:scheduledaction:
    MinSize: '5'
    MaxSize: '10'
    DesiredCapacity: '5'
    StartTime: '2015-12-12T00:00:00Z'
```

Example S cheduled-scale-down-specific -time.config

Ce fichier de configuration indique à Elastic Beanstalk de diminuer en puissance à 2015-12-12T 07:00:00 Z.

```
option_settings:
  ScheduledScaleDownSpecificTime.aws:autoscaling:scheduledaction:
    MinSize: '1'
    MaxSize: '1'
    DesiredCapacity: '1'
    StartTime: '2015-12-12T07:00:00Z'
```

Example S cheduled-periodic-scale-up .config

Ce fichier de configuration indique à Elastic Beanstalk de monter en puissance tous les jours à 9h du matin. L'action doit commencer le 14 mai 2015 et se terminer le 12 janvier 2016.

```
option_settings:
  ScheduledPeriodicScaleUp.aws:autoscaling:scheduledaction:
    MinSize: '5'
    MaxSize: '10'
```

```
    DesiredCapacity: '5'
    StartTime: '2015-05-14T07:00:00Z'
    EndTime: '2016-01-12T07:00:00Z'
    Recurrence: 0 9 * * *
```

Example S cheduled-periodic-scale-down .config

Ce fichier de configuration indique à Elastic Beanstalk de diminuer en puissance afin de pas exécuter d'instance chaque jour à 18 h. Si vous savez que votre application est surtout inactive en dehors des heures d'ouverture, vous pouvez créer une action planifiée similaire. Si votre application doit être arrêtée en dehors des heures d'ouverture, définissez `MaxSize` sur `0`.

```
option_settings:
  ScheduledPeriodicScaleDown.aws:autoscaling:scheduledaction:
    MinSize: '0'
    MaxSize: '1'
    DesiredCapacity: '0'
    StartTime: '2015-05-14T07:00:00Z'
    EndTime: '2016-01-12T07:00:00Z'
    Recurrence: 0 18 * * *
```

Example S cheduled-weekend-scale-down .config

Ce fichier de configuration indique à Elastic Beanstalk de mettre à l'échelle tous les vendredis à 18h. Si vous savez que votre application ne reçoit pas autant de trafic au cours du week-end, vous pouvez créer une action planifiée similaire.

```
option_settings:
  ScheduledWeekendScaleDown.aws:autoscaling:scheduledaction:
    MinSize: '1'
    MaxSize: '4'
    DesiredCapacity: '1'
    StartTime: '2015-12-12T07:00:00Z'
    EndTime: '2016-01-12T07:00:00Z'
    Recurrence: 0 18 * * 5
```

Paramètre de vérification de l'état Auto Scaling pour votre environnement Elastic Beanstalk

Amazon EC2 Auto Scaling surveille l'état de santé de chaque instance Amazon Elastic Compute Cloud (Amazon EC2) qu'il lance. Si une instance est résiliée de façon inattendue, Auto Scaling

détecte cette résiliation et lance une instance de remplacement. Par défaut, le groupe Auto Scaling créé pour votre environnement utilise les contrôles de EC2 statut Amazon. Si une instance de votre environnement échoue à la vérification EC2 du statut d'Amazon, Auto Scaling la retire et la remplace.

Les vérifications de EC2 statut Amazon ne portent que sur l'état de santé d'une instance, et non sur l'état de votre application, de votre serveur ou de tout conteneur Docker exécuté sur l'instance. Si votre application se bloque, mais que l'instance sur laquelle elle s'exécute reste saine, elle peut être exclue de l'équilibreur de charge, mais elle ne sera pas automatiquement remplacée par Auto Scaling. Le comportement par défaut convient pour la résolution des problèmes. Si Auto Scaling a remplacé l'instance dès le blocage de l'application, il est possible que vous ne vous rendiez pas compte du problème, même si le blocage s'est produit peu de temps après le démarrage.

Si vous souhaitez qu'Auto Scaling remplace les instances dont l'application a cessé de répondre, vous pouvez utiliser un fichier de configuration pour configurer le groupe Auto Scaling afin qu'il utilise les vérifications de l'état d'Elastic Load Balancing. L'exemple suivant montre comment configurer le groupe pour qu'il utilise les vérifications de santé de l'équilibreur de charge, en plus de la vérification du EC2 statut Amazon, pour déterminer l'état de santé d'une instance.

Example.ebextensions/autoscaling.config

```
Resources:
  AWSEBAutoScalingGroup:
    Type: "AWS::AutoScaling::AutoScalingGroup"
    Properties:
      HealthCheckType: ELB
      HealthCheckGracePeriod: 300
```

Pour plus d'informations sur les `HealthCheckGracePeriod` propriétés `HealthCheckType` et les propriétés, consultez AWS::AutoScaling::AutoScalingGroup dans le guide de l'AWS CloudFormation utilisateur et Health Checks for Auto Scaling Instances dans le guide de l'utilisateur Amazon EC2 Auto Scaling.

Par défaut, la vérification de l'état Elastic Load Balancing est configurée pour tenter une connexion TCP vers votre instance via le port 80. Cela confirme que le serveur web s'exécutant sur l'instance accepte les connexions. Cependant, il se peut que vous souhaitiez personnaliser la vérification de l'état de l'équilibreur de charge afin de vous assurer que votre application, et non pas seulement le serveur web, se trouve dans un état correct. Le paramètre de période de grâce définit le laps de temps (en secondes) pendant lequel une instance peut échouer aux vérifications de l'état sans être

suspendue ou remplacée. Comme les instances peuvent être restaurées après avoir été exclues de l'équilibreur de charge, veuillez attribuer à votre instance un laps de temps adapté à votre application.

Équilibreur de charge pour votre environnement Elastic Beanstalk

L'équilibreur de charge distribue le trafic entre les instances de votre environnement. Lorsque vous activez l'équilibrage de charge, vous AWS Elastic Beanstalk créez un équilibreur de charge Elastic Load Balancing dédié à votre environnement. Elastic Beanstalk gère entièrement cet équilibreur de charge, en prenant soin des paramètres de sécurité et en veillant à résilier l'équilibreur de charge lorsque vous arrêtez votre environnement.

Vous pouvez également choisir de partager un équilibreur de charge entre plusieurs environnements Elastic Beanstalk. Avec un équilibreur de charge partagé, vous économisez sur les coûts opérationnels en évitant d'utiliser un équilibreur de charge dédié pour chaque environnement. Vous assumez également davantage la responsabilité de gestion de l'équilibreur de charge partagé que vos environnements utilisent.

Les types d'équilibreur de charge d'Elastic Load Balancing sont les suivants :

- Classic Load Balancer – Équilibreur de charge de la génération précédente. Achemine le trafic de demandes HTTP, HTTPS ou TCP vers différents ports sur des instances d'environnement.
- Application Load Balancer – Équilibreur de charge de couche d'application. Achemine le trafic de demandes HTTP ou HTTPS vers différents ports sur des instances d'environnement en fonction du chemin d'accès de la demande.
- Network Load Balancer – Équilibreur de charge de couche réseau. Achemine le trafic de demandes TCP vers différents ports sur des instances d'environnement. Prend en charge les vérifications de l'état actives et passives.

Elastic Beanstalk prend en charge les trois types d'équilibreur de charge. Le tableau suivant indique les types que vous pouvez utiliser avec les deux modèles d'utilisation :

Type d'équilibreur de charge	Dédié	Partagé
Classic Load Balancer	✓ Oui	✕ Non
Application Load Balancer	✓ Oui	✓ Oui
Network Load Balancer	✓ Oui	✕ Non

> **Note**
>
> L'option Classic Load Balancer (CLB) est désactivée dans l'assistant de création d'un environnement de la console. Si vous disposez d'un environnement existant configuré avec un Classic Load Balancer, vous pouvez en créer un nouveau en clonant l'environnement existant à l'aide de la console Elastic Beanstalk ou de l'interface de ligne de commande EB. Vous avez également la possibilité d'utiliser l'EB CLI ou AWS CLIde créer un nouvel environnement configuré avec un Classic Load Balancer. Ces outils de ligne de commande créeront un nouvel environnement avec un CLB, même s'il n'en existe pas déjà un dans votre compte.

Par défaut, Elastic Beanstalk crée un Application Load Balancer pour votre environnement lorsque vous activez l'équilibrage de charge avec la console Elastic Beanstalk ou l'interface de ligne de commande EB. Il configure l'équilibreur de charge pour écouter le trafic HTTP sur le port 80 et transmettre ce trafic aux instances sur le même port. Vous pouvez choisir le type d'équilibreur de charge que votre environnement utilise uniquement lors de la création de l'environnement. Ultérieurement, vous pourrez modifier les paramètres pour gérer le comportement de l'équilibreur de charge de votre environnement d'exécution, mais pas en changer le type.

> **Note**
>
> Votre environnement doit se trouver dans un VPC avec des sous-réseaux dans au moins deux zones de disponibilité pour créer un Application Load Balancer. Tous les nouveaux AWS comptes incluent des comptes par défaut VPCs qui répondent à cette exigence.

Consultez les rubriques suivantes pour en savoir plus sur chaque type d'équilibreur de charge pris en charge par Elastic Beanstalk, sur ses fonctionnalités et sur la façon de le configurer et de le gérer dans un environnement Elastic Beanstalk, ainsi que pour découvrir comment configurer un équilibreur de charge pour qu'il charge les journaux d'accès dans Amazon S3.

Rubriques

- Configuration d'un Classic Load Balancer
- Configuration d'un Application Load Balancer
- Configuration d'un Application Load Balancer partagé
- Configuration d'un Network Load Balancer

- Configuration des équilibreurs de charge Elastic Beanstalk à double pile
- Configuration des journaux d'accès

Configuration d'un Classic Load Balancer

Lorsque vous activez l'équilibrage de charge, votre AWS Elastic Beanstalk environnement est équipé d'un équilibreur de charge Elastic Load Balancing pour répartir le trafic entre les instances de votre environnement. Elastic Load Balancing prend en charge plusieurs types d'équilibreur de charge. Pour en savoir plus, consultez le Guide de l'utilisateur Elastic Load Balancing. Elastic Beanstalk peut créer un équilibreur de charge pour vous, ou vous permettre de spécifier un équilibreur de charge partagé que vous avez créé.

Cette rubrique décrit la configuration d'un équilibreur de charge Classic Load Balancer créé par Elastic Beanstalk et dédié à votre environnement. Pour plus d'informations sur la configuration de tous les types d'équilibreur de charge pris en charge par Elastic Beanstalk, consultez Équilibreur de charge pour votre environnement Elastic Beanstalk.

> **Note**
>
> Vous pouvez choisir le type d'équilibreur de charge que votre environnement utilise uniquement lors de la création de l'environnement. Ultérieurement, vous pourrez modifier les paramètres pour gérer le comportement de l'équilibreur de charge de votre environnement d'exécution, mais pas en changer le type.

Introduction

Un Classic Load Balancer est l'équilibreur de charge Elastic Load Balancing de génération précédente. Il prend en charge l'acheminement du trafic des demandes HTTP, HTTPS ou TCP vers différents ports sur des instances d'environnement.

Lorsque votre environnement utilise un Classic Load Balancer, Elastic Beanstalk le configure par défaut pour écouter le trafic HTTP sur le port 80 et le transférer aux instances du même port. Bien que vous ne puissiez pas supprimer l'écouteur par défaut du port 80, vous pouvez le désactiver, ce qui permet d'obtenir les mêmes fonctionnalités en bloquant le trafic. Notez que vous pouvez ajouter ou supprimer d'autres écouteurs. Pour prendre en charge les connexions sécurisées, vous pouvez configurer votre équilibreur de charge avec un écouteur sur le port 443 et un certificat TLS.

L'équilibreur de charge utilise un contrôle de santé pour déterminer si les EC2 instances Amazon exécutant votre application sont saines. La vérification de l'état envoie une demande à une URL spécifiée selon un intervalle défini. Si l'URL renvoie un message d'erreur ou ne répond pas dans un délai spécifique, la vérification de l'état échoue.

Si votre application enregistre de meilleures performances en traitant plusieurs demandes à partir du même client sur un seul serveur, vous pouvez configurer votre équilibreur de charge pour utiliser des séances permanentes. Dans le cas des sessions persistantes, l'équilibreur de charge ajoute un cookie aux réponses HTTP qui identifie l' EC2 instance Amazon qui a répondu à la demande. Lorsqu'une demande ultérieure est reçue du même client, l'équilibreur de charge utilise le cookie pour envoyer la demande à la même instance.

Avec l'équilibrage de charge entre zones, chaque nœud de l'équilibreur de charge pour votre Classic Load Balancer répartit les demandes uniformément entre les instances enregistrées dans toutes les zones de disponibilité activées. Si l'équilibrage de charge entre zones est désactivé, chaque nœud de l'équilibreur de charge répartit les demandes uniformément entre les instances enregistrées dans sa zone de disponibilité uniquement.

Lorsqu'une instance est supprimée de l'équilibreur de charge parce qu'elle n'est plus saine ou parce que la capacité de l'environnement est diminuée, Connection Draining laisse à l'instance le temps de terminer les demandes avant de fermer la connexion entre l'instance et l'équilibreur de charge. Vous pouvez modifier le laps de temps accordé aux instances pour l'envoi d'une réponse ou désactiver totalement le drainage de la connexion.

> **Note**
>
> Connection Draining est activé par défaut lorsque vous créez un environnement via la console Elastic Beanstalk ou l'interface de ligne de commande EB. Pour les autres clients, vous pouvez l'activer via les options de configuration.

Vous pouvez utiliser les paramètres avancés de l'équilibreur de charge pour configurer des écouteurs sur des ports arbitraires, modifier des paramètres de session persistants supplémentaires et configurer l'équilibreur de charge pour qu'il se connecte aux EC2 instances en toute sécurité. Ces paramètres sont disponibles via les options de configuration que vous pouvez définir avec des fichiers de configuration dans votre code source, ou directement sur un environnement via l'API Elastic Beanstalk. Un grand nombre de ces paramètres sont également disponibles dans la console

Elastic Beanstalk. Vous pouvez également configurer un équilibreur de charge pour charger les journaux d'accès dans Amazon S3.

Configuration d'un Classic Load Balancer à l'aide de la console Elastic Beanstalk

Vous pouvez utiliser la console Elastic Beanstalk pour configurer le port, le certificat HTTPS et d'autres paramètres d'un Classic Load Balancer lors de la création de l'environnement ou plus tard, lorsque votre environnement est en cours d'exécution.

> **Note**
>
> L'option Classic Load Balancer (CLB) est désactivée dans l'assistant de création d'un environnement de la console. Si vous disposez d'un environnement existant configuré avec un Classic Load Balancer, vous pouvez en créer un nouveau en clonant l'environnement existant à l'aide de la console Elastic Beanstalk ou de l'interface de ligne de commande EB. Vous avez également la possibilité d'utiliser l'EB CLI ou AWS CLIde créer un nouvel environnement configuré avec un Classic Load Balancer. Ces outils de ligne de commande créeront un nouvel environnement avec un CLB, même s'il n'en existe pas déjà un dans votre compte.

Pour configurer un Classic Load Balancer d'un environnement en cours d'exécution dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Load balancer (Équilibreur de charge), choisissez Edit (Modifier).

   > **Note**
   >
   > Si la catégorie de configuration Load balancer (Équilibreur de charge) ne dispose pas du bouton Edit (Modifier), cela signifie que votre environnement ne dispose pas d'un

équilibreur de charge. Pour apprendre à en configurer un, consultez Changement de type d'environnement.

5. Effectuez les modifications de configuration du Classic Load Balancer exigées par votre environnement.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

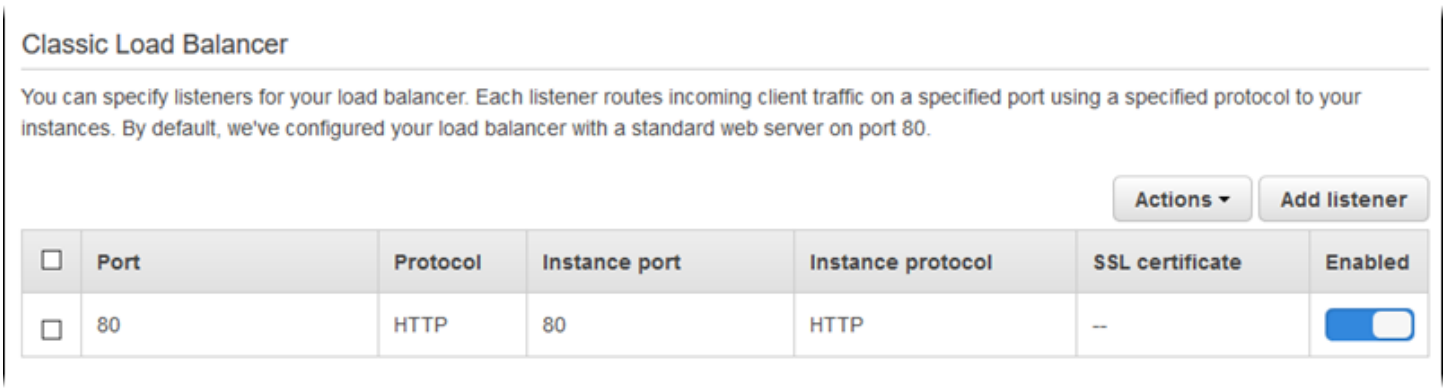
Paramètres du Classic Load Balancer

- Écouteurs
- Séances
- Equilibrage de charge entre zones
- Drainage de la connexion
- Surveillance de l'état

Écouteurs

Utilisez cette liste pour spécifier plusieurs écouteurs pour votre équilibreur de charge. Chaque écouteur achemine le trafic client entrant sur un port spécifié à l'aide d'un protocole spécifié vers vos instances. Au départ, la liste affiche l'écouteur par défaut, lequel achemine le trafic HTTP entrant sur le port 80 vers les serveurs d'instance de votre environnement qui écoutent le trafic HTTP sur le port 80.

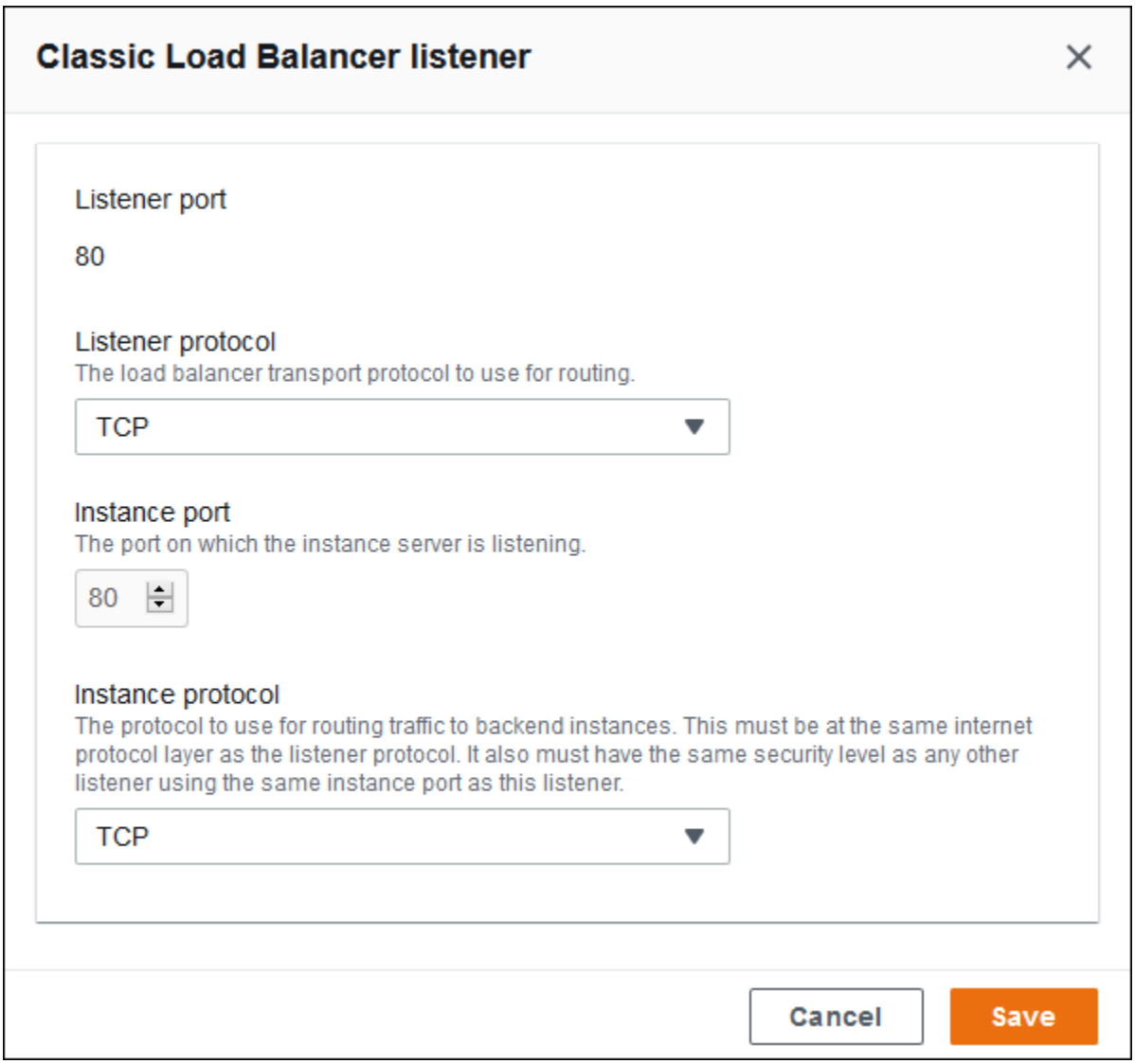
> **Note**
>
> Bien que vous ne puissiez pas supprimer l'écouteur par défaut du port 80, vous pouvez le désactiver, ce qui permet d'obtenir les mêmes fonctionnalités en bloquant le trafic.



Pour configurer un écouteur existant

1. Cochez la case en regard de son entrée de table, puis choisissez Actions et sélectionnez l'action de votre choix.
2. Si vous choisissez Modifier, utilisez la boîte de dialogue écouteur Classic Load Balancer pour modifier les paramètres, puis choisissez Enregistrer.

Par exemple, vous pouvez modifier l'écouteur par défaut et remplacer la valeur du paramètre Protocole, HTTP, par TCP si vous souhaitez que l'équilibreur de charge transfère la demande en l'état. Ce paramètre évite que l'équilibreur de charge réécrive les en-têtes (y compris `X-Forwarded-For`). La technique ne fonctionne pas avec les sessions permanentes.



Pour ajouter un écouteur

1. Choisissez Add listener (Ajouter un écouteur).
2. Dans la boîte de dialogue écouteur Classic Load Balancer, configurez les paramètres de votre choix, puis sélectionnez Ajouter.

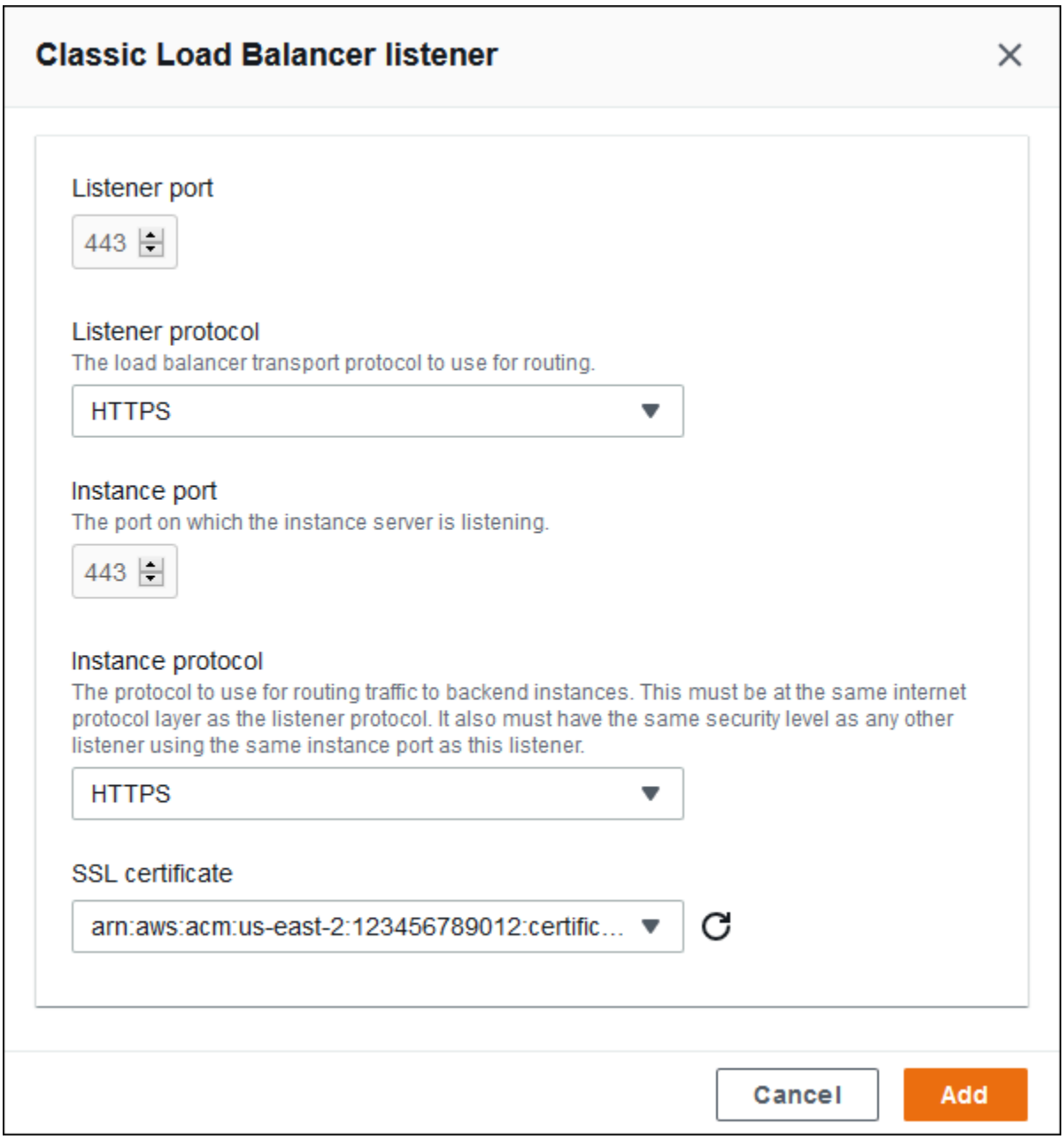
L'ajout d'un écouteur sécurisé est une cas d'utilisation courant. L'exemple présenté dans l'image qui suit illustre l'ajout d'un écouteur pour le trafic HTTPS sur le port 443. Cet écouteur achemine le trafic entrant vers les serveurs d'instances de l'environnement qui écoutent le trafic HTTPS sur le port 443.

Pour pouvoir configurer un écouteur HTTPS, assurez-vous que vous disposez d'un certificat SSL valide. Effectuez l’une des actions suivantes :

- Si AWS Certificate Manager (ACM) est disponible dans votre AWS région, créez ou importez un certificat à l'aide d'ACM. Pour plus d'informations sur une demande de certificat ACM, consultez Request a Certificate (Demande de certificat) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur AWS Certificate Manager). Pour plus d'informations sur l'importation de certificats tiers dans ACM, consultez Importing Certificates (Importation de certificats) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur AWS Certificate Manager).
- Si ACM n'est pas disponible dans votre AWS région, téléchargez votre certificat et votre clé existants dans IAM. Pour plus d'informations sur la création et le chargement des certificats dans IAM, consultez Utilisation des certificats de serveur dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur la configuration HTTPS et l'utilisation des certificats dans Elastic Beanstalk, consultez Configuration de HTTPS pour votre environnement Elastic Beanstalk.

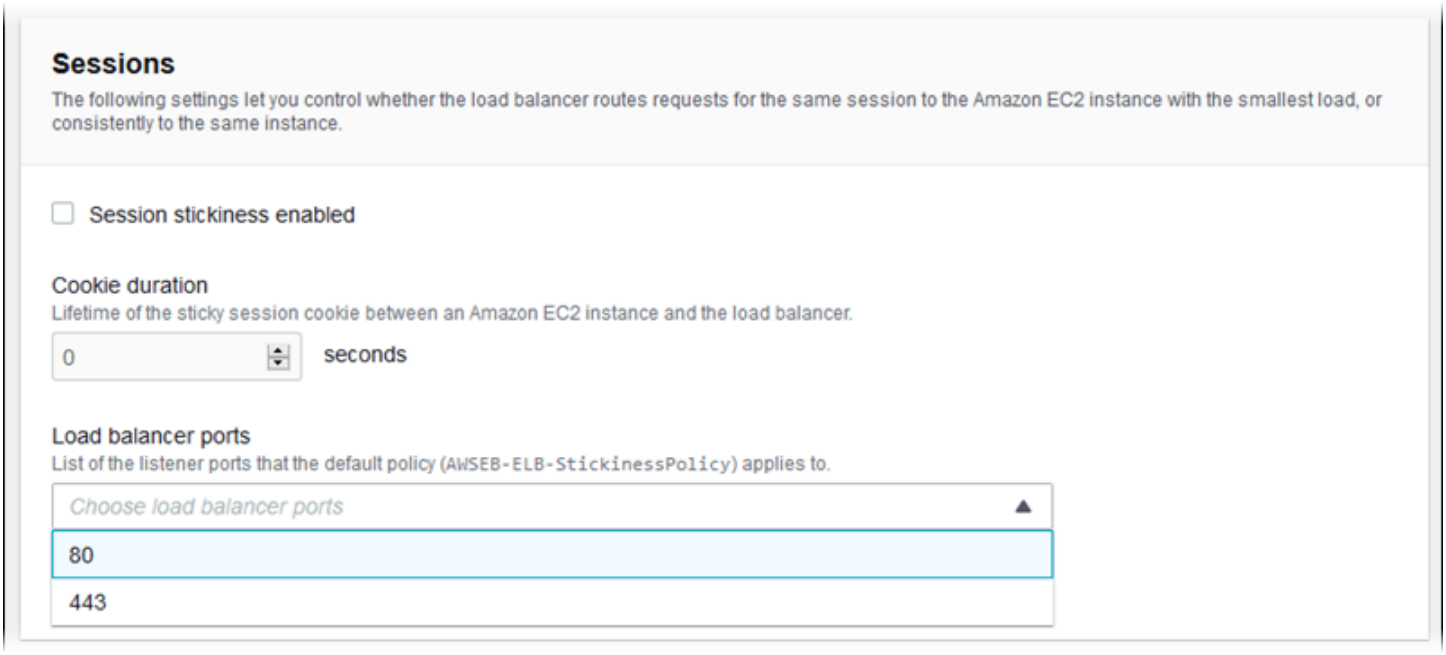
Pour Certificat SSL, choisissez l'ARN de votre certificat SSL. Par exemple, `arn:aws:iam::123456789012:server-certificate/abc/certs/build` ou `arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678`.



Pour plus d'informations sur la configuration HTTPS et l'utilisation des certificats dans Elastic Beanstalk, consultez Configuration de HTTPS pour votre environnement Elastic Beanstalk.

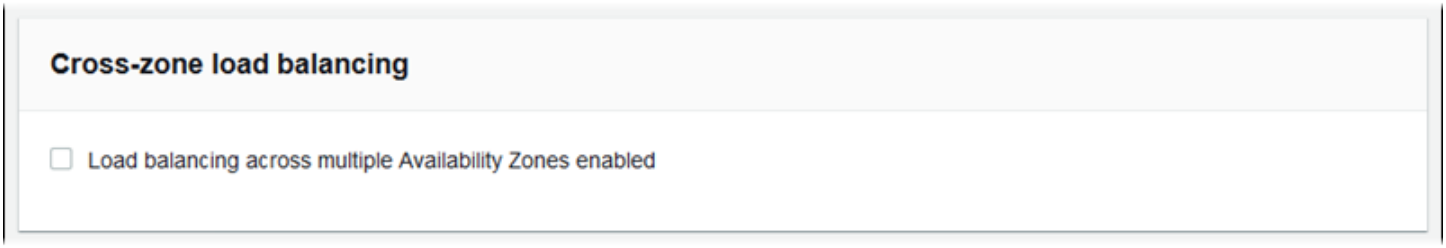
Séances

Activez ou désactivez la case Permanence de la séance activée pour activer ou désactiver les sessions permanentes. Utilisez l'option Durée du cookie pour configurer la durée d'une session permanente, pouvant aller jusqu'à **`1000000`** secondes. Dans la liste Load balancer ports (Ports de l'équilibreur de charge, sélectionnez les ports écouteurs auxquels la politique par défaut (`AWSEB-ELB-StickinessPolicy`) s'applique.



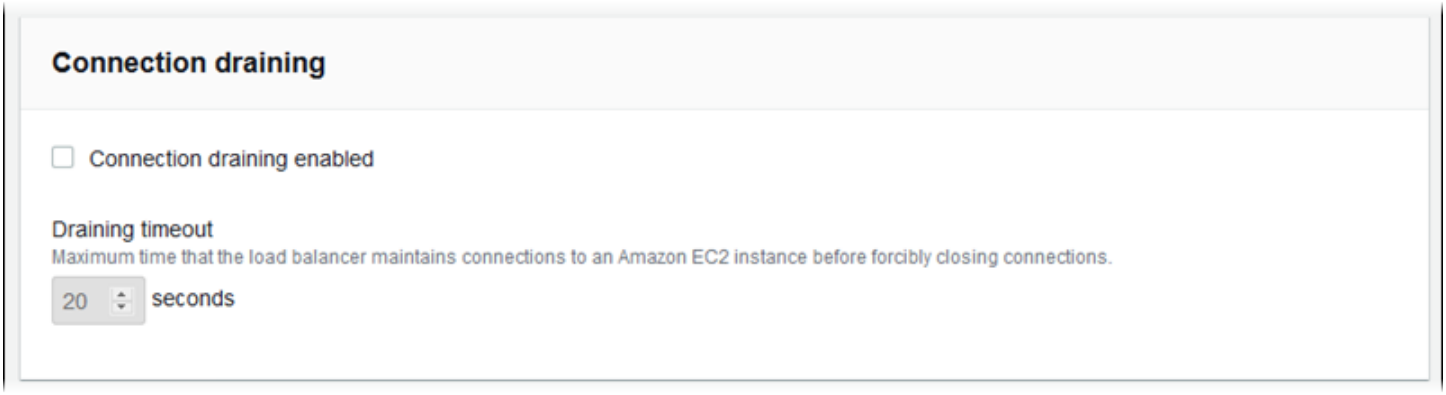
Equilibrage de charge entre zones

Cochez ou décochez Équilibrage de charge entre plusieurs zones de disponibilité activé pour activer ou désactiver l'équilibrage de charge entre zones.



Drainage de la connexion

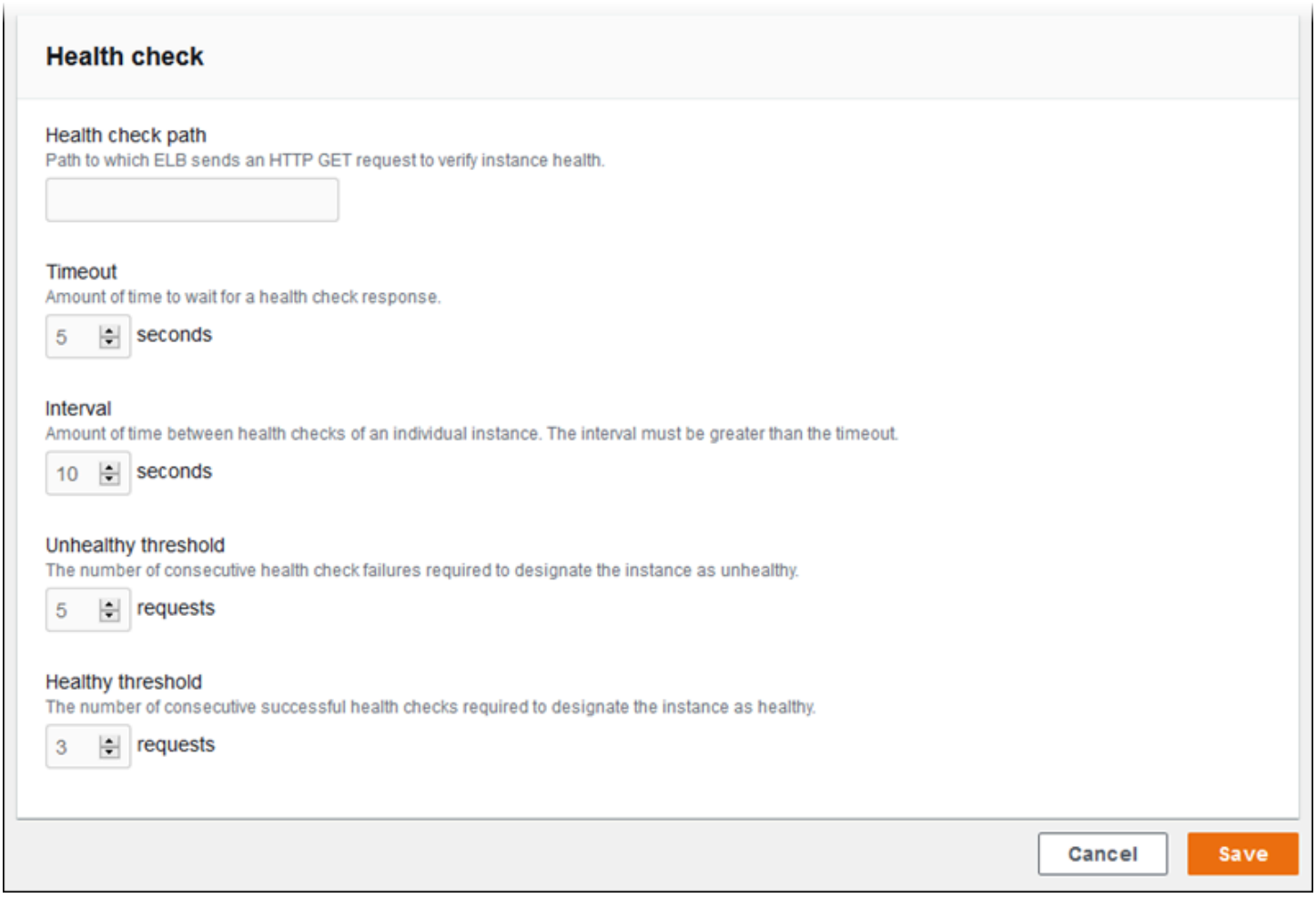
Sélectionnez ou désélectionnez la case Drainage de la séance activée pour activer ou désactiver le drainage de la connexion. Définissez le délai de drainage, pouvant aller jusqu'à **`3600`** secondes.



Surveillance de l'état

Utilisez les paramètres suivants pour configurer les vérifications de l'état de l'équilibreur de charge :

- Health check path (Chemin de vérification de l'état) – Chemin d'accès vers lequel l'équilibreur de charge envoie les demandes de vérification de l'état. Si ce chemin n'est pas défini, l'équilibreur de charge tente d'établir une connexion TCP sur le port 80 pour vérifier l'intégrité.
- Timeout (Délai) – Durée, en secondes, d'attente d'une réponse de la vérification de l'état.
- Interval (Intervalle) – Durée, en secondes, entre les vérifications de l'état d'une instance. L'intervalle doit être supérieur au délai.
- Unhealthy threshold (Seuil de défectuosité), Healthy threshold (Seuil de bonne santé) – Nombre de vérifications de l'état qui doivent échouer ou réussir avant qu'Elastic Load Balancing modifie l'état de santé d'une instance.



> **Note**
>
> La vérification de l'état Elastic Load Balancing n'a pas d'incidence sur le comportement de vérification de l'état du groupe Auto Scaling d'un environnement. Les instances qui échouent à un test de santé d'Elastic Load Balancing ne sont pas automatiquement remplacées par Amazon EC2 Auto Scaling, sauf si vous configurez manuellement Amazon EC2 Auto Scaling pour le faire. Consultez Paramètre de vérification de l'état Auto Scaling pour votre environnement Elastic Beanstalk pour plus de détails.

Pour plus d'informations sur les surveillances de l'état et leur influence sur l'état global de votre environnement, consultez Création de rapports d'intégrité de base.

Configuration d'un Classic Load Balancer à l'aide de l'interface de ligne de commande EB

L'interface de ligne de commande EB vous invite à choisir un type d'équilibreur de charge lorsque vous exécutez la commande eb create.

```
$ eb create
Enter Environment Name
(default is my-app): test-env
Enter DNS CNAME prefix
(default is my-app): test-env-DLW24ED23SF

Select a load balancer type
1) classic
2) application
3) network
(default is 1):
```

Appuyez sur Entrée pour sélectionner l'option `classic`.

Vous pouvez également spécifier un type d'équilibreur de charge à l'aide de l'option `--elb-type`.

```
$ eb create test-env --elb-type classic
```

Espaces de noms pour la configuration d'un Classic Load Balancer

Vous trouverez les paramètres liés aux Classic Load Balancer dans les espaces de noms suivants :

- `aws:elb:healthcheck` – Configurez les seuils, l'intervalle de vérification et le délai d'attente pour les vérifications de l'état de l'équilibreur de charge.
- `aws:elasticbeanstalk:application` – Configurez l'URL de la vérification de l'état.
- `aws:elb:loadbalancer` – Activez l'équilibrage de charge entre zones. Affectez des groupes de sécurité à l'équilibreur de charge et remplacez le groupe de sécurité par défaut créé par Elastic Beanstalk. Cet espace de noms inclut également des options obsolètes pour la configuration des écouteurs standard et sécurisés, qui ont été remplacées par les options de l'espace de noms `aws:elb:listener`.
- `aws:elb:listener` – Configurez l'écouteur par défaut sur le port 80, un écouteur sécurisé sur le port 443 ou des écouteurs supplémentaires pour n'importe quel protocole sur n'importe quel port. Si vous spécifiez l'espace de noms `aws:elb:listener`, les paramètres s'appliquent à l'écouteur par défaut sur le port 80. Si vous spécifiez un port (par exemple, `aws:elb:listener:443`), un écouteur est configuré sur ce port.
- `aws:elb:policies` – Configurez des paramètres supplémentaires pour votre équilibreur de charge. Utilisez les options de cet espace de noms pour configurer les écouteurs sur des ports arbitraires, modifier les paramètres de session persistants supplémentaires et configurer l'équilibreur de charge pour qu'il se connecte aux instances Amazon EC2 en toute sécurité.

L'interface de ligne de commande (CLI) EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Consultez Valeurs recommandées pour plus de détails.

Example.ebextensions/loadbalancer-terminatehttps.config

L'exemple de fichier de configuration suivant crée un écouteur HTTPS sur le port 443, affecte un certificat que l'équilibreur de charge utilise pour mettre la connexion sécurisée hors service, et désactive l'écouteur par défaut sur le port 80. L'équilibreur de charge transmet les demandes déchiffrées aux EC2 instances de votre environnement sur HTTP/80.

```
option_settings:
  aws:elb:listener:443:
    ListenerProtocol: HTTPS
    SSLCertificateId: arn:aws:acm:us-
east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678
    InstancePort: 80
    InstanceProtocol: HTTP
  aws:elb:listener:
```

```
    ListenerEnabled: false
```

Configuration d'un Application Load Balancer

Lorsque vous activez l'équilibrage de charge, votre AWS Elastic Beanstalk environnement est équipé d'un équilibreur de charge Elastic Load Balancing pour répartir le trafic entre les instances de votre environnement. Elastic Load Balancing prend en charge plusieurs types d'équilibreur de charge. Pour en savoir plus, consultez le Guide de l'utilisateur Elastic Load Balancing. Elastic Beanstalk peut créer un équilibreur de charge pour vous, ou vous permettre de spécifier un équilibreur de charge partagé que vous avez créé.

Cette rubrique décrit la configuration d'un équilibreur de charge Application Load Balancer créé par Elastic Beanstalk et dédié à votre environnement. Consultez également the section called “Application Load Balancer partagé”. Pour plus d'informations sur la configuration de tous les types d'équilibreur de charge pris en charge par Elastic Beanstalk, consultez the section called “Équilibreur de charge”.

> **Note**
>
> Vous pouvez choisir le type d'équilibreur de charge que votre environnement utilise uniquement lors de la création de l'environnement. Vous pouvez modifier les paramètres pour gérer le comportement de l'équilibreur de charge de votre environnement d'exécution, mais pas en changer le type. Vous ne pouvez pas non plus passer d'un équilibreur de charge dédié à un équilibreur de charge partagé ou inversement.

Introduction

Un équilibreur de charge Application Load Balancer inspecte le trafic au niveau de la couche protocole réseau de l'application pour identifier le chemin d'accès des demandes afin de les diriger vers différentes destinations pour divers chemins d'accès.

Lorsque votre environnement utilise un Application Load Balancer, Elastic Beanstalk le configure par défaut pour exécuter la même fonction qu'un Classic Load Balancer. L'écouteur par défaut accepte les demandes HTTP sur le port 80 et les distribue aux instances de votre environnement. Vous pouvez ajouter un écouteur sécurisé sur le port 443 avec un certificat permettant de déchiffrer le trafic HTTPS, configurer le fonctionnement de la vérification de l'état et acheminer les journaux d'accès de l'équilibreur de charge vers un compartiment Amazon Simple Storage Service (Amazon S3).

> **Note**
>
> Contrairement à un Classic Load Balancer ou à un Network Load Balancer, un Application Load Balancer ne peut pas avoir de couche de transport (couche 4) TCP ou d'écouteurs. SSL/TLS Il ne prend en charge que les écouteurs HTTP et HTTPS. En outre, il ne peut pas utiliser l'authentification backend pour authentifier les connexions HTTPS entre l'équilibreur de charge et les instances backend.

Dans un environnement Elastic Beanstalk, vous pouvez utiliser un équilibreur de charge Application Load Balancer pour diriger le trafic de certains chemins d'accès vers un autre processus sur vos instances de serveur web. Avec un Classic Load Balancer, l'ensemble du trafic vers un écouteur est acheminé vers un seul processus sur les instances de backend. Un équilibreur de charge Application Load Balancer vous permet de configurer plusieurs règles sur l'écouteur afin d'acheminer les demandes de certains chemins d'accès vers d'autres processus de backend. Vous configurez chaque processus avec le port sur lequel le processus écoute.

Par exemple, vous pouvez exécuter un processus de connexion séparément de votre application principale. Tandis que l'application principale sur les instances de votre environnement accepte la plupart des demandes et écoute sur le port 80, votre processus de connexion écoute sur le port 5000 et accepte les demandes du chemin d'accès `/login`. Toutes les demandes entrantes provenant de clients arrivent sur le port 80. Avec un équilibreur de charge Application Load Balancer, vous pouvez configurer un seul écouteur pour le trafic entrant sur le port 80, avec deux règles qui acheminent le trafic vers deux processus distincts, selon le chemin d'accès de la requête. Vous ajoutez une règle personnalisée qui achemine le trafic vers `/login` pour le processus de connexion écoutant sur le port 5000. La règle par défaut achemine tout le reste du trafic vers le processus d'application principale à l'écoute sur le port 80.

Une règle d'équilibreur de charge Application Load Balancer fait correspondre une demande à un groupe cible. Dans Elastic Beanstalk, un groupe cible est représenté par un processus. Vous pouvez configurer un processus avec un protocole, le un et des paramètres de vérification de l'état. Le processus représente le processus en cours d'exécution sur les instances de votre environnement. Le processus par défaut est un écouteur sur le port 80 du proxy inverse (nginx ou Apache) qui s'exécute devant votre application.

> **Note**
>
> En dehors d'Elastic Beanstalk, un groupe cible est mappé à un groupe d'instances. Un écouteur peut utiliser des règles et des groupes cibles pour acheminer le trafic vers différentes instances en fonction du chemin d'accès. Dans Elastic Beanstalk, toutes les instances de votre environnement sont identiques et la distinction s'applique donc aux processus écoutant sur des ports différents.

Un Classic Load Balancer utilise un chemin de vérification de l'état unique pour l'ensemble de l'environnement. Avec un équilibreur de charge Application Load Balancer, chaque processus a un chemin de vérification de l'état distinct, surveillé par l'équilibreur de charge et la surveillance améliorée de l'état Elastic Beanstalk.

Pour utiliser un équilibreur de charge Application Load Balancer, votre environnement doit se trouver dans un VPC par défaut ou personnalisé et doit avoir un rôle de service avec un ensemble standard d'autorisations. Si vous avez un rôle de service plus ancien, il sera peut-être nécessaire de mettre à jour ses autorisations afin d'inclure `elasticloadbalancing:DescribeTargetHealth` et `elasticloadbalancing:DescribeLoadBalancers`. Pour plus d'informations sur un Application Load Balancer, consultez Qu'est-ce qu'un Application Load Balancer ?.

> **Note**
>
> La vérification de l'état de l'équilibreur de charge Application Load Balancer n'utilise pas le chemin d'accès à la vérification de l'état Elastic Beanstalk. Elle utilise le chemin spécifique configuré pour chaque processus séparément.

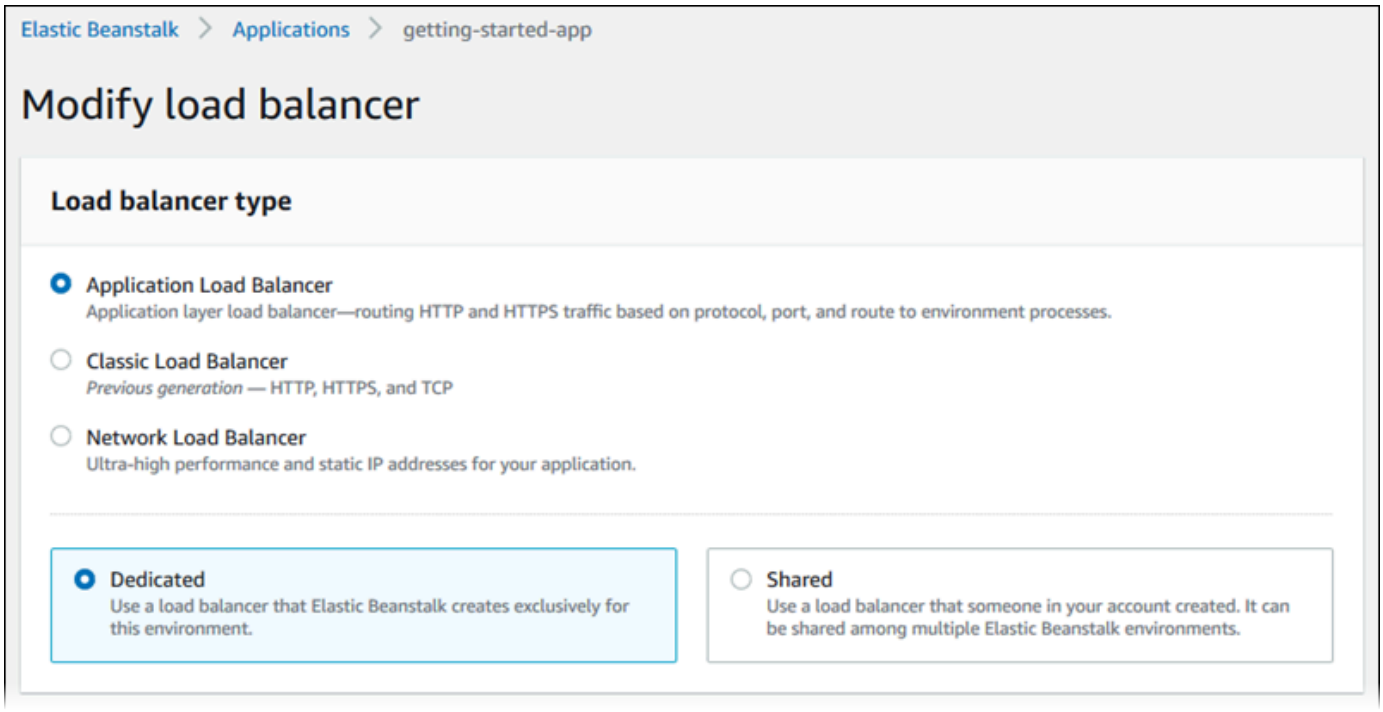
Configuration d'un équilibreur de charge Application Load Balancer à l'aide de la console Elastic Beanstalk

Vous pouvez utiliser la console Elastic Beanstalk pour configurer les écouteurs, les processus et les règles d'un équilibreur de charge Application Load Balancer lors de la création de l'environnement ou plus tard alors que votre environnement est en cours d'exécution.

Pour configurer un Application Load Balancer dans la console Elastic Beanstalk lors de la création de l'environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements).
3. Choisissez Create a new environment (Créer un nouvel environnement) pour commencer à créer votre environnement.
4. Sur la page principale de l'assistant, avant de choisir Créer un environnement, choisissez Configurer plus d'options.
5. Choisissez le préréglage de configuration Haute disponibilité.

   Sinon, dans la catégorie de configuration Capacité, configurez un type d'environnement avec un Équilibrage de charge. Pour en savoir plus, consultez Capacity.
6. Dans la catégorie de configuration Load balancer (Équilibreur de charge), choisissez Edit (Modifier).
7. Sélectionnez les options Application Load Balancer (Équilibreur de charge d'application) et Dedicated (Dédié) si elles ne sont pas déjà sélectionnées.



8. Effectuez toutes les modifications de configuration de l'équilibreur de charge Application Load Balancer exigées par votre environnement.

9. Choisissez Enregistrer, puis effectuez toutes les autres modifications de configuration exigées par votre environnement.
10. Choisissez Create environment.

Pour configurer un équilibreur de charge Application Load Balancer d'un environnement en cours d'exécution dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Load balancer (Équilibreur de charge), choisissez Edit (Modifier).

> **Note**
>
> Si la catégorie de configuration Load balancer (Équilibreur de charge) ne dispose pas du bouton Edit (Modifier), cela signifie que votre environnement ne dispose pas d'un équilibreur de charge. Pour apprendre à en configurer un, consultez Changement de type d'environnement.

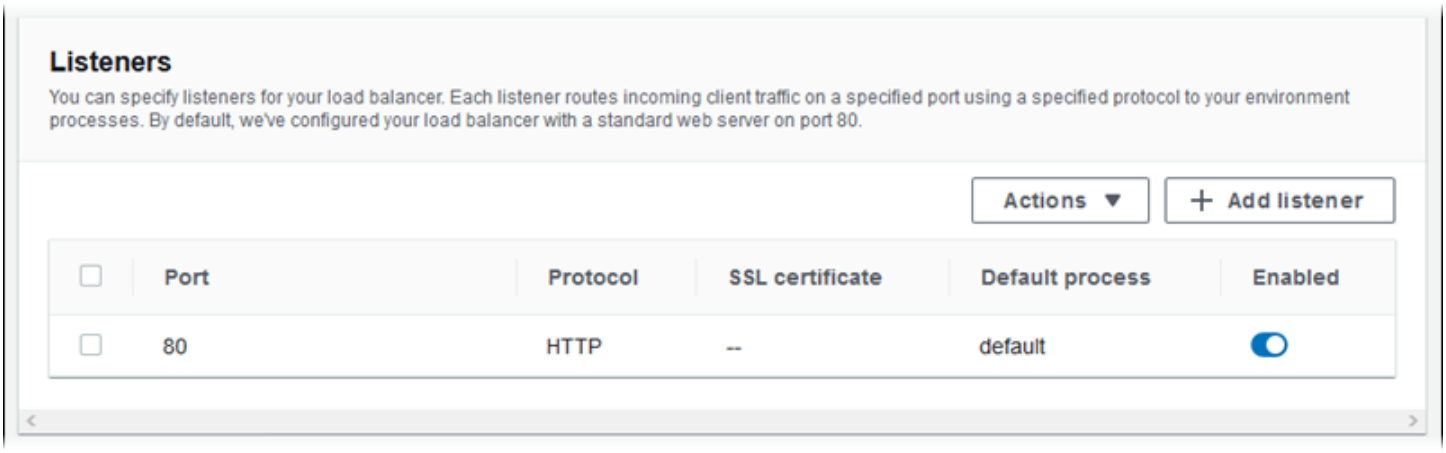
5. Effectuez les modifications de configuration de l'équilibreur de charge Application Load Balancer exigées par votre environnement.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Paramètres de l’Application Load Balancer

- Écouteurs
- Processus
- Rules
- Capture des journaux d'accès

Écouteurs

Utilisez cette liste pour spécifier plusieurs écouteurs pour votre équilibreur de charge. Chaque écouteur achemine le trafic client entrant sur un port spécifié à l'aide d'un protocole spécifié vers un ou plusieurs processus sur vos instances. Initialement, la liste indique l'écouteur par défaut, qui achemine le trafic HTTP entrant sur le port 80 pour un processus nommé default.



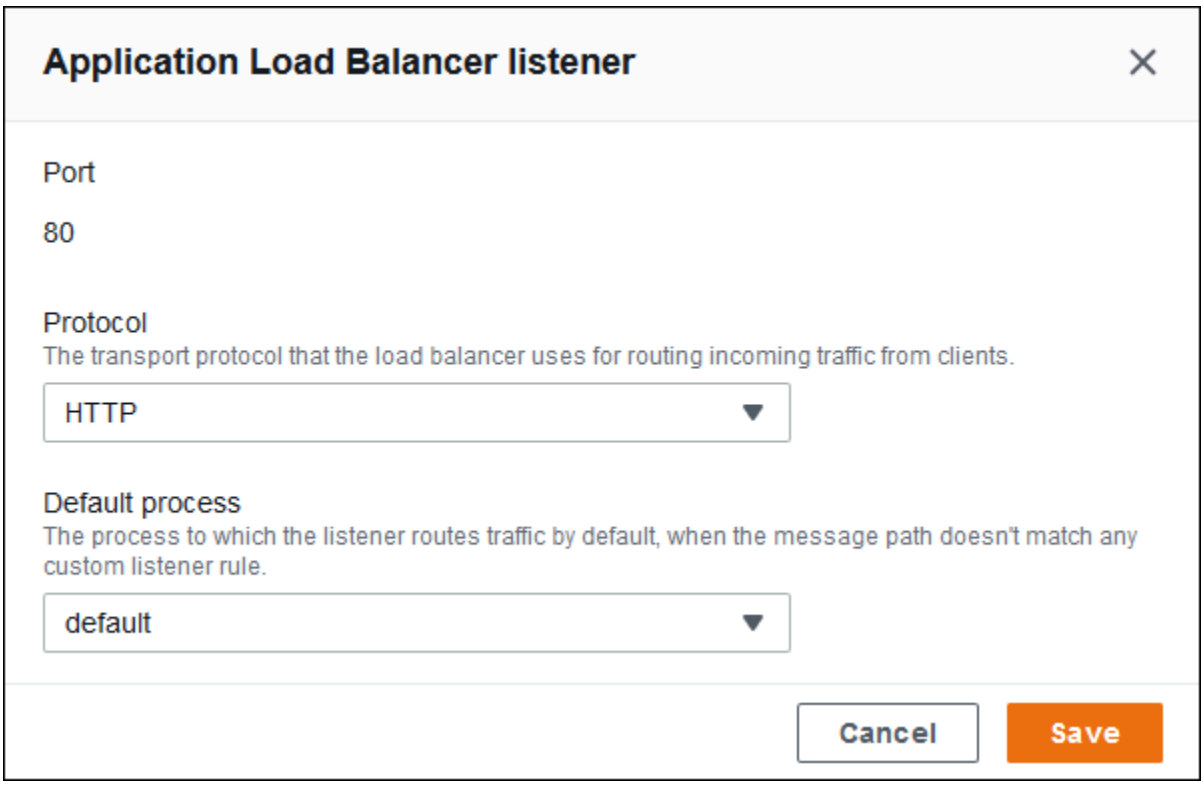
Pour configurer un écouteur existant

1. Cochez la case en regard de son entrée de table, puis choisissez Actions, Modifier.
2. Utilisez la boîte de dialogue Écouteur de l'Application Load Balancer pour modifier les paramètres, puis choisissez Enregistrer.

Pour ajouter un écouteur

1. Choisissez Add listener (Ajouter un écouteur).
2. Dans la boîte de dialogue Écouteur de l'équilibreur de charge d'application, configurez les paramètres désirés, puis choisissez Ajouter.

Utilisez les paramètres de la boîte de dialogue Écouteur de l'Application Load Balancer pour choisir le port et le protocole sur lesquels l'écouteur écoute le trafic, ainsi que le processus vers lequel acheminer le trafic. Si vous choisissez le protocole HTTPS, configurez les paramètres SSL.



Pour pouvoir configurer un écouteur HTTPS, assurez-vous que vous disposez d'un certificat SSL valide. Effectuez l'une des actions suivantes :

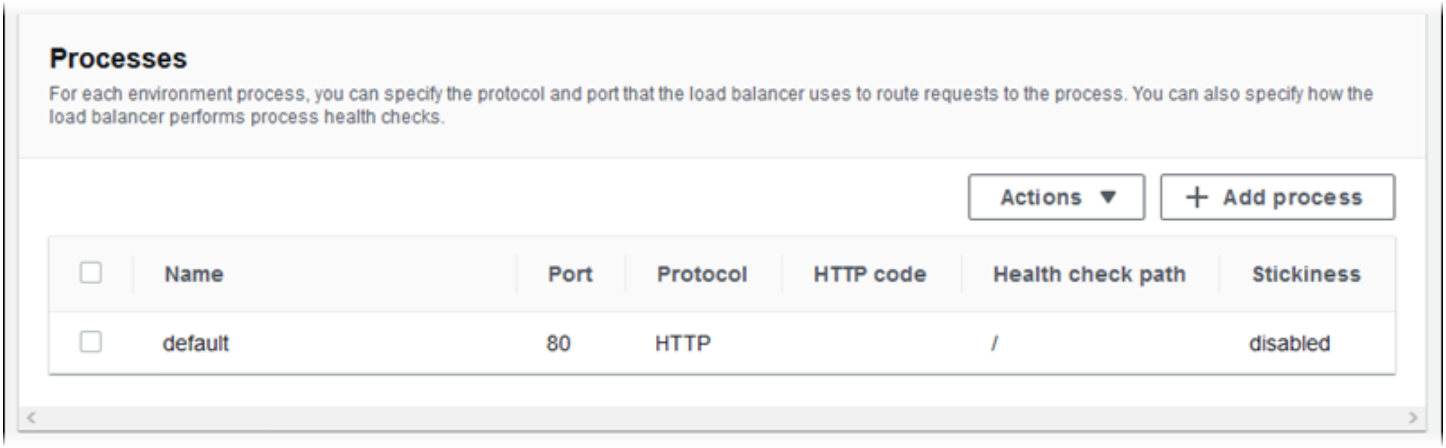
- Si AWS Certificate Manager (ACM) est disponible dans votre AWS région, créez ou importez un certificat à l'aide d'ACM. Pour plus d'informations sur une demande de certificat ACM, consultez Request a Certificate (Demande de certificat) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur AWS Certificate Manager). Pour plus d'informations sur l'importation de certificats tiers dans ACM, consultez Importing Certificates (Importation de certificats) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur AWS Certificate Manager).
- Si ACM n'est pas disponible dans votre AWS région, téléchargez votre certificat et votre clé existants dans IAM. Pour plus d'informations sur la création et le chargement des certificats dans IAM, consultez Utilisation des certificats de serveur dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur la configuration HTTPS et l'utilisation des certificats dans Elastic Beanstalk, consultez Configuration de HTTPS pour votre environnement Elastic Beanstalk.

Processus

Utilisez cette liste pour spécifier les processus pour votre équilibreur de charge. Un processus est une cible pour le routage du trafic par les écouteurs. Chaque écouteur achemine le trafic client

entrant sur un port spécifié à l'aide d'un protocole spécifié vers un ou plusieurs processus sur vos instances. Initialement, la liste indique le processus par défaut, qui écoute le trafic HTTP entrant sur le port 80.



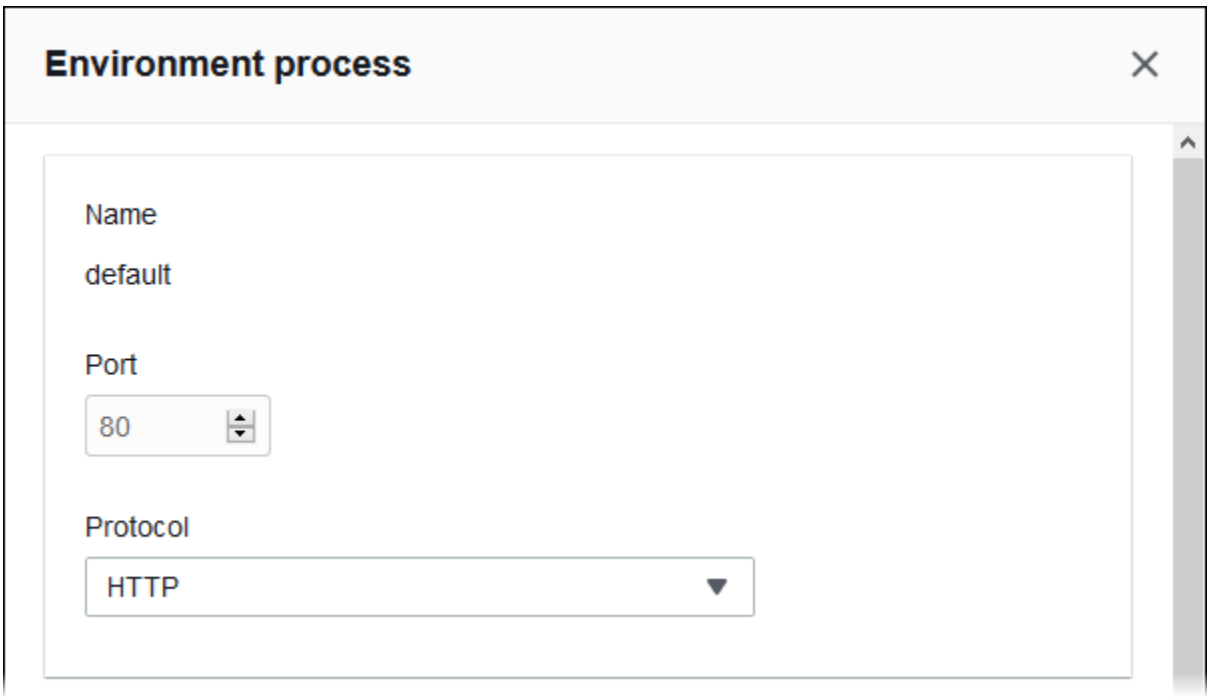
Vous pouvez modifier les paramètres d'un processus existant ou ajouter un nouveau processus. Pour commencer la modification d'un processus de la liste ou l'ajout d'un processus, utilisez la procédure définie pour la liste d'écouteurs. La boîte de dialogue Processus d'environnement s'ouvre.

Paramètres de la boîte de dialogue de processus d'environnement de l'Application Load Balancer

- Définition
- Surveillance de l'état
- Séances

Définition

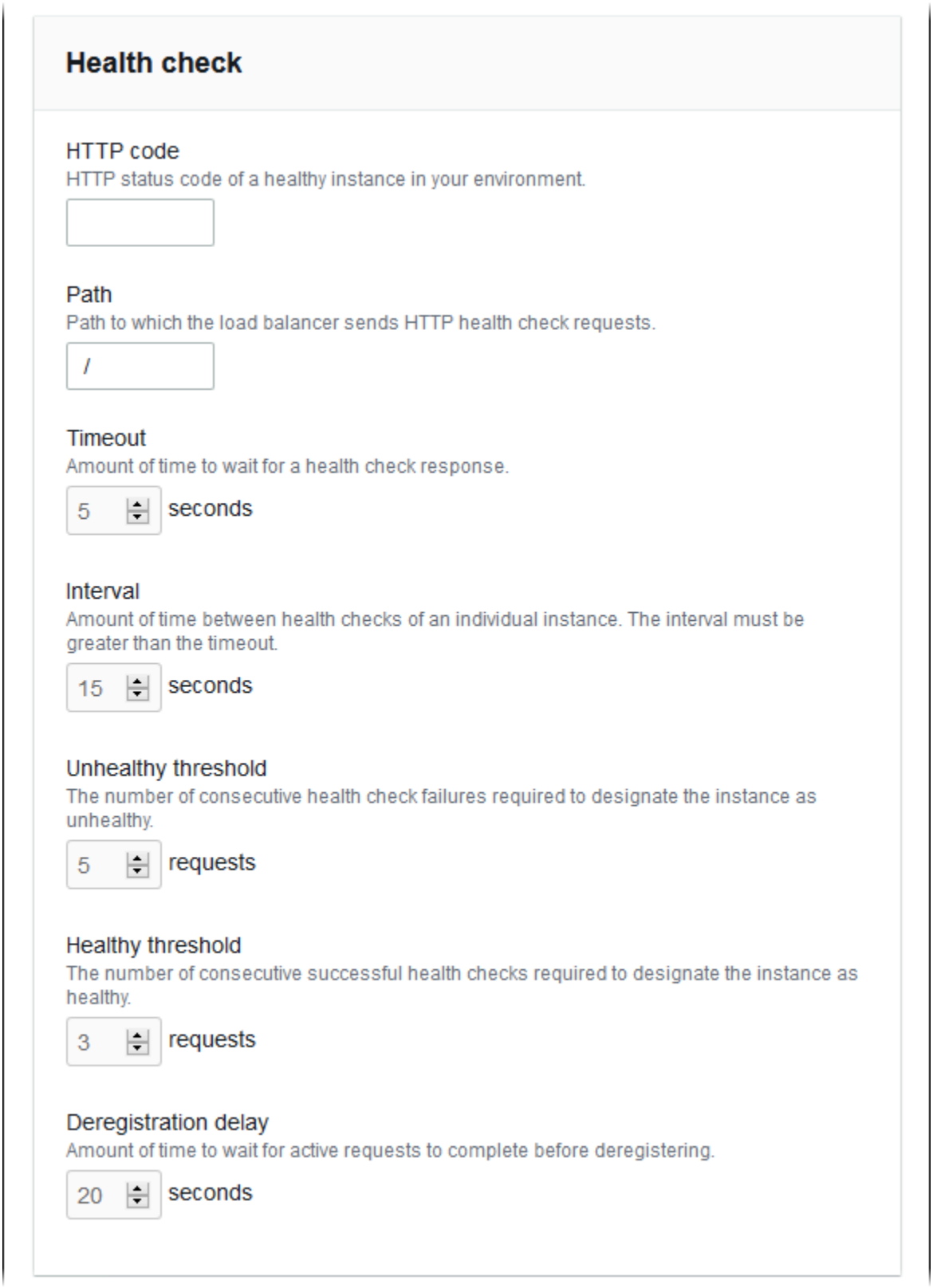
Utilisez ces paramètres pour définir le processus : son nom et le port et le protocole sur lesquels il écoute les requêtes.



Surveillance de l'état

Utilisez les paramètres suivants pour configurer les processus des vérifications de l'état :

- HTTP code (Code HTTP) – Code d'état HTTP désignant un processus sain.
- Path (Chemin) – Chemin d'accès de la demande de vérification de l'état du processus.
- Timeout (Délai) – Durée, en secondes, d'attente d'une réponse de la vérification de l'état.
- Interval (Intervalle) – Durée, en secondes, entre les vérifications de l'état d'une instance. L'intervalle doit être supérieur au délai.
- Unhealthy threshold (Seuil de défectuosité), Healthy threshold (Seuil de bonne santé) – Nombre de vérifications de l'état qui doivent échouer ou réussir avant qu'Elastic Load Balancing modifie l'état de santé d'une instance.
- Deregistration delay (Retard d'annulation d'enregistrement) – Délai, en secondes, d'attente de la fin des requêtes actives avant l'annulation de l'enregistrement d'une instance.

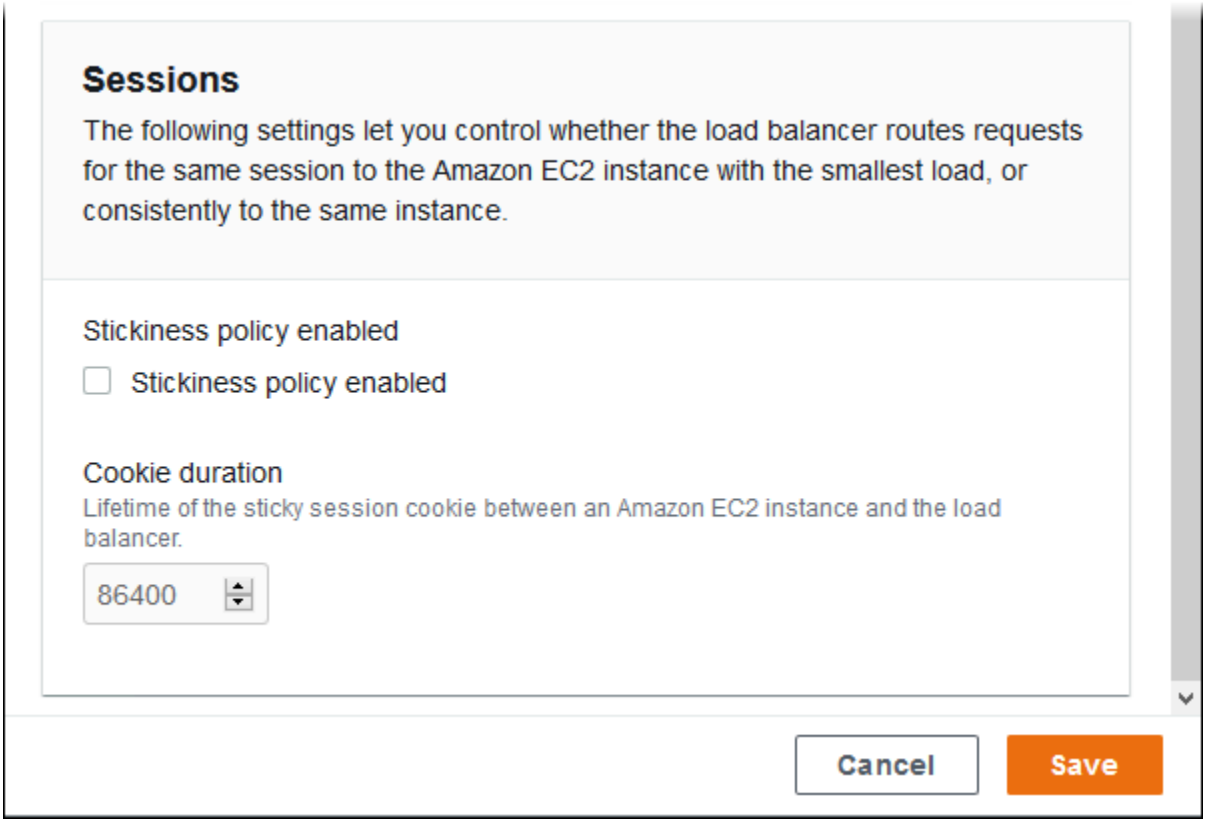

Health check
HTTP code
HTTP status code of a healthy instance in your environment.
Path
Path to which the load balancer sends HTTP health check requests.
/
Timeout
Amount of time to wait for a health check response.
5
seconds
Interval
Amount of time between health checks of an individual instance. The interval must be greater than the timeout.
15
seconds
Unhealthy threshold
The number of consecutive health check failures required to designate the instance as unhealthy.
5
requests
Healthy threshold
The number of consecutive successful health checks required to designate the instance as healthy.
3
requests
Deregistration delay
Amount of time to wait for active requests to complete before deregistering.
20
seconds

> **Note**
>
> La vérification de l'état Elastic Load Balancing n'a pas d'incidence sur le comportement de vérification de l'état du groupe Auto Scaling d'un environnement. Les instances qui échouent à un test de santé d'Elastic Load Balancing ne sont pas automatiquement remplacées par Amazon EC2 Auto Scaling, sauf si vous configurez manuellement Amazon EC2 Auto Scaling pour le faire. Consultez Paramètre de vérification de l'état Auto Scaling pour votre environnement Elastic Beanstalk pour plus de détails.

Pour plus d'informations sur les vérifications de l'état et leur influence sur l'état global de votre environnement, consultez Création de rapports d'intégrité de base.

Séances

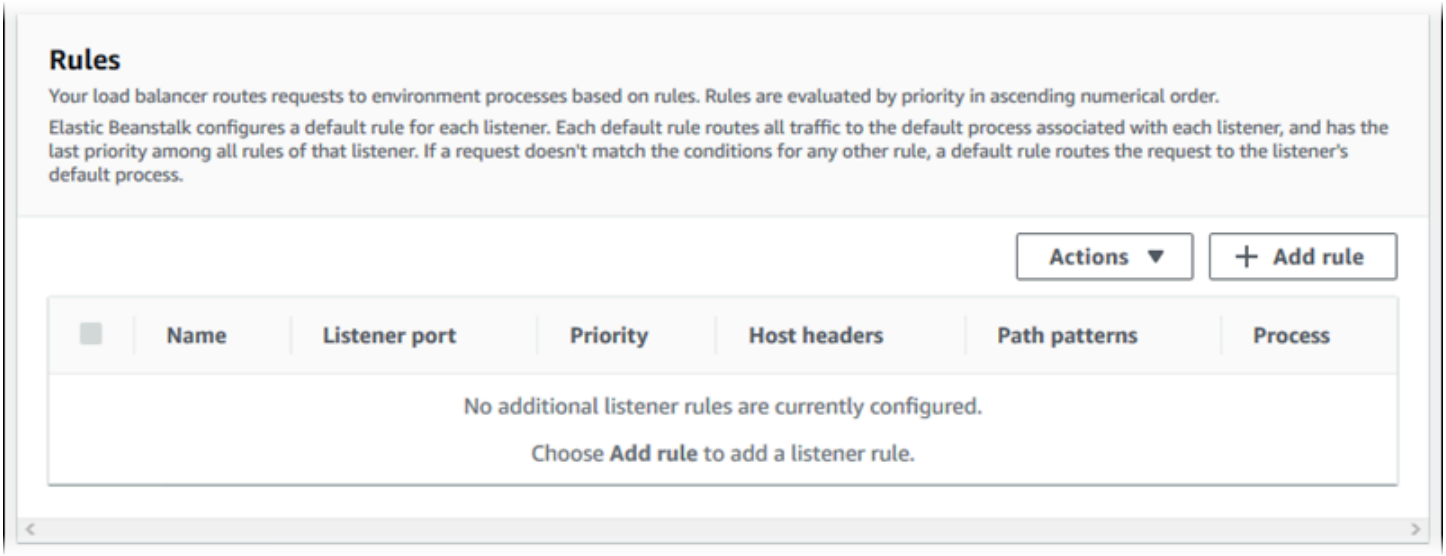
Activez ou désactivez la case Politique de permanence activée pour activer ou désactiver les sessions permanentes. Utilisez l'option Durée du cookie pour configurer la durée d'une session permanente, pouvant aller jusqu'à **`604800`** secondes.



Rules

Utilisez cette liste pour spécifier des règles d'écouteur personnalisées pour votre équilibreur de charge. Une règle mappe les requêtes que l'écouteur reçoit sur un modèle de chemin spécifique à un processus cible. Chaque écouteur peut avoir plusieurs règles acheminant les requêtes sur différents chemins à différents processus sur vos instances.

Les règles ont des priorités numériques qui déterminent la priorité selon laquelle elles sont appliquées aux demandes entrantes. Pour chaque nouvel écouteur que vous ajoutez, Elastic Beanstalk ajoute une règle par défaut qui achemine l'ensemble du trafic de l'écouteur au processus par défaut. La priorité de la règle par défaut est la plus basse ; elle n'est appliquée que si aucune autre règle pour le même écouteur ne correspond à la requête entrante. Initialement, si vous n'avez pas ajouté de règles personnalisées, la liste est vide. Les règles par défaut de tous les écouteurs ne sont pas affichées.

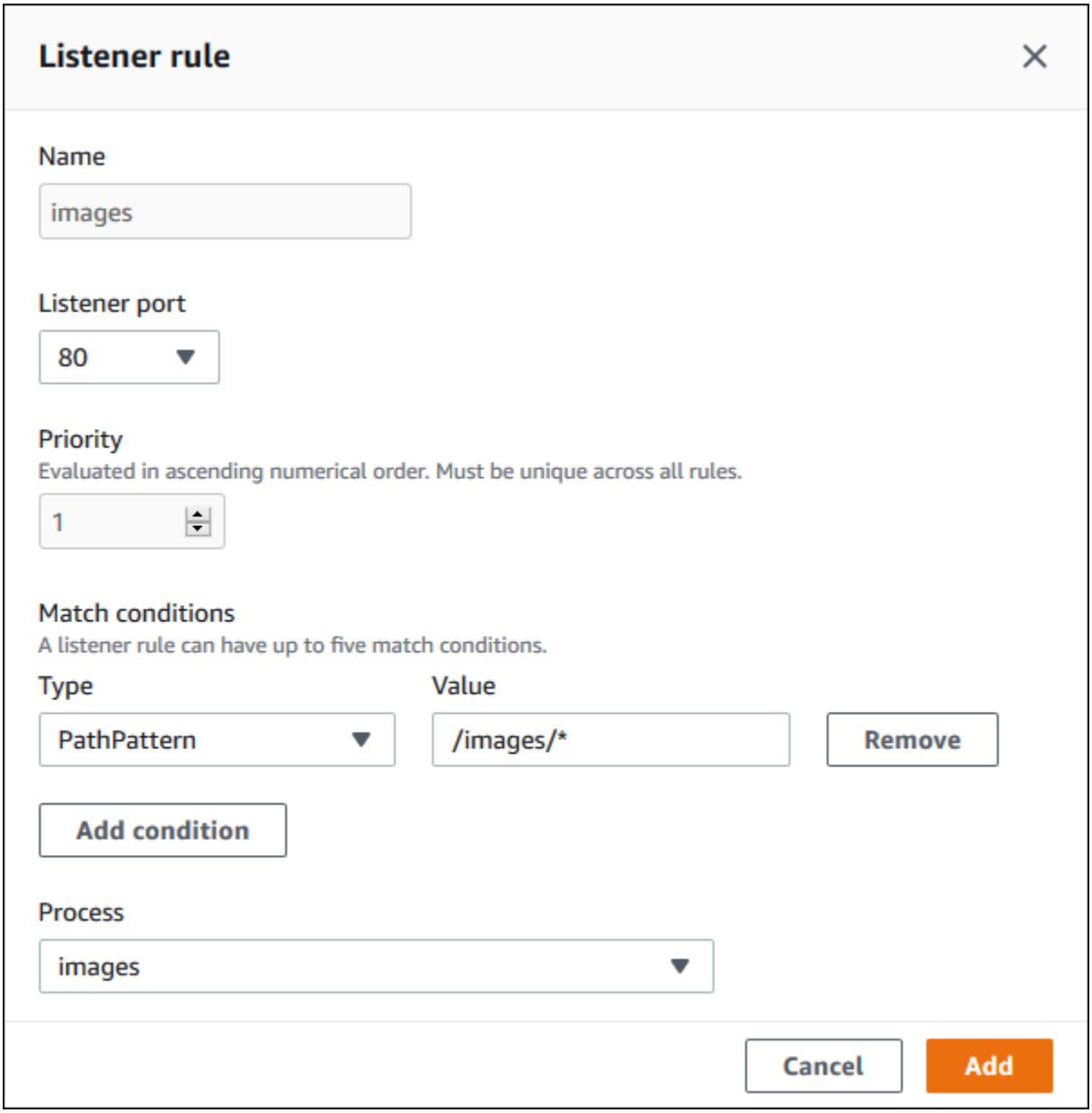


Vous pouvez modifier les paramètres d'une règle existante ou ajouter une nouvelle règle. Pour commencer la modification d'une règle de la liste ou l'ajout d'une règle, utilisez la procédure définie pour la liste d'écouteurs. La boîte de dialogue Règle d'écouteur s'ouvre, avec les paramètres suivants :

- Name (Nom) – Nom de la règle.
- Listener port (Port d'écoute) – Port d'écoute auquel la règle s'applique.
- Priority (Priorité) – Priorité de la règle. Une valeur de priorité plus faible a une priorité plus élevée. Les priorités des règles d'un écouteur doivent être uniques.

- Match conditions (Conditions de correspondance) – Liste des conditions d'URL de requête auxquelles la règle s'applique. Il existe deux types de conditions : HostHeader(la partie domaine de l'URL) et PathPattern(la partie chemin de l'URL). Vous pouvez ajouter jusqu'à cinq conditions. Chaque valeur de condition comporte jusqu'à 128 caractères et peut inclure des caractères génériques.
- Process (Processus) – Processus auquel l'équilibreur de charge achemine les demandes qui correspondent à la règle.

Lorsque vous modifiez une règle existante, vous ne pouvez pas changer son Nom ni son Port d'écoute.

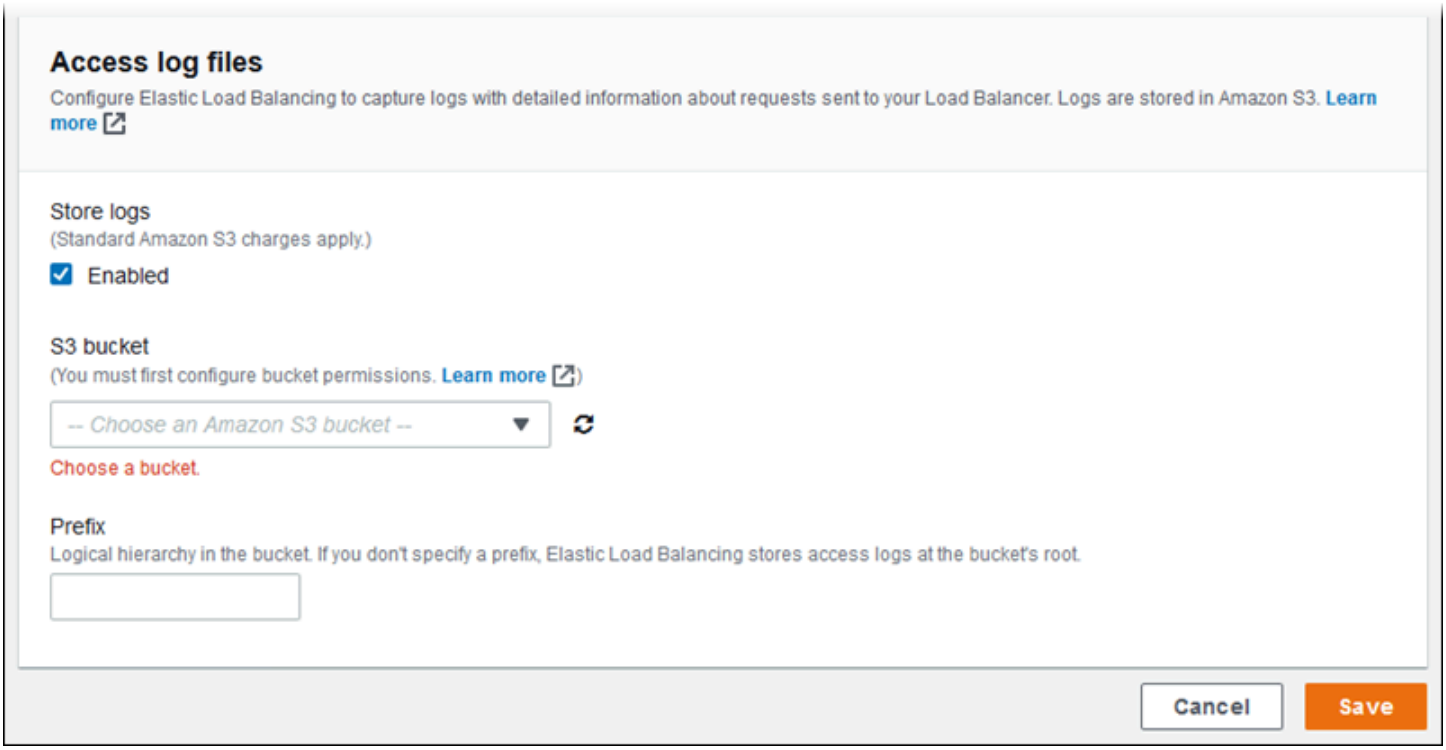


Capture des journaux d'accès

Utilisez ces paramètres pour configurer Elastic Load Balancing afin qu'il capture les journaux avec des informations détaillées sur les demandes envoyées à votre équilibreur de charge Application Load Balancer. La capture des journaux d'accès est désactivée par défaut. Lorsque l'option Store logs (Stocker des journaux) est activée, Elastic Load Balancing stocke les journaux dans le compartiment S3 que vous configurez. Le paramètre Prefix (Préfixe) spécifie un dossier de niveau supérieur dans le compartiment pour les journaux. Elastic Load Balancing place les journaux dans un dossier nommé `AWSLogs` sous votre préfixe. Si vous ne spécifiez pas de préfixe, Elastic Load Balancing place son dossier au niveau racine du compartiment.

> **Note**
>
> Si le compartiment Amazon S3 que vous configurez pour la capture du journal d'accès n'est pas le compartiment créé par Elastic Beanstalk pour votre compte, veillez à ajouter une politique utilisateur avec les autorisations appropriées Gestion des identités et des accès AWS pour vos utilisateurs (IAM). Les politiques d'utilisateur gérées fournies par Elastic Beanstalk ne couvrent que les autorisations pour les ressources gérées par Elastic Beanstalk.

Pour plus d'informations sur les journaux d'accès, y compris les autorisations et d'autres exigences, consultez la section Journaux d'accès pour votre Equilibreur de charge d'application.



Exemple : Équilibreur de charge Application Load Balancer avec un écouteur sécurisé et deux processus

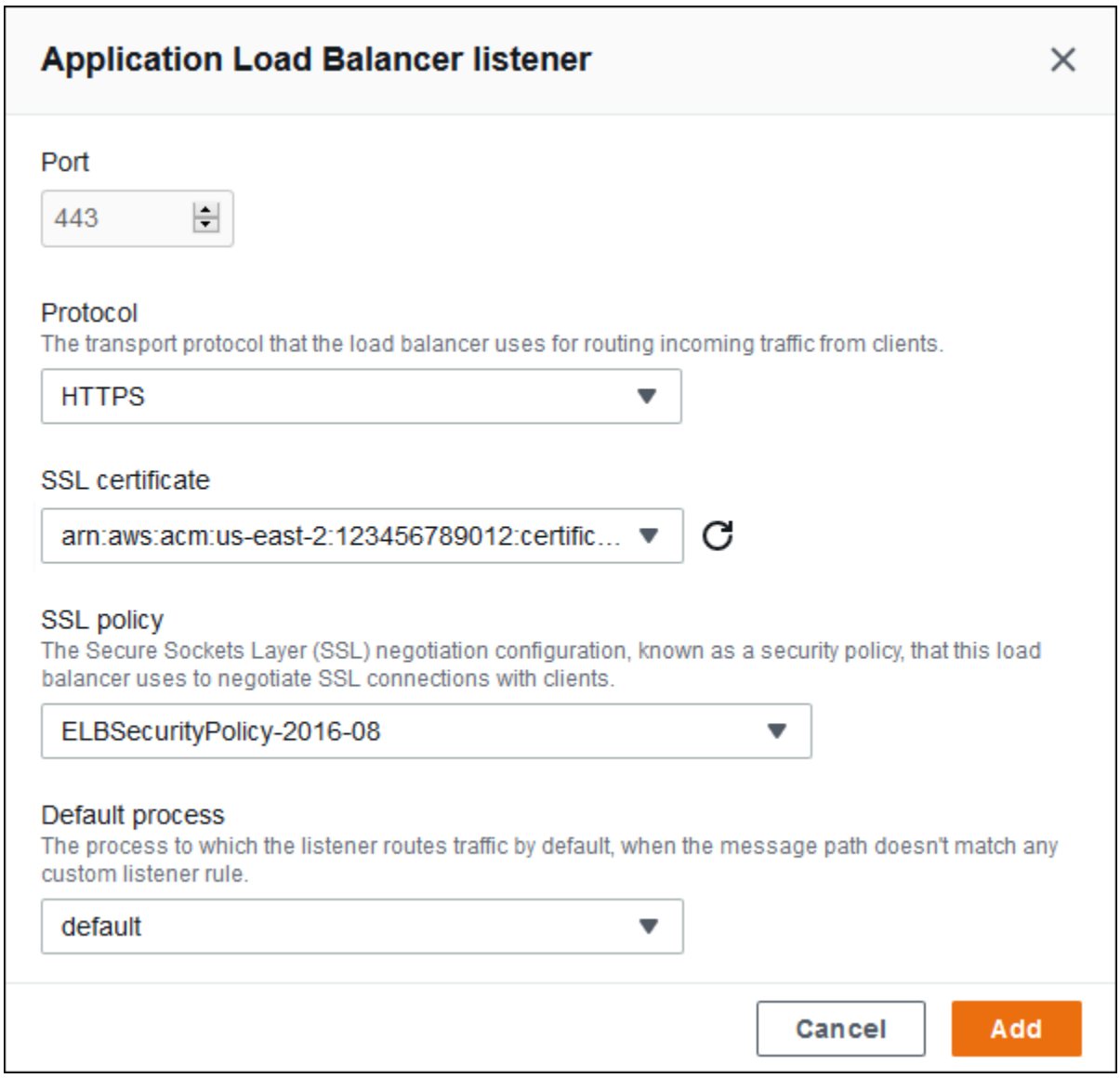
Dans cet exemple, votre application nécessite le chiffrement end-to-end du trafic et un processus distinct pour le traitement des demandes administratives.

Pour configurer l'équilibreur de charge Application Load Balancer de votre environnement afin qu'il réponde à ces exigences, vous devez supprimer l'écouteur par défaut, ajouter un écouteur HTTPS, indiquer le processus par défaut qui écoute le port 443 sur HTTPS et ajouter un processus et une règle d'écouteur pour le trafic d'administration sur un autre chemin.

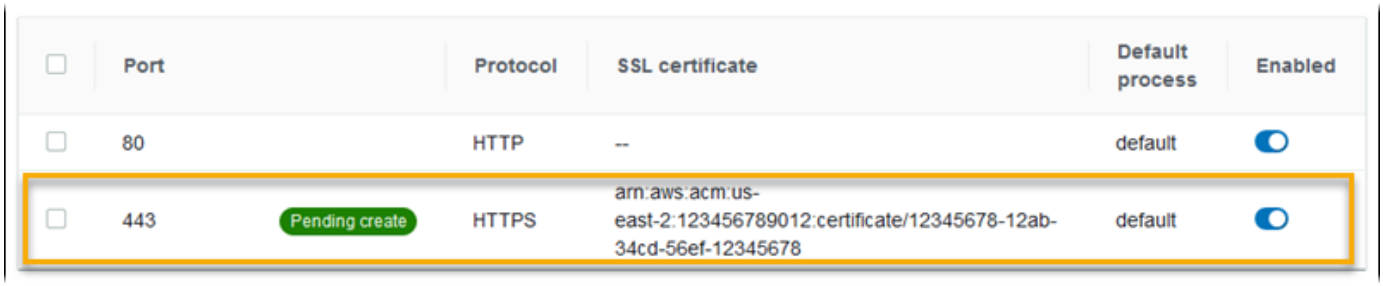
Pour configurer l'équilibreur de charge de cet exemple

1. Ajoutez un écouteur sécurisé. Pour Port, entrez **`443`**. Pour Protocole, sélectionnez **`HTTPS`**. Pour Certificat SSL, choisissez l'ARN de votre certificat SSL. Par exemple, **`arn:aws:iam::123456789012:server-certificate/abc/certs/build`** ou **`arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678`**.

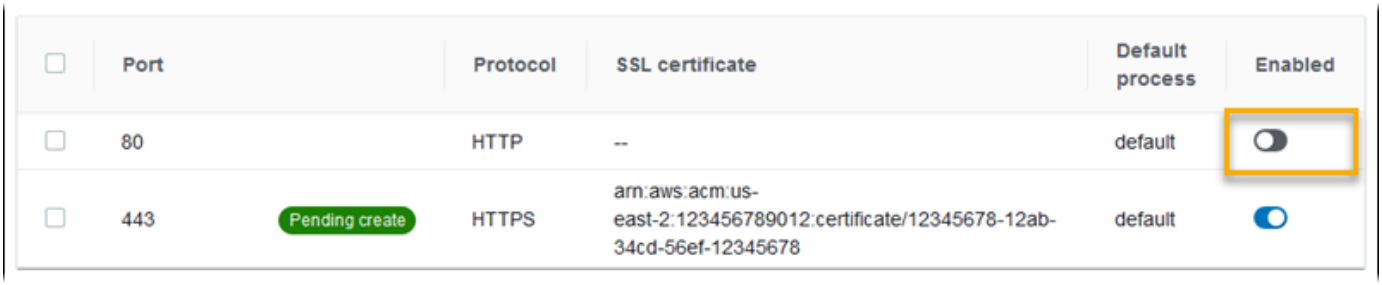
   Pour Processus par défaut, gardez **`default`** sélectionné.



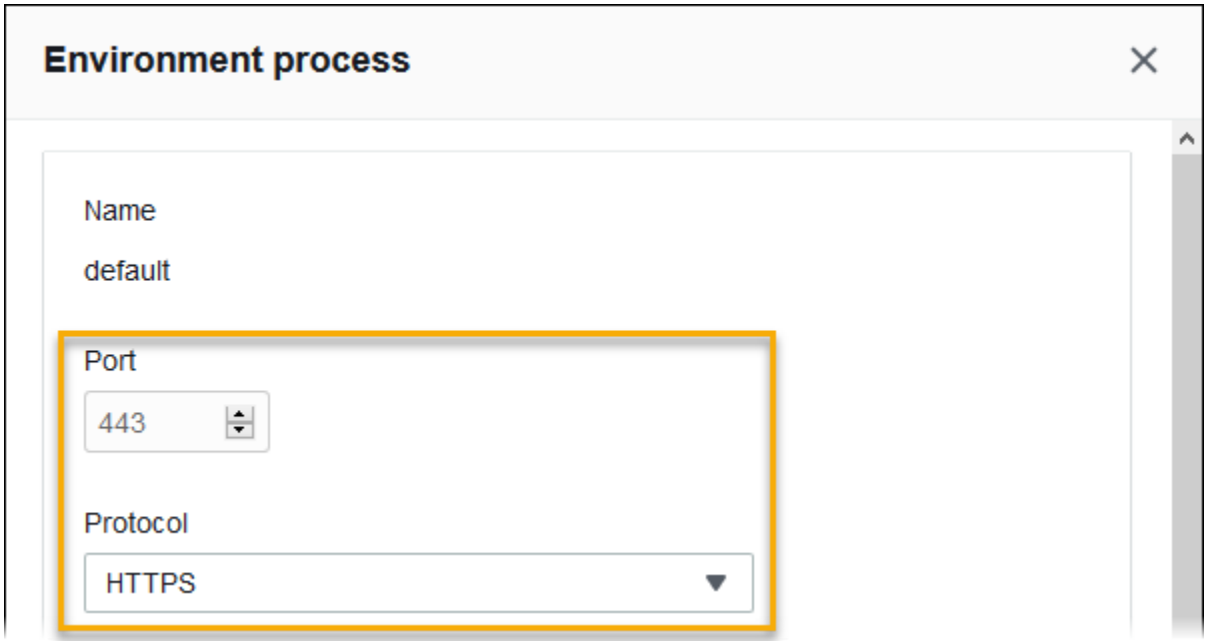
Vous voyez désormais votre écouteur supplémentaires dans la liste.



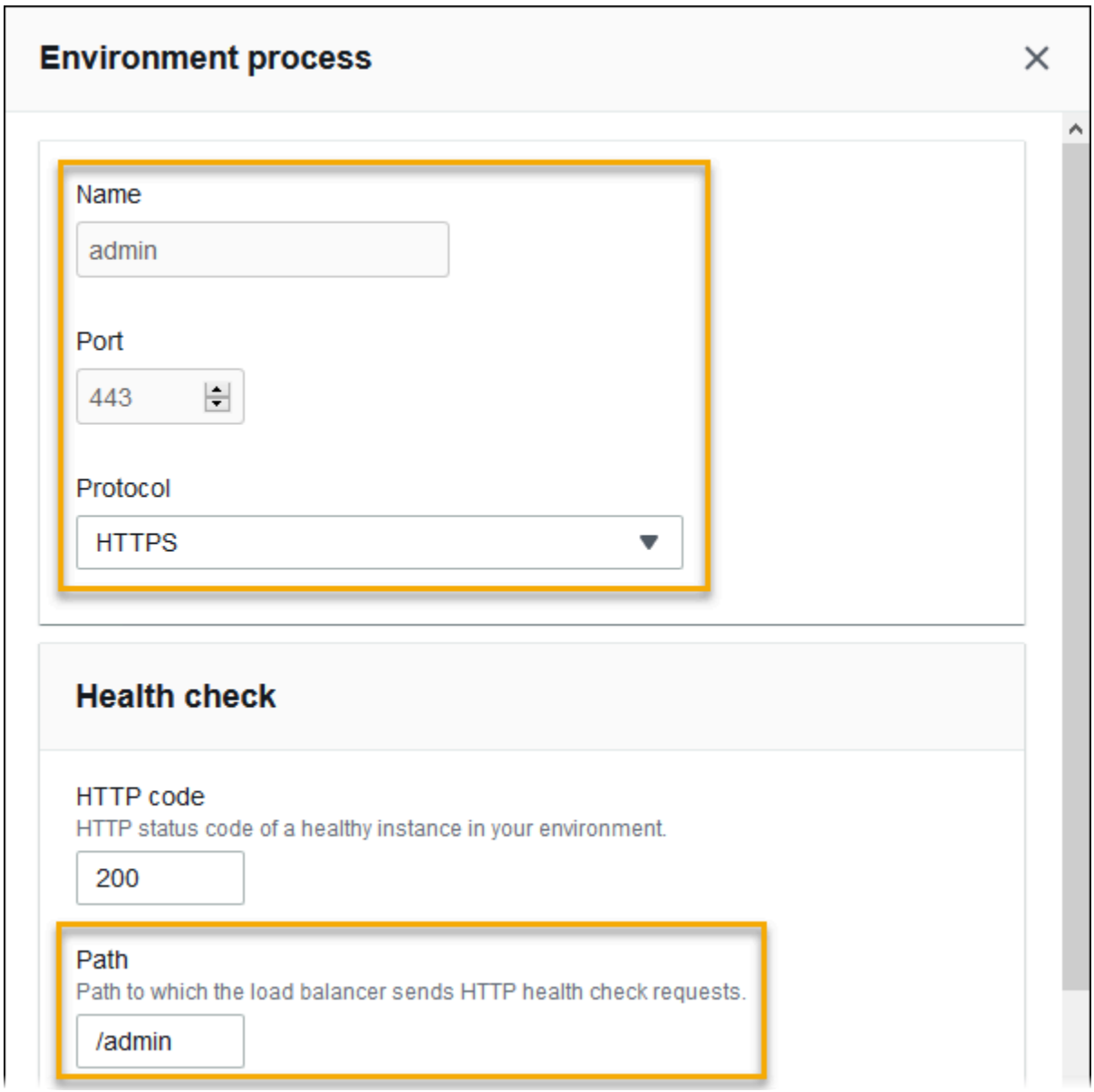
2. Désactivez l'écouteur HTTP du port 80 par défaut. Pour l'écouteur par défaut, désactivez l'option Activé.



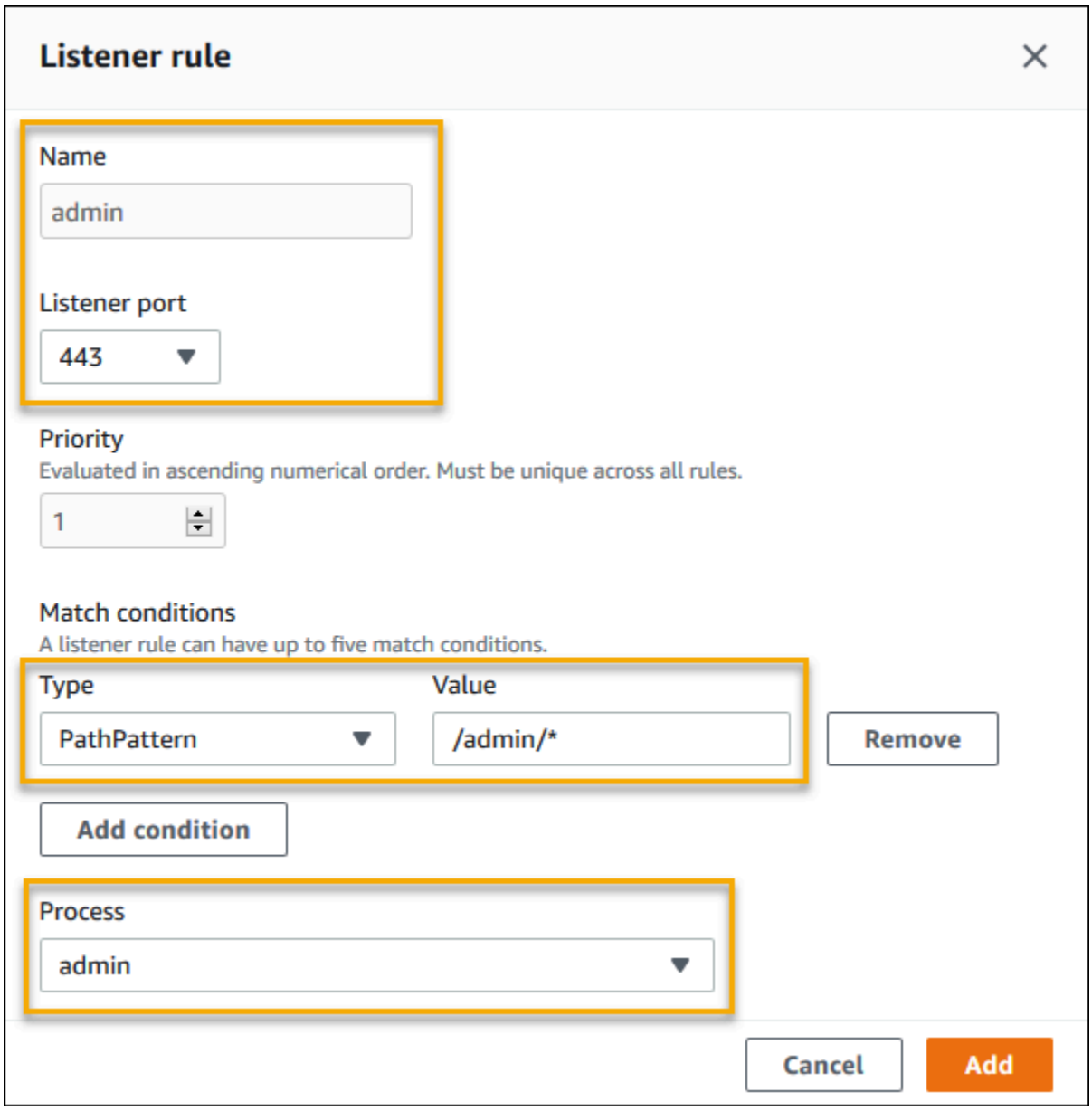
3. Configurez le processus par défaut sur HTTPS. Sélectionnez le processus par défaut, puis, pour Actions, choisissez Edit (Modifier). Pour Port, entrez **`443`**. Pour Protocole, sélectionnez **`HTTPS`**.



4. Ajoutez un processus d'administration. Pour Name (Nom), tapez **`admin`**. Pour Port, entrez **`443`**. Pour Protocole, sélectionnez **`HTTPS`**. Sous Vérification de l'état, pour Chemin, tapez **`/admin`**.



5. Ajoutez une règle pour le trafic d'administration. Pour Name (Nom), tapez **`admin`**. Pour Port d'écoute, tapez **`443`**. Pour les conditions de correspondance, ajoutez un PathPatternavec la valeur**`/admin/*`**. Pour Processus, sélectionnez **`admin`**.



Configuration d'un équilibreur de charge Application Load Balancer à l'aide de l'interface de ligne de commande EB

L'interface de ligne de commande EB vous invite à choisir un type d'équilibreur de charge lorsque vous exécutez la commande eb create.

```
$ eb create
Enter Environment Name
(default is my-app): test-env
Enter DNS CNAME prefix
```

```
(default is my-app): test-env-DLW24ED23SF

Select a load balancer type
1) classic
2) application
3) network
(default is 2):
```

Vous pouvez également spécifier un type d'équilibreur de charge à l'aide de l'option `--elb-type`.

```
$ eb create test-env --elb-type application
```

Espaces de noms Application Load Balancer

Vous trouverez les paramètres liés aux équilibreurs Application Load Balancers dans les espaces de noms suivants :

- `aws:elasticbeanstalk:environment` – Choisissez le type d'équilibreur de charge pour votre environnement. La valeur d'un équilibreur de charge Application Load Balancer est `application`.

  Vous ne pouvez pas définir cette option dans les fichiers de configuration (.Ebextensions).
- `aws:elbv2:loadbalancer` – Configurez les journaux d'accès et autres paramètres s'appliquant à l'équilibreur de charge Application Load Balancer dans son ensemble.
- `aws:elbv2:listener` – Configurez les écouteurs sur l'équilibreur de charge Application Load Balancer. Ces paramètres sont mappés aux paramètres de `aws:elb:listener` pour les Classic Load Balancers.
- `aws:elbv2:listenerrule` – Configurez les règles qui acheminent le trafic vers différents processus, selon le chemin de la demande. Les règles sont spécifiques aux équilibreurs de charge Application Load Balancer.
- `aws:elasticbeanstalk:environment:process` – Configurez les vérifications de l'état et spécifiez le port et le protocole des processus qui s'exécutent sur les instances de votre environnement. Les paramètres de port et de protocole sont mappés sur ceux de l'instance dans `aws:elb:listener` pour un écouteur sur un Classic Load Balancer. Les paramètres de vérification de l'état sont mappés sur les paramètres des espaces de noms `aws:elb:healthcheck` et `aws:elasticbeanstalk:application`.

Example extensions .eb/ .config alb-access-logs

Le fichier de configuration suivant permet le chargement de journaux d'accès pour un environnement comportant un équilibreur de charge Application Load Balancer.

```
option_settings:
  aws:elbv2:loadbalancer:
    AccessLogsS3Bucket: amzn-s3-demo-bucket
    AccessLogsS3Enabled: 'true'
    AccessLogsS3Prefix: beanstalk-alb
```

Example extensions .eb/ .config alb-default-process

Le fichier de configuration suivant modifie les paramètres de vérification de l'état et de permanence sur le processus par défaut.

```
option_settings:
  aws:elasticbeanstalk:environment:process:default:
    DeregistrationDelay: '20'
    HealthCheckInterval: '15'
    HealthCheckPath: /
    HealthCheckTimeout: '5'
    HealthyThresholdCount: '3'
    UnhealthyThresholdCount: '5'
    Port: '80'
    Protocol: HTTP
    StickinessEnabled: 'true'
    StickinessLBCookieDuration: '43200'
```

Example extensions .eb/ .config alb-secure-listener

Le fichier de configuration suivant ajoute un écouteur sécurisé et un processus correspondant sur le port 443.

```
option_settings:
  aws:elbv2:listener:443:
    DefaultProcess: https
    ListenerEnabled: 'true'
    Protocol: HTTPS
    SSLCertificateArns: arn:aws:acm:us-
east-2:123456789012:certificate/21324896-0fa4-412b-bf6f-f362d6eb6dd7
  aws:elasticbeanstalk:environment:process:https:
    Port: '443'
```

```
    Protocol: HTTPS
```

Example extensions .eb/ .config alb-admin-rule

Le fichier de configuration suivant ajoute un écouteur sécurisé avec une règle qui achemine le trafic dont le chemin de la demande est `/admin` vers un processus nommé `admin` qui écoute sur le port 4443.

```
option_settings:
  aws:elbv2:listener:443:
    DefaultProcess: https
    ListenerEnabled: 'true'
    Protocol: HTTPS
    Rules: admin
    SSLCertificateArns: arn:aws:acm:us-
east-2:123456789012:certificate/21324896-0fa4-412b-bf6f-f362d6eb6dd7
  aws:elasticbeanstalk:environment:process:https:
    Port: '443'
    Protocol: HTTPS
  aws:elasticbeanstalk:environment:process:admin:
    HealthCheckPath: /admin
    Port: '4443'
    Protocol: HTTPS
  aws:elbv2:listenerrule:admin:
    PathPatterns: /admin/*
    Priority: 1
    Process: admin
```

Configuration d'un Application Load Balancer partagé

Lorsque vous activez l'équilibrage de charge, votre AWS Elastic Beanstalk environnement est équipé d'un équilibreur de charge Elastic Load Balancing pour répartir le trafic entre les instances de votre environnement. Elastic Load Balancing prend en charge plusieurs types d'équilibreur de charge. Pour en savoir plus, consultez le Guide de l'utilisateur Elastic Load Balancing. Elastic Beanstalk peut créer un équilibreur de charge pour vous, ou vous permettre de spécifier un équilibreur de charge partagé que vous avez créé.

Cette rubrique décrit la configuration d'un Application Load Balancer partagé que vous créez et associez à votre environnement. Consultez également the section called “Application Load Balancer”. Pour plus d'informations sur la configuration de tous les types d'équilibreur de charge pris en charge par Elastic Beanstalk, consultez Équilibreur de charge pour votre environnement Elastic Beanstalk.

> **Note**
>
> Vous pouvez choisir le type d'équilibreur de charge que votre environnement utilise uniquement lors de la création de l'environnement. Vous pouvez modifier les paramètres pour gérer le comportement de l'équilibreur de charge de votre environnement d'exécution, mais pas en changer le type. Vous ne pouvez pas non plus passer d'un équilibreur de charge dédié à un équilibreur de charge partagé ou inversement.

Introduction

Un équilibreur de charge partagé est un équilibreur de charge que vous créez et gérez vous-même à l'aide du service Amazon Elastic Compute Cloud EC2 (Amazon), puis que vous utilisez dans plusieurs environnements Elastic Beanstalk.

Lorsque vous créez un environnement de mise à l'échelle à charge équilibrée et que vous choisissez d'utiliser un Application Load Balancer, Elastic Beanstalk crée un équilibreur de charge dédié à votre environnement par défaut. Pour comprendre ce qu'est un Application Load Balancer et comment il fonctionne dans un environnement Elastic Beanstalk, consultez l'introduction à la configuration d'un Application Load Balancer pour Elastic Beanstalk.

Dans certaines situations, vous pouvez économiser le coût d'avoir plusieurs équilibreurs de charge dédiés. Cela peut s’avérer utile avec plusieurs environnements, par exemple, si votre application est une suite de micro-services au lieu d'un service monolithique. Dans de tels cas, vous pouvez choisir d'utiliser un équilibreur de charge partagé.

Pour utiliser un équilibreur de charge partagé, créez-le d'abord dans Amazon EC2 et ajoutez un ou plusieurs écouteurs. Lors de la création d'un environnement Elastic Beanstalk, vous fournissez ensuite l'équilibreur de charge et choisissez un port d'écoute. Elastic Beanstalk associe l'écouteur au processus par défaut dans votre environnement. Vous pouvez ajouter des règles d'écouteur personnalisées pour acheminer le trafic à partir d'en-têtes et de chemins d'accès d'hôte spécifiques vers d'autres processus d'environnement.

Elastic Beanstalk ajoute une balise à l'équilibreur de charge partagé. Le nom de la balise est `elasticbeanstalk:shared-elb-environment-count`, et sa valeur est le nombre d'environnements partageant cet équilibreur de charge.

L'utilisation d'un équilibreur de charge partagé est différente de l'utilisation d'un équilibreur dédié de plusieurs façons.

Regarding	Application Load Balancer dédié	Application Load Balancer partagé
Gestion	Elastic Beanstalk crée et gère les écouteurs, les règles d'écoute et les processus (groupes cibles) de l'équilibreur de charge. Elastic Beanstalk les supprime également lorsque vous arrêtez votre environnement. Elastic Beanstalk peut définir la capture des journaux d'accès de l'équilib reur de charge, si vous choisissez cette option.	Vous créez et gérez l'équilibreur de charge et les écouteurs en dehors d'Elastic Beanstalk . Elastic Beanstalk crée et gère une règle par défaut et un processus par défaut, et vous pouvez ajouter des règles et des processus . Elastic Beanstalk supprime les règles et processus d'écoute ajoutés lors de la création de l'environnement.
Règles d'un écouteur	Elastic Beanstalk crée une règle par défaut pour chaque écouteur, pour acheminer l'ensemble du trafic vers le processus par défaut de l'écouteur.	Elastic Beanstalk associe une règle par défaut uniquement à un écouteur de port 80, le cas échéant. Si vous choisissez un autre port d'écoute par défaut, vous devez lui associer la règle par défaut (la console Elastic Beanstalk et l'interface de ligne de commande EB le font pour vous). Pour résoudre les conflits de condition de règle d'écoute entre les environnements partagean t l'équilibreur de charge, Elastic Beanstalk ajoute le CNAME de l'environnement à la règle d'écoute en tant que condition d'en-tête d'hôte. Elastic Beanstalk traite les paramètres de priorité de règle comme relatifs dans les environnements partageant l'équilibreur de charge et les mappe aux priorités absolues lors de la création.
Groupes de sécurité	Elastic Beanstalk crée un groupe de sécurité par défaut et l'attache à l'équilibreur de charge.	Vous pouvez configurer un ou plusieurs groupes de sécurité à utiliser pour l'équilibreur de charge. Si vous ne le faites pas, Elastic

Regarding	Application Load Balancer dédié	Application Load Balancer partagé
		Beanstalk vérifie si un groupe de sécurité existant géré par Elastic Beanstalk est déjà attaché à l'équilibreur de charge. Si ce n'est pas le cas, Elastic Beanstalk crée un groupe de sécurité et l'attache à l'équilibreur de charge. Elastic Beanstalk supprime ce groupe de sécurité lorsque le dernier environnement partageant l'équilibreur de charge s'arrête.
Mises à jour	Vous pouvez mettre à jour votre Application Load Balancer après la création de l'environnement. Vous pouvez modifier les écouteurs, les règles des écouteurs et les processus. Vous pouvez configure r la capture du journal d'accès de l'équilibreur de charge.	Vous ne pouvez pas utiliser Elastic Beanstalk pour configurer la capture du journal d'accès dans votre Application Load Balancer, et vous ne pouvez pas mettre à jour les écouteurs et les règles d'écoute après la création de l'environnement. Vous ne pouvez mettre à jour que les processus (groupes cibles). Pour configurer la capture du journal d'accès et pour mettre à jour les auditeurs et les règles d'écoute, utilisez Amazon. EC2

Configuration d'un Application Load Balancer partagé à l'aide de la console Elastic Beanstalk

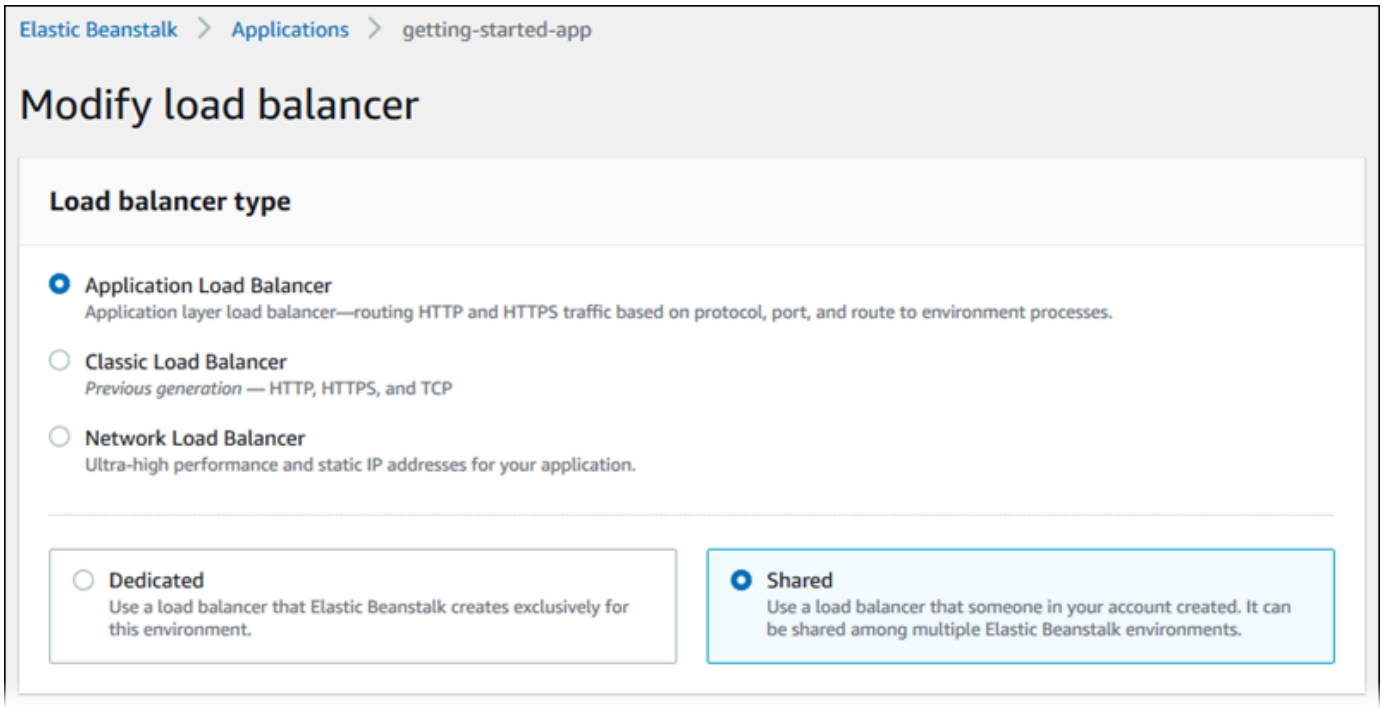
Vous pouvez utiliser la console Elastic Beanstalk pour configurer un Application Load Balancer partagé lors de la création de l'environnement. Vous pouvez sélectionner l'un des équilibreurs de charge partageables de votre compte à utiliser dans l'environnement, sélectionner le port de l'écouteur par défaut et configurer d'autres processus et règles de l'écouteur.

Vous ne pouvez pas modifier la configuration de votre Application Load Balancer partagé dans la console Application Load Balancer une fois votre environnement créé. Pour configurer les écouteurs, les règles d'écoute, les processus (groupes cibles) et la capture des journaux d'accès, utilisez Amazon. EC2

Pour configurer un Application Load Balancer dans la console Elastic Beanstalk lors de la création de l'environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements).
3. Choisissez Create a new environment (Créer un nouvel environnement) pour commencer à créer votre environnement.
4. Sur la page principale de l'assistant, avant de choisir Créer un environnement, choisissez Configurer plus d'options.
5. Choisissez le préréglage de configuration Haute disponibilité.

   Sinon, dans la catégorie de configuration Capacité, configurez un type d'environnement avec un Équilibrage de charge. Pour en savoir plus, consultez Capacity.
6. Dans la catégorie de configuration Load balancer (Équilibreur de charge), choisissez Edit (Modifier).
7. Sélectionnez l'option Application Load Balancer (Équilibreur de charge d'application), si elle n'est pas déjà sélectionnée, puis sélectionnez l'option Shared (Partagé).



8. Effectuez toutes les modifications de configuration de l'Application Load Balancer partagé exigées par votre environnement.

9. Choisissez Enregistrer, puis effectuez toutes les autres modifications de configuration exigées par votre environnement.
10. Choisissez Create environment.

Déclencheurs des Application Load Balancer partagé

- Application Load Balancer partagé
- Processus
- Rules

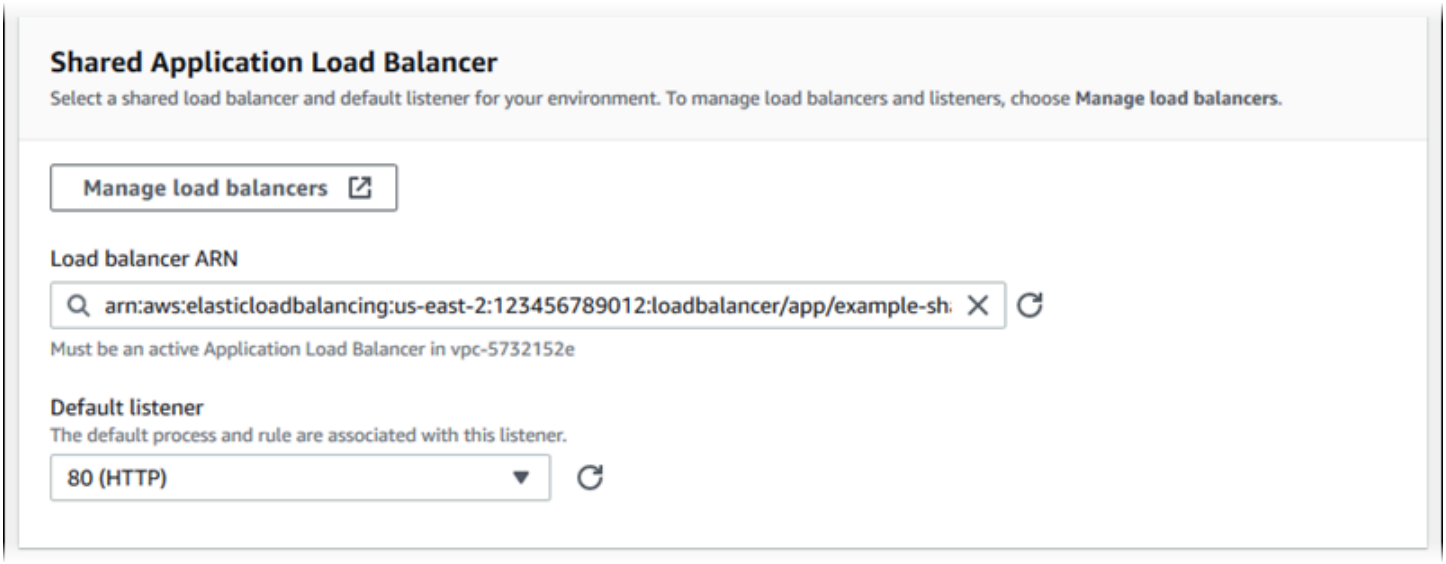
Application Load Balancer partagé

Utilisez cette section pour choisir un Application Load Balancer partagé pour votre environnement et configurer le routage du trafic par défaut.

Avant de configurer un Application Load Balancer partagé ici, utilisez Amazon EC2 pour définir au moins un Application Load Balancer à partager, avec au moins un écouteur, dans votre compte. Si vous ne l'avez pas déjà fait, vous pouvez choisir Manage load balancers (Gérer les équilibreurs de charge). Elastic Beanstalk ouvre EC2 la console Amazon dans un nouvel onglet du navigateur.

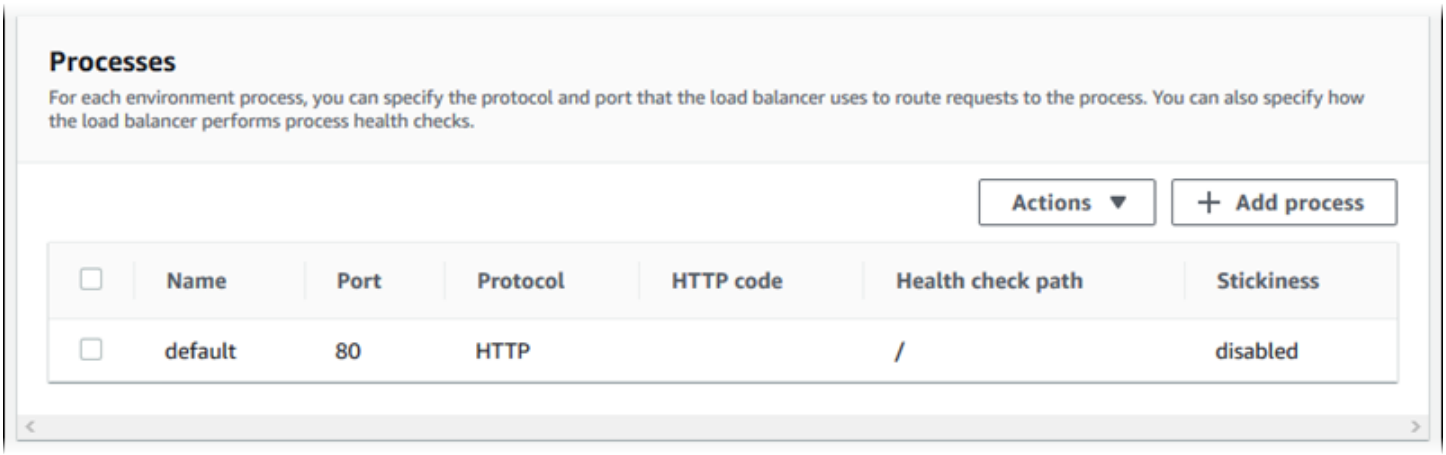
Une fois que vous avez terminé de configurer des équilibreurs de charge partagés en dehors d'Elastic Beanstalk, configurez les paramètres suivants dans cette section de la console :

- Load balancer ARN (ARN de l'équilibreur de charge) – Équilibreur de charge partagé à utiliser dans cet environnement. Sélectionnez dans une liste d'équilibreurs de charge ou entrez un Amazon Resource Name (ARN) d'équilibrage de charge.
- Default listener port (Port d'écoute par défaut) – Port du processus d'écoute que l'équilibreur de charge partagé écoute. Faites votre choix dans une liste de ports d'écouteur existants. Le trafic de cet écouteur avec le CNAME de l'environnement dans l'en-tête hôte est routé vers un processus par défaut dans cet environnement.



Processus

Utilisez cette liste pour spécifier les processus de votre équilibreur de charge. Un processus est une cible pour le routage du trafic par les écouteurs. Initialement, la liste affiche le processus par défaut, qui reçoit le trafic de l'écouteur par défaut.



Pour configurer un processus existant

1. Cochez la case en regard de son entrée de table, puis choisissez Actions, Modifier.
2. Utilisez la boîte de dialogue du processus Environnement pour modifier les paramètres, puis choisissez Enregistrer.

Pour ajouter un processus

1. Choisissez Ajouter un processus.

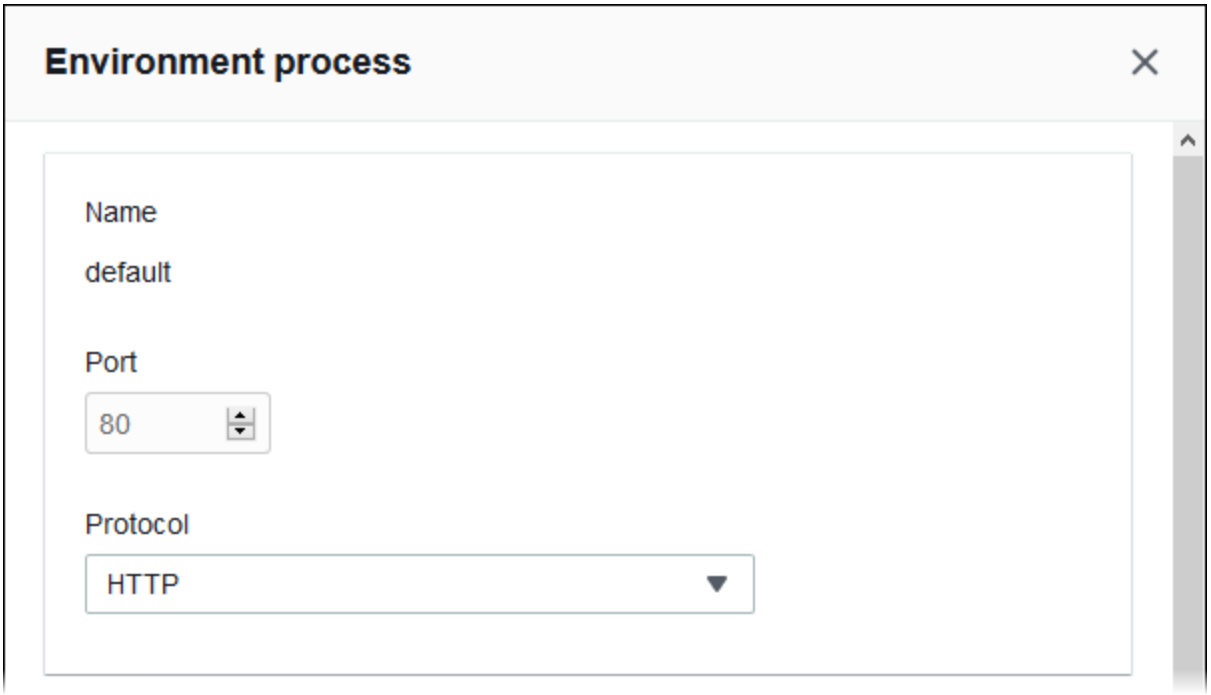
2. Dans la boîte de dialogue Processus d'environnement, configurez les paramètres souhaités, puis choisissez Ajouter.

Paramètres de la boîte de dialogue de processus d'environnement de l'Application Load Balancer

- Définition
- Surveillance de l'état
- Séances

Définition

Utilisez ces paramètres pour définir le processus : son nom et le port et le protocole sur lesquels il écoute les requêtes.

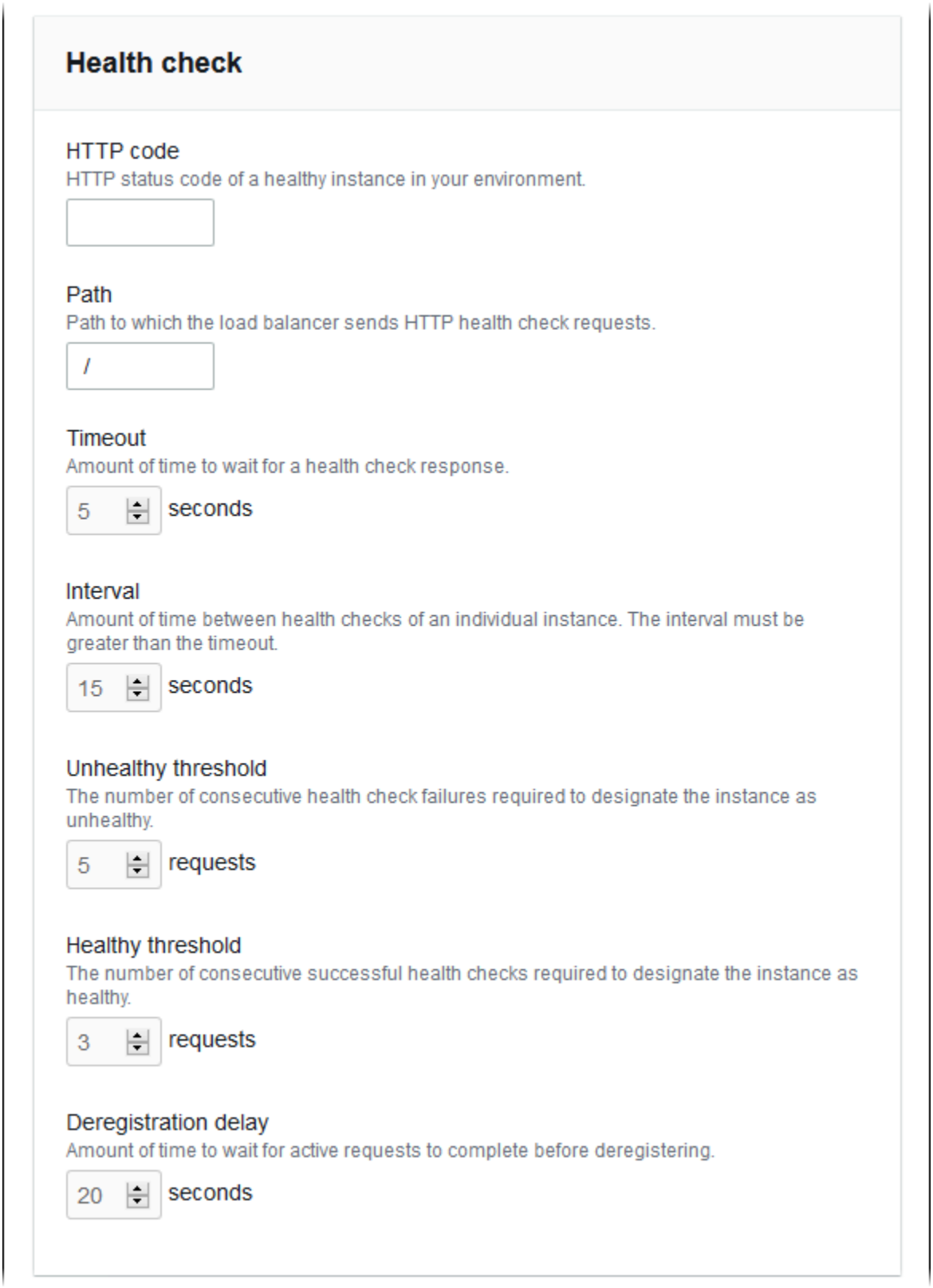


Surveillance de l'état

Utilisez les paramètres suivants pour configurer les processus des vérifications de l'état :

- HTTP code (Code HTTP) – Code d'état HTTP désignant un processus sain.
- Path (Chemin) – Chemin d'accès de la demande de vérification de l'état du processus.
- Timeout (Délai) – Durée, en secondes, d'attente d'une réponse de la vérification de l'état.

- Interval (Intervalle) – Durée, en secondes, entre les vérifications de l'état d'une instance. L'intervalle doit être supérieur au délai.
- Unhealthy threshold (Seuil de défectuosité), Healthy threshold (Seuil de bonne santé) – Nombre de vérifications de l'état qui doivent échouer ou réussir avant qu'Elastic Load Balancing modifie l'état de santé d'une instance.
- Deregistration delay (Retard d'annulation d'enregistrement) – Délai, en secondes, d'attente de la fin des requêtes actives avant l'annulation de l'enregistrement d'une instance.

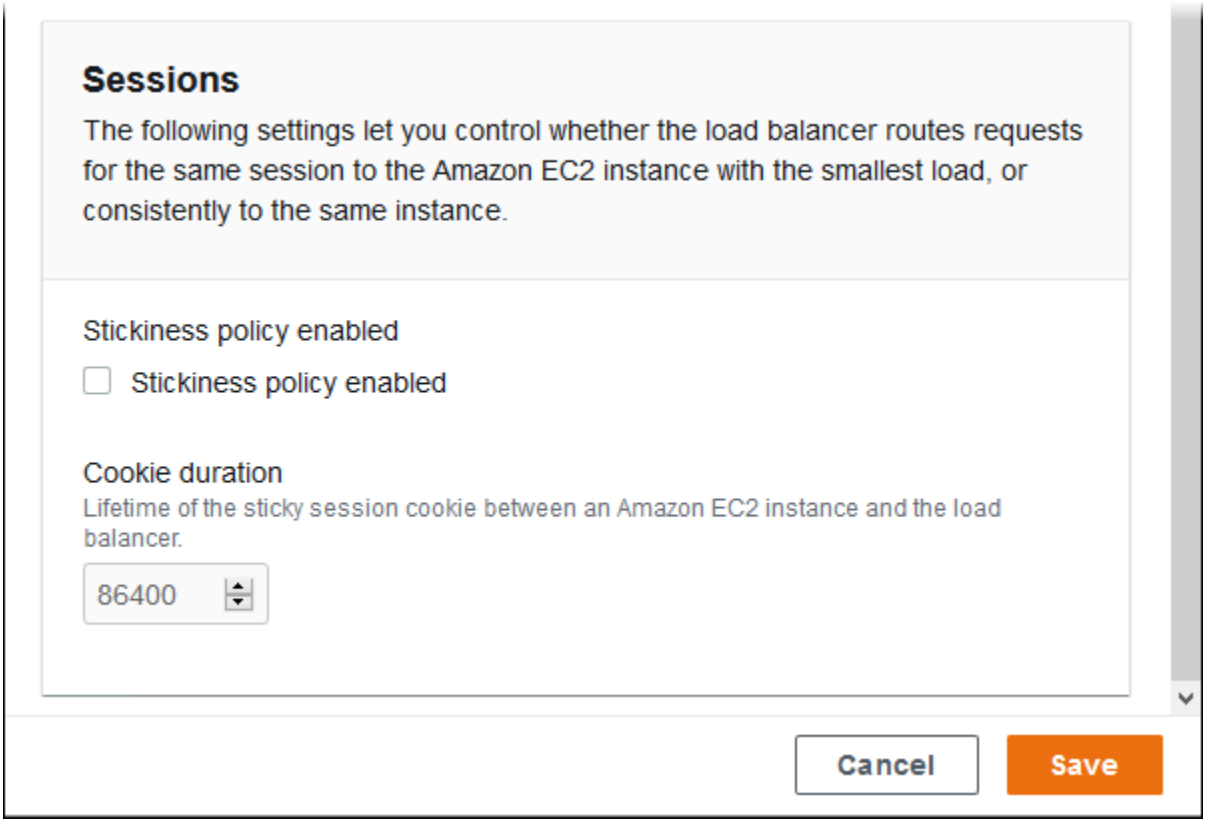

Health check
HTTP code
HTTP status code of a healthy instance in your environment.
Path
Path to which the load balancer sends HTTP health check requests.
/
Timeout
Amount of time to wait for a health check response.
5
seconds
Interval
Amount of time between health checks of an individual instance. The interval must be greater than the timeout.
15
seconds
Unhealthy threshold
The number of consecutive health check failures required to designate the instance as unhealthy.
5
requests
Healthy threshold
The number of consecutive successful health checks required to designate the instance as healthy.
3
requests
Deregistration delay
Amount of time to wait for active requests to complete before deregistering.
20
seconds

> **Note**
>
> La vérification de l'état Elastic Load Balancing n'a pas d'incidence sur le comportement de vérification de l'état du groupe Auto Scaling d'un environnement. Les instances qui échouent à un test de santé d'Elastic Load Balancing ne sont pas automatiquement remplacées par Amazon EC2 Auto Scaling, sauf si vous configurez manuellement Amazon EC2 Auto Scaling pour le faire. Consultez Paramètre de vérification de l'état Auto Scaling pour votre environnement Elastic Beanstalk pour plus de détails.

Pour plus d'informations sur les vérifications de l'état et leur influence sur l'état global de votre environnement, consultez Création de rapports d'intégrité de base.

Séances

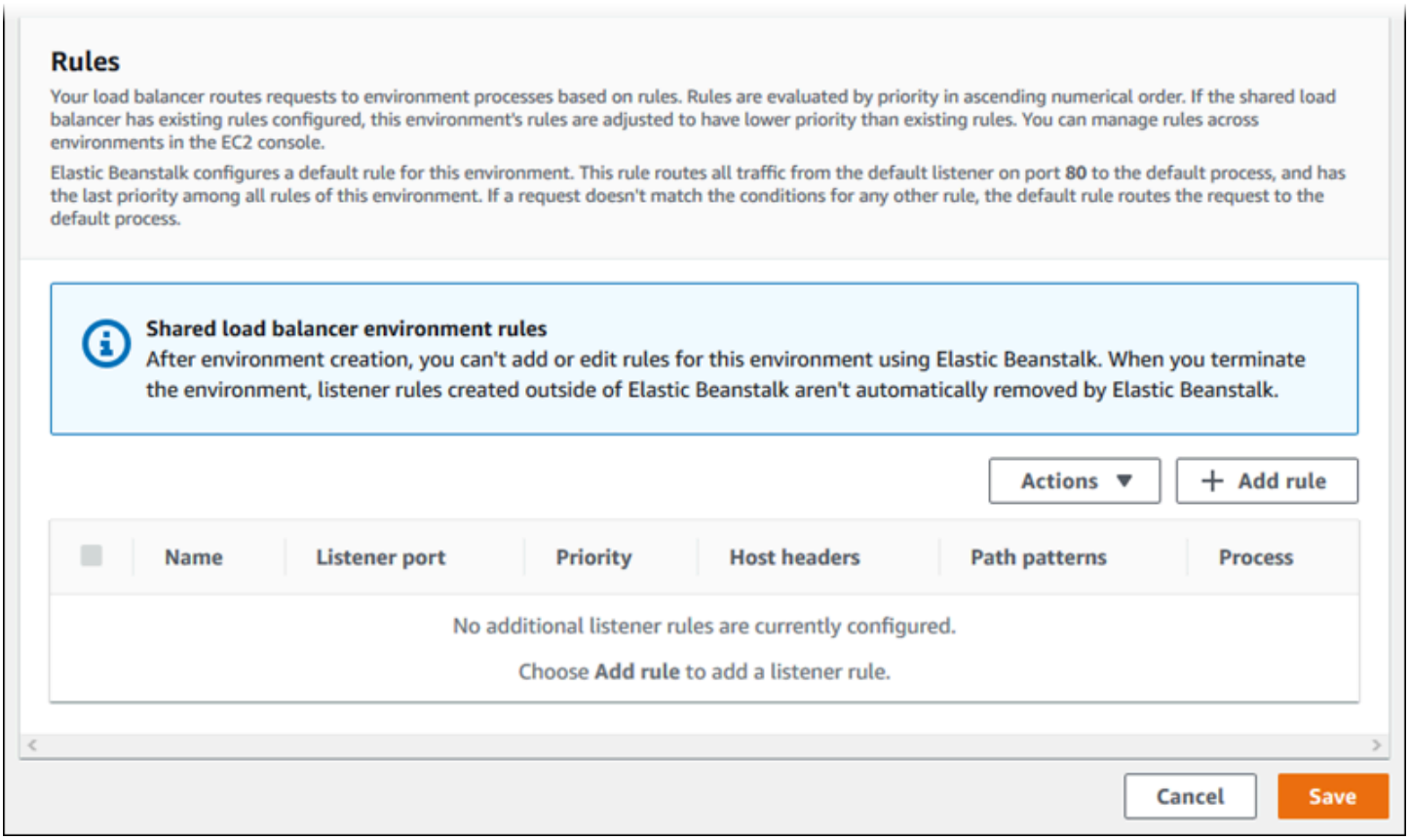
Activez ou désactivez la case Politique de permanence activée pour activer ou désactiver les sessions permanentes. Utilisez l'option Durée du cookie pour configurer la durée d'une session permanente, pouvant aller jusqu'à **`604800`** secondes.



Rules

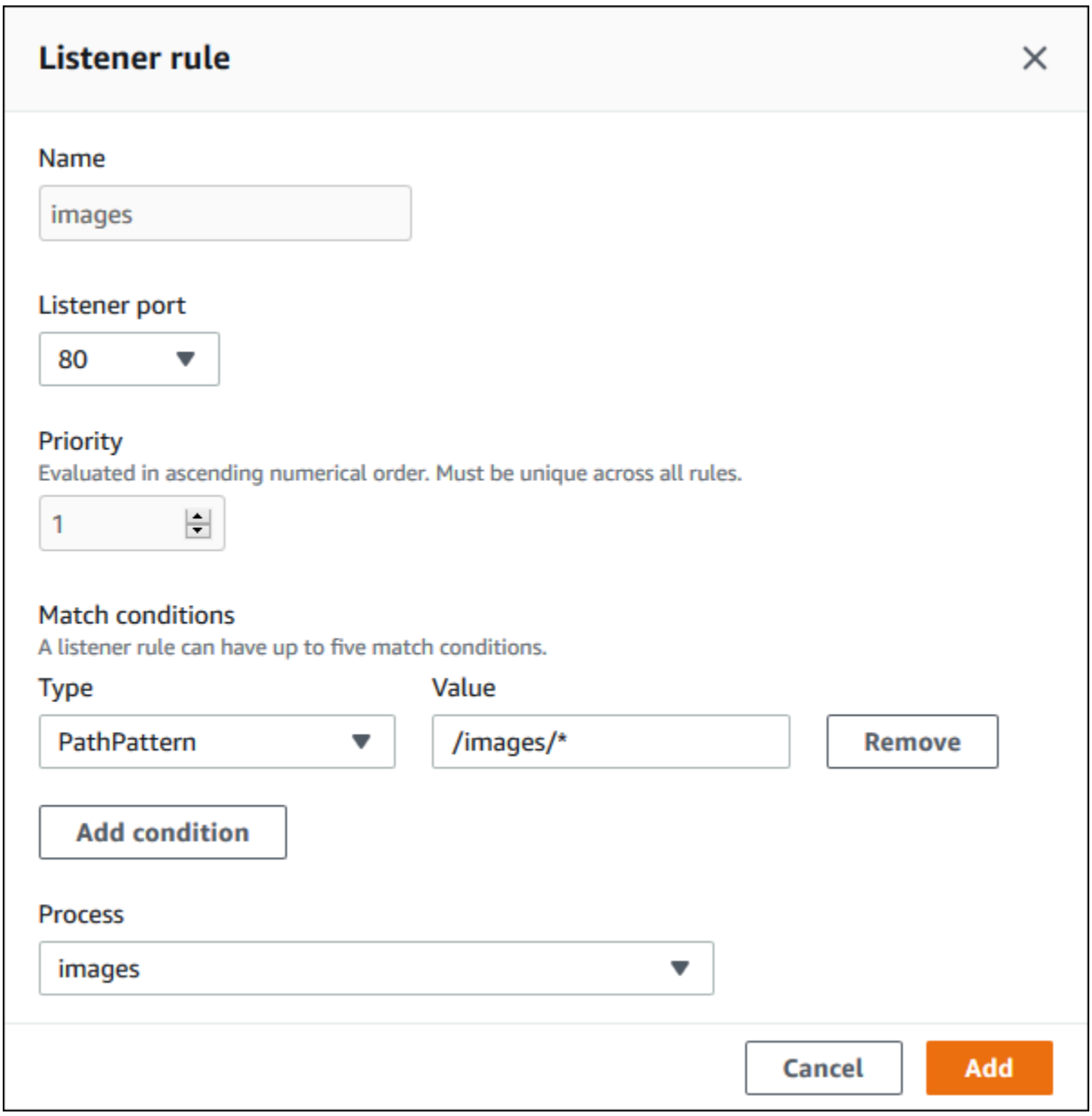
Utilisez cette liste pour spécifier des règles d'écouteur personnalisées pour votre équilibreur de charge. Une règle mappe les requêtes que l'écouteur reçoit sur un modèle de chemin spécifique à un processus cible. Chaque écouteur peut avoir plusieurs règles, acheminer les demandes sur différents chemins vers différents processus sur des instances des différents environnements partageant l'écouteur.

Les règles ont des priorités numériques qui déterminent la priorité selon laquelle elles sont appliquées aux demandes entrantes. Elastic Beanstalk ajoute une règle par défaut qui achemine l'ensemble du trafic de l'écouteur par défaut vers le processus par défaut de votre nouvel environnement. La priorité de la règle par défaut est la plus basse ; elle n'est appliquée que si aucune autre règle pour le même écouteur ne correspond à la requête entrante. Initialement, si vous n'avez pas ajouté de règles personnalisées, la liste est vide. La règle par défaut n'est pas affichée.



Vous pouvez modifier les paramètres d'une règle existante ou ajouter une nouvelle règle. Pour commencer la modification d'une règle de la liste ou l'ajout d'une règle, utilisez la procédure définie pour la liste d'écouteurs. La boîte de dialogue Règle d'écouteur s'ouvre, avec les paramètres suivants :

- Name (Nom) – Nom de la règle.
- Listener port (Port d'écoute) – Port d'écoute auquel la règle s'applique.
- Priority (Priorité) – Priorité de la règle. Une valeur de priorité plus faible a une priorité plus élevée. Les priorités des règles d'un écouteur doivent être uniques. Elastic Beanstalk traite les priorités des règles comme relatives dans les environnements de partage et les mappe aux priorités absolues lors de la création.
- Match conditions (Conditions de correspondance) – Liste des conditions d'URL de requête auxquelles la règle s'applique. Il existe deux types de conditions : HostHeader(la partie domaine de l'URL) et PathPattern(la partie chemin de l'URL). Une condition est réservée au sous-domaine d'environnement et vous pouvez ajouter jusqu'à quatre conditions. Chaque valeur de condition a jusqu'à 128 caractères et peut inclure des caractères génériques.
- Process (Processus) – Processus auquel l'équilibreur de charge achemine les demandes qui correspondent à la règle.



Exemple : utiliser un Application Load Balancer partagé pour une application sécurisée micro-service-based

Dans cet exemple, votre application se compose de plusieurs microservices, chacun implémenté en tant qu'environnement Elastic Beanstalk. En outre, vous avez besoin d'un chiffrement end-to-end du trafic. Nous allons démontrer l'un des environnements de microservices, qui a un processus principal pour les demandes des utilisateurs et un processus distinct pour le traitement des demandes administratives.

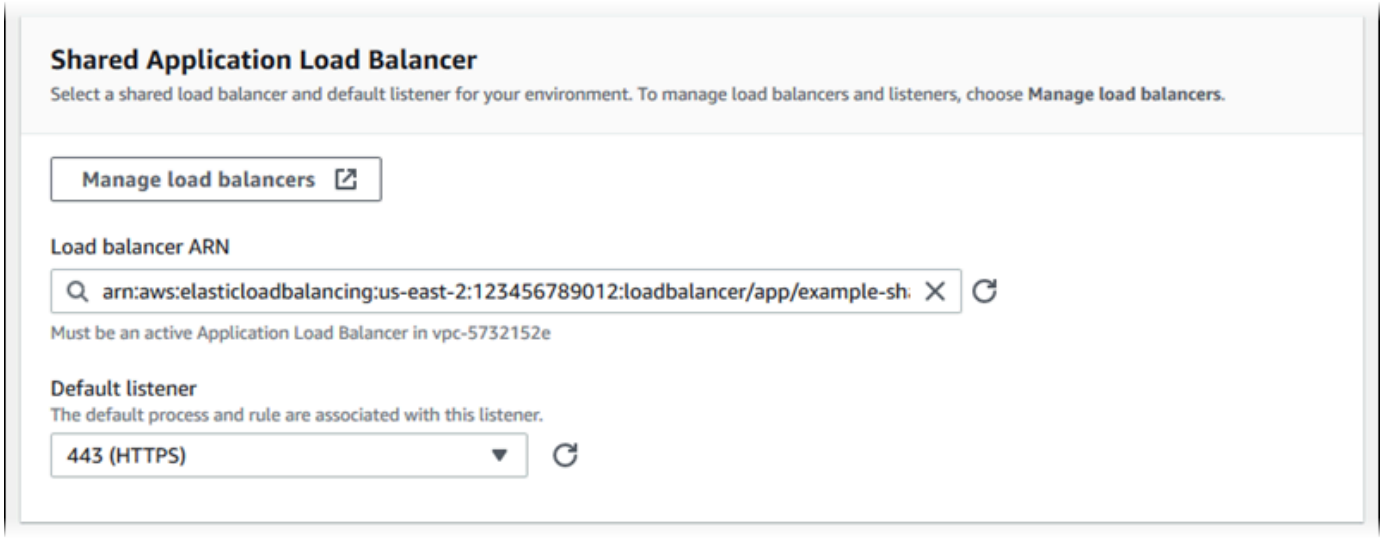
Pour répondre à ces exigences, utilisez Amazon EC2 pour créer un Application Load Balancer que vous partagerez avec vos microservices. Ajoutez un écouteur sécurisé sur le port 443 et le protocole

HTTPS. Ajoutez ensuite plusieurs certificats SSL à l'écouteur, un par domaine de microservice. Pour plus d'informations sur la création de Application Load Balancer et de l'écouteur sécurisé, consultez Création d'un Application Load Balancer et Création d'un écouteur HTTPS pour votre Application Load Balancer dans le Guide de l'utilisateur des Application Load Balancer.

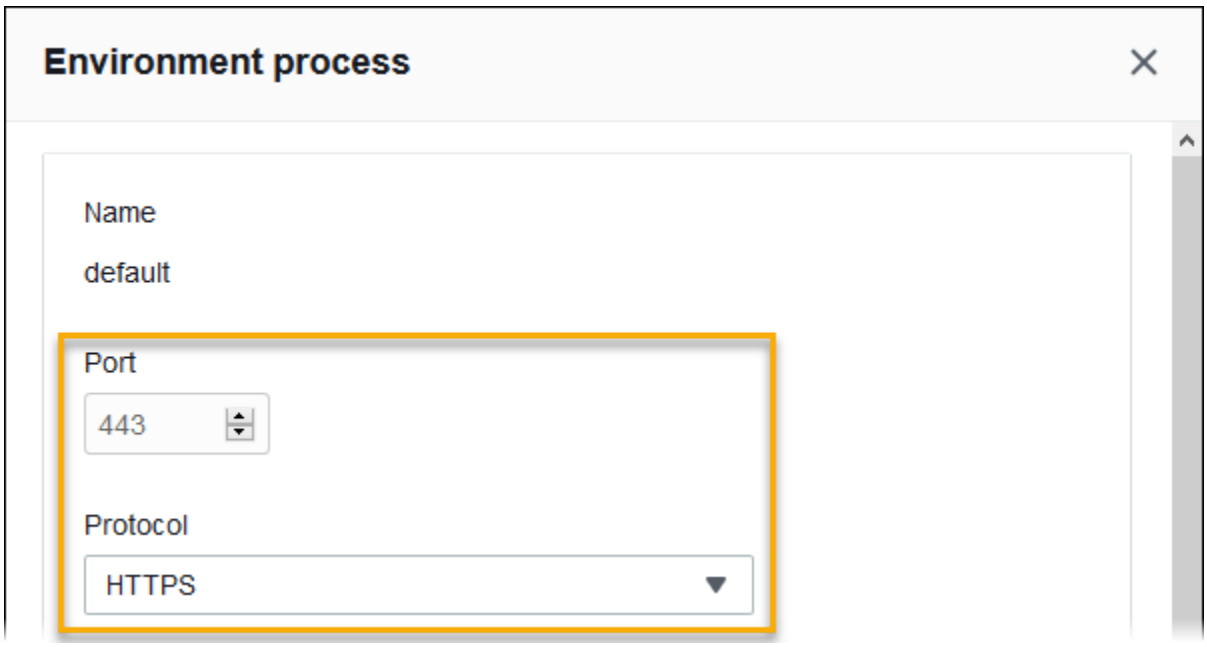
Dans Elastic Beanstalk, configurez chaque environnement de microservice pour utiliser l'Application Load Balancer partagé et définissez le port de l'écouteur par défaut sur 443. Dans le cas de l'environnement présenté ici, indiquez que le processus par défaut écoute le port 443 sur HTTPS, et ajoutez un processus et une règle d'écouteur pour le trafic admin sur un chemin différent.

Pour configurer l'équilibreur de charge partagé de cet exemple

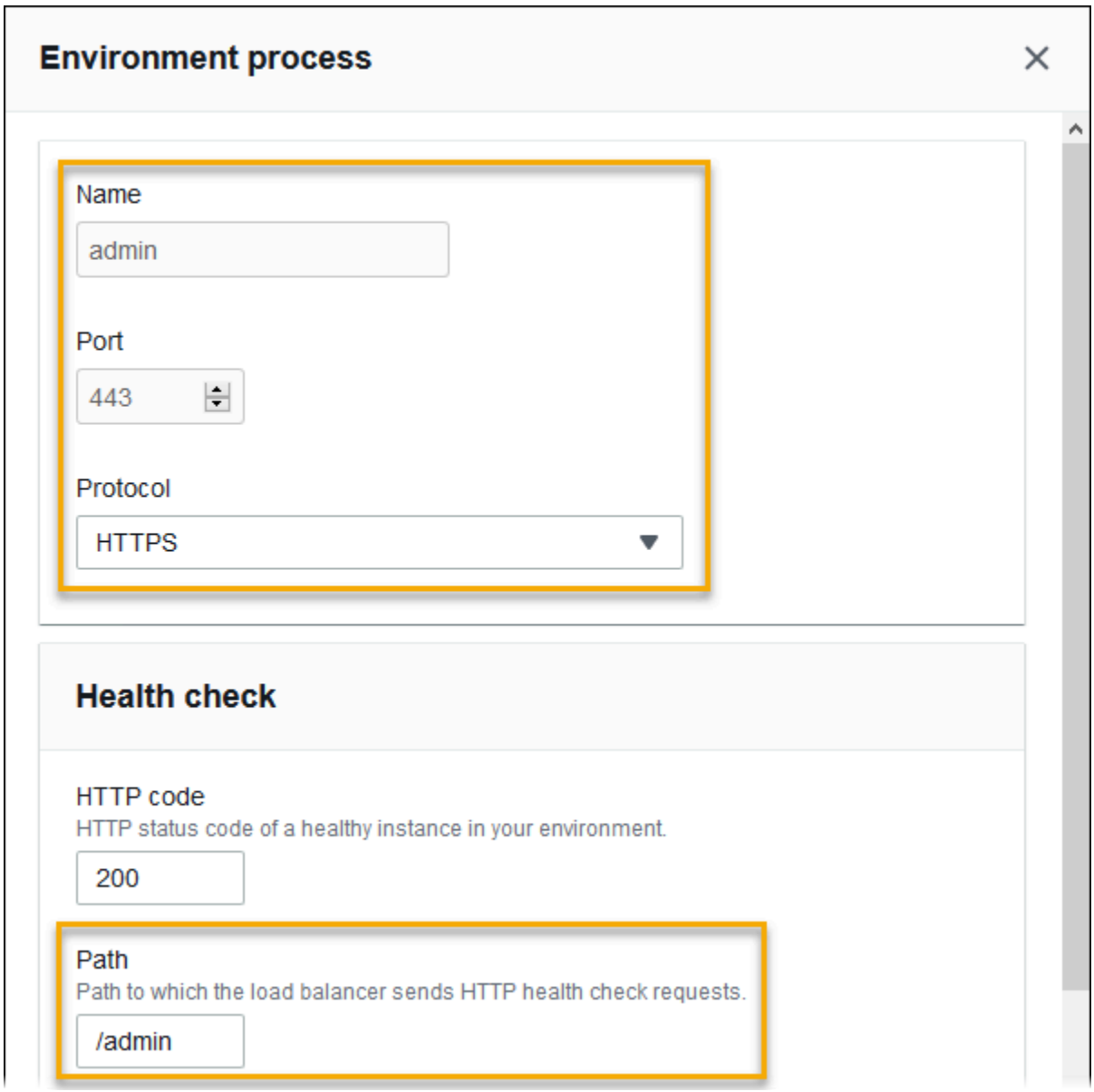
1. Dans la section Shared Application Load Balancer (Équilibreur de charge Application Load Balancer partagé), sélectionnez votre équilibreur de charge, puis, pour Default listener port (Port d'écoute par défaut), sélectionnez **443**. Le port de l'écouteur doit déjà être sélectionné s'il s'agit du seul écouteur que possède l'équilibreur de charge.



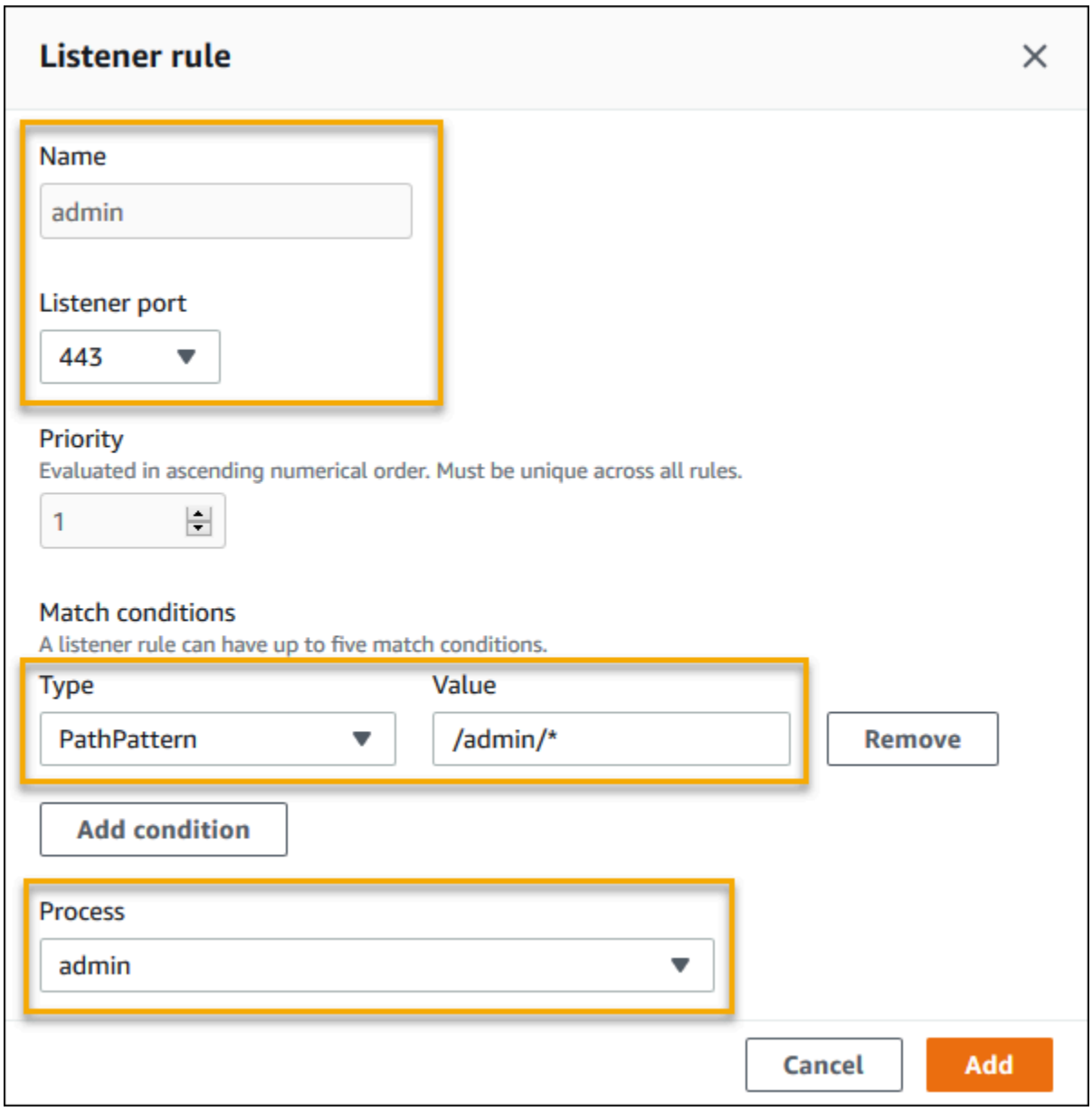
2. Configurez le processus par défaut sur HTTPS. Sélectionnez le processus par défaut, puis, pour Actions, choisissez Edit (Modifier). Pour Port, entrez **443**. Pour Protocole, sélectionnez **HTTPS**.



3. Ajoutez un processus d'administration. Pour Nom, saisissez **`admin`**. Pour Port, entrez **`443`**. Pour Protocole, sélectionnez **`HTTPS`**. Sous Vérification de l'état, pour Chemin, tapez **`/admin`**.



4. Ajoutez une règle pour le trafic d'administration. Pour Nom, saisissez **`admin`**. Pour Port d'écouteur, entrez **`443`**. Pour les conditions de correspondance, ajoutez un PathPatternavec la valeur**`/admin/*`**. Pour Processus, sélectionnez **`admin`**.



Configuration d'un équilibreur de charge Application Load Balancer partagé à l'aide de l'interface de ligne de commande EB

L'interface de ligne de commande EB vous invite à choisir un type d'équilibreur de charge lorsque vous exécutez la commande eb create. Si vous choisissez `application` (valeur par défaut) et si votre compte dispose d'au moins un équilibreur de charge Application Load Balancer partageable, l'interface de ligne de commande EB vous demande également si vous souhaitez utiliser un équilibreur de charge Application Load Balancer partagé. Si vous répondez **y**, vous êtes également invité à sélectionner l'équilibreur de charge et le port par défaut.

```
$ eb create
Enter Environment Name
(default is my-app): test-env
Enter DNS CNAME prefix
(default is my-app): test-env-DLW24ED23SF

Select a load balancer type
1) classic
2) application
3) network
(default is 2):

Your account has one or more sharable load balancers. Would you like your new
 environment to use a shared load balancer?(y/N) y

Select a shared load balancer
1)MySharedALB1 - arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/app/
MySharedALB1/6d69caa75b15d46e
2)MySharedALB2 - arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/app/
MySharedALB2/e574ea4c37ad2ec8
(default is 1): 2

Select a listener port for your shared load balancer
1) 80
2) 100
3) 443
(default is 1): 3
```

Vous pouvez également spécifier un équilibreur de charge partagé à l'aide des options de commande.

```
$ eb create test-env --elb-type application --shared-lb MySharedALB2 --shared-lb-
port 443
```

Espaces de noms d'équilibreurs de charge Application Load Balancer partagés

Vous trouverez les paramètres liés aux équilibreurs Application Load Balancers partagés dans les espaces de noms suivants :

- `aws:elasticbeanstalk:environment` – Choisissez le type d'équilibreur de charge pour l'environnement et indiquez à Elastic Beanstalk que vous allez utiliser un équilibreur de charge partagé.

Vous ne pouvez pas définir ces deux options dans les fichiers de configuration (.Ebextensions).

- `aws:elbv2:loadbalancer` – Configurez l'ARN et les groupes de sécurité partagés de l'équilibreur de charge Application Load Balancer.
- `aws:elbv2:listener` – Associez les écouteurs de l'équilibreur de charge Application Load Balancer partagé avec l'environnement en listant les règles d'écoute.
- `aws:elbv2:listenerrule` – Configurez les règles d'écoute qui acheminent le trafic vers différents processus, selon le chemin de la demande. Les règles sont spécifiques aux équilibreurs de charge Application Load Balancer, qu'ils soient dédiés ou partagés.
- `aws:elasticbeanstalk:environment:process` – Configurez les vérifications de l'état et spécifiez le port et le protocole des processus qui s'exécutent sur les instances de votre environnement.

Example extensions .eb/ .config application-load-balancer-shared

Pour commencer avec un équilibreur de charge Application Load Balancer partagé, utilisez la console Elastic Beanstalk, l'interface de ligne de commande EB ou l'API pour définir le type d'équilibreur de charge sur `application` et choisissez d'utiliser un équilibreur de charge partagé. Utilisez un fichier de configuration pour configurer l'équilibreur de charge partagé.

```
option_settings:
  aws:elbv2:loadbalancer:
    SharedLoadBalancer: arn:aws:elasticloadbalancing:us-
east-2:123456789012:loadbalancer/app/MySharedALB2/e574ea4c37ad2ec8
```

> **Note**
>
> Vous ne pouvez configurer cette option que lors de la création de l'environnement.

Example extensions .eb/ .config alb-shared-secure-listener

Le fichier de configuration suivant sélectionne un écouteur sécurisé par défaut sur le port 443 pour l'équilibreur de charge partagé et définit le processus par défaut pour écouter le port 443.

```
option_settings:
  aws:elbv2:loadbalancer:
```

```
    SharedLoadBalancer: arn:aws:elasticloadbalancing:us-
east-2:123456789012:loadbalancer/app/MySharedALB2/e574ea4c37ad2ec8
  aws:elbv2:listener:443:
    rules: default
  aws:elasticbeanstalk:environment:process:default:
    Port: '443'
    Protocol: HTTPS
```

Example extensions .eb/ .config alb-shared-admin-rule

Le fichier de configuration suivant s'appuie sur l'exemple précédent et ajoute une règle qui achemine le trafic avec un chemin de requête de `/admin` vers un processus nommé `admin` qui écoute sur le port 4443.

```
option_settings:
  aws:elbv2:loadbalancer:
    SharedLoadBalancer: arn:aws:elasticloadbalancing:us-
east-2:123456789012:loadbalancer/app/MySharedALB2/e574ea4c37ad2ec8
  aws:elbv2:listener:443:
    rules: default,admin
  aws:elasticbeanstalk:environment:process:default:
    Port: '443'
    Protocol: HTTPS
  aws:elasticbeanstalk:environment:process:admin:
    HealthCheckPath: /admin
    Port: '4443'
    Protocol: HTTPS
  aws:elbv2:listenerrule:admin:
    PathPatterns: /admin/*
    Priority: 1
    Process: admin
```

Configuration d'un Network Load Balancer

Lorsque vous activez l'équilibrage de charge, votre AWS Elastic Beanstalk environnement est équipé d'un équilibreur de charge Elastic Load Balancing pour répartir le trafic entre les instances de votre environnement. Elastic Load Balancing prend en charge plusieurs types d'équilibreur de charge. Pour en savoir plus, consultez le Guide de l'utilisateur Elastic Load Balancing. Elastic Beanstalk peut créer un équilibreur de charge pour vous, ou vous permettre de spécifier un équilibreur de charge partagé que vous avez créé.

Cette rubrique décrit la configuration d'un Network Load Balancer créé par Elastic Beanstalk et dédié à votre environnement. Pour plus d'informations sur la configuration de tous les types d'équilibreur de charge pris en charge par Elastic Beanstalk, consultez Équilibreur de charge pour votre environnement Elastic Beanstalk.

> **Note**
>
> Vous pouvez choisir le type d'équilibreur de charge que votre environnement utilise uniquement lors de la création de l'environnement. Vous pouvez modifier les paramètres pour gérer le comportement de l'équilibreur de charge de votre environnement d'exécution, mais pas en changer le type.

Introduction

Avec un Network Load Balancer, l'écouteur par défaut accepte les demandes TCP sur le port 80 et les distribue aux instances de votre environnement. Vous pouvez configurer le comportement de la vérification de l'état, configurer le port d'écoute ou ajouter un écouteur sur un autre port.

> **Note**
>
> Contrairement à un Classic Load Balancer ou Application Load Balancer, un Network Load Balancer ne peut pas avoir d'écouteurs HTTP or HTTPS de couche de transport (couche 7). Il ne prend en charge que les écouteurs TCP de la couche de transport (couche 4). Le trafic HTTP et HTTPS peut être acheminé vers votre environnement via TCP. Pour établir des connexions HTTPS sécurisées entre les clients web et votre environnement, installez un certificat auto-signé sur les instances de l'environnement, et configurez les instances pour écouter sur le port approprié (généralement, le port 443) et arrêtez les connexions HTTPS. La configuration varie selon la plateforme. Pour obtenir des instructions, consultez Configuration de la terminaison HTTPS sur l'instance. Ensuite, configurez votre Network Load Balancer pour ajouter un écouteur qui est mappé à un processus à l'écoute sur le port approprié.

Un Network Load Balancer prend en charge les vérifications de l'état actives. Ces vérifications sont basées sur les messages transmis au chemin d'accès racine (`/`). Un Network Load Balancer prend également en charge les vérifications de l'état passives. Il détecte automatiquement les instances backend défectueuses et achemine uniquement le trafic vers des instances saines.

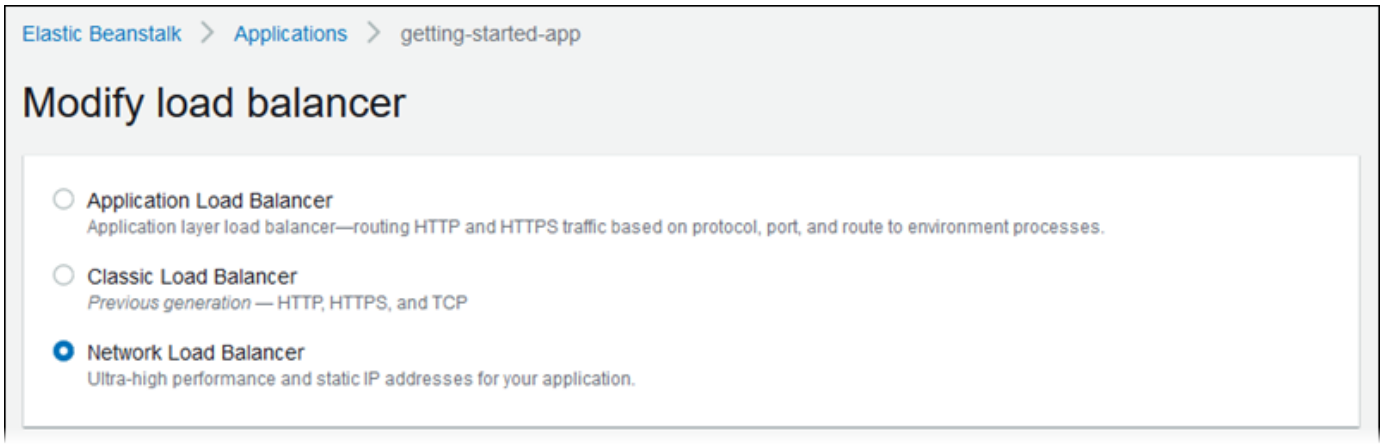
Configuration d'un Network Load Balancer à l'aide de la console Elastic Beanstalk

Vous pouvez utiliser la console Elastic Beanstalk pour configurer les écouteurs et les processus d'un Network Load Balancer lors de la création de l'environnement ou plus tard alors que votre environnement est en cours d'exécution.

Pour configurer un Network Load Balancer dans la console Elastic Beanstalk lors de la création de l'environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements).
3. Choisissez Create a new environment (Créer un nouvel environnement) pour commencer à créer votre environnement.
4. Sur la page principale de l'assistant, avant de choisir Créer un environnement, choisissez Configurer plus d'options.
5. Choisissez le préréglage de configuration Haute disponibilité.

   Sinon, dans la catégorie de configuration Capacité, configurez un type d'environnement avec un Équilibrage de charge. Pour en savoir plus, consultez Capacity.
6. Dans la catégorie de configuration Load balancer (Équilibreur de charge), choisissez Edit (Modifier).
7. Sélectionnez l'option Network Load Balancer (Équilibreur de charge du réseau) si elle n'est pas déjà sélectionnée.



8. Effectuez toutes les modifications de configuration du Network Load Balancer exigées par votre environnement.

9. Choisissez Enregistrer, puis effectuez toutes les autres modifications de configuration exigées par votre environnement.
10. Choisissez Create environment.

Pour configurer un Network Load Balancer d'un environnement en cours d'exécution dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Load balancer (Équilibreur de charge), choisissez Edit (Modifier).

> **Note**
>
> Si la catégorie de configuration Load balancer (Équilibreur de charge) ne dispose pas du bouton Edit (Modifier), cela signifie que votre environnement ne dispose pas d'un équilibreur de charge. Pour apprendre à en configurer un, consultez Changement de type d'environnement.

5. Effectuez les modifications de configuration du Network Load Balancer exigées par votre environnement.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

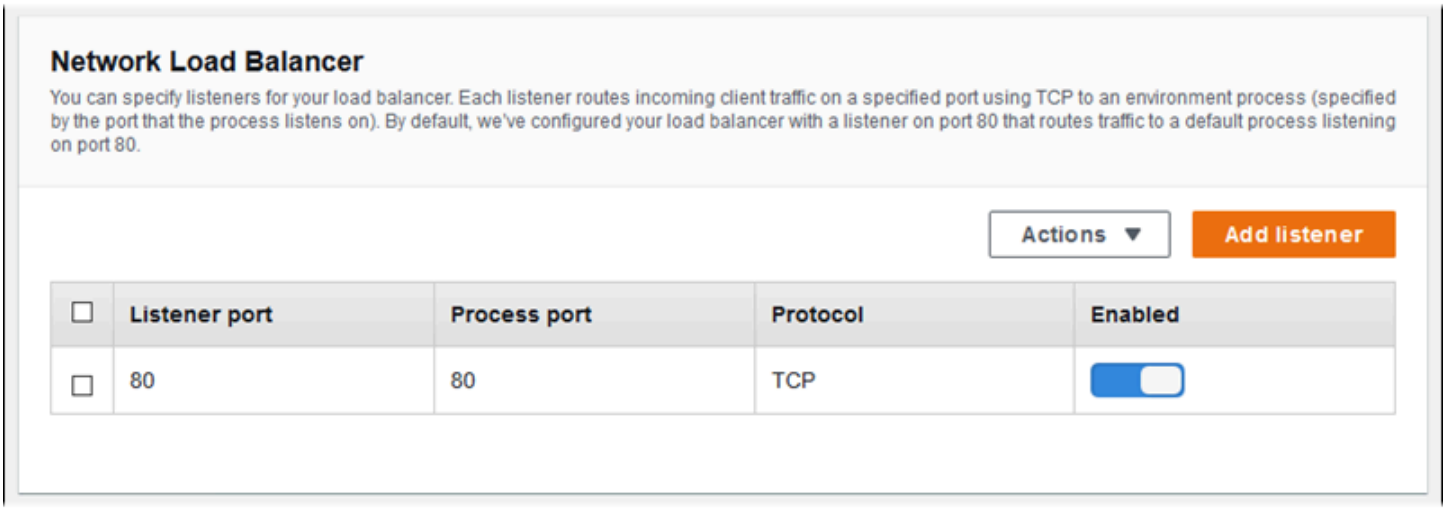
Paramètres du Network Load Balancer

- Écouteurs
- Processus

Écouteurs

Utilisez cette liste pour spécifier plusieurs écouteurs pour votre équilibreur de charge. Chaque écouteur achemine le trafic client entrant sur un port spécifié vers un processus spécifié sur vos

instances. Initialement, la liste indique l'écouteur par défaut, qui achemine le trafic HTTP entrant sur le port 80 pour un processus nommé default, qui écoute le port HTTP 80.



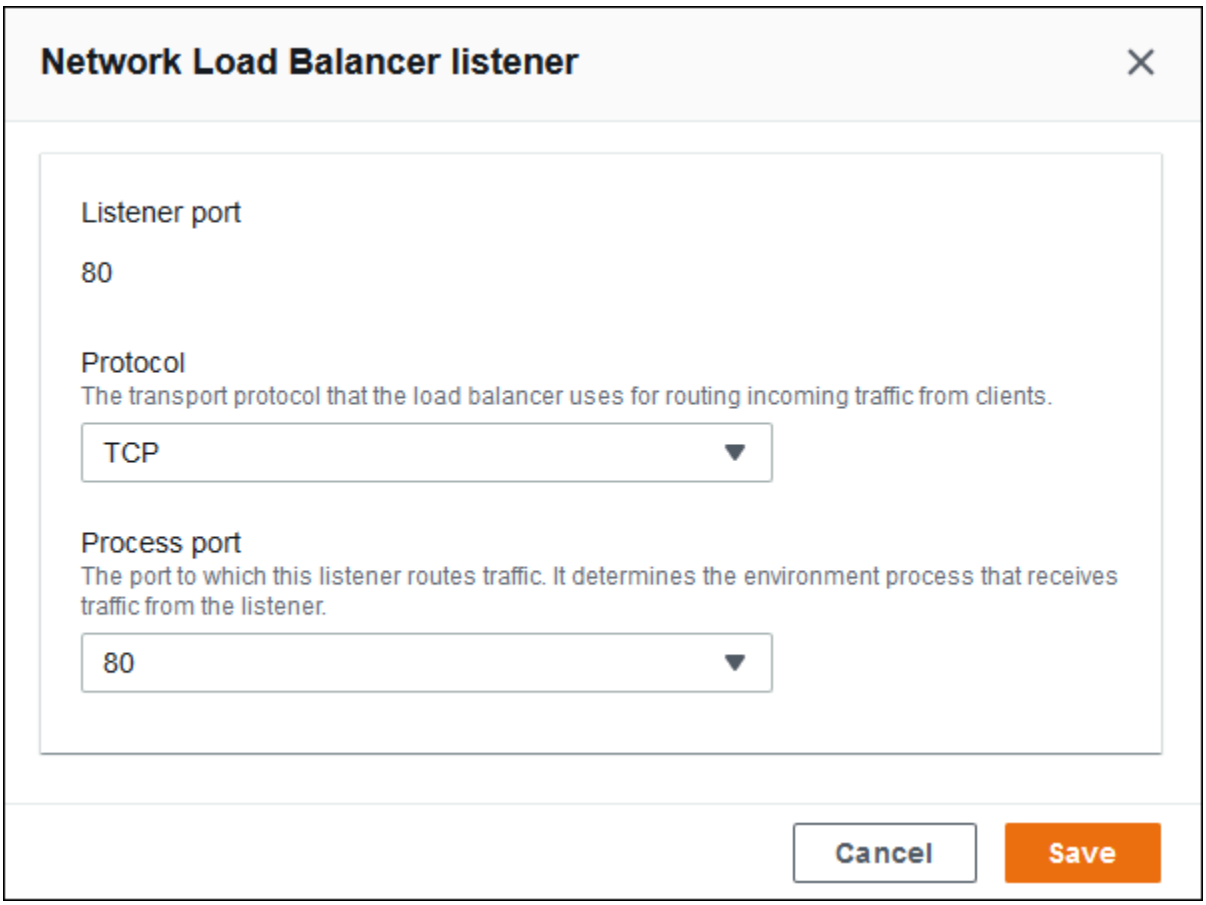
Pour configurer un écouteur existant

1. Cochez la case en regard de son entrée de table, puis choisissez Actions, Modifier.
2. Utilisez la boîte de dialogue Network Load Balancer listener (Écouteur Network Load Balancer) pour modifier les paramètres, puis choisissez Save (Enregistrer).

Pour ajouter un écouteur

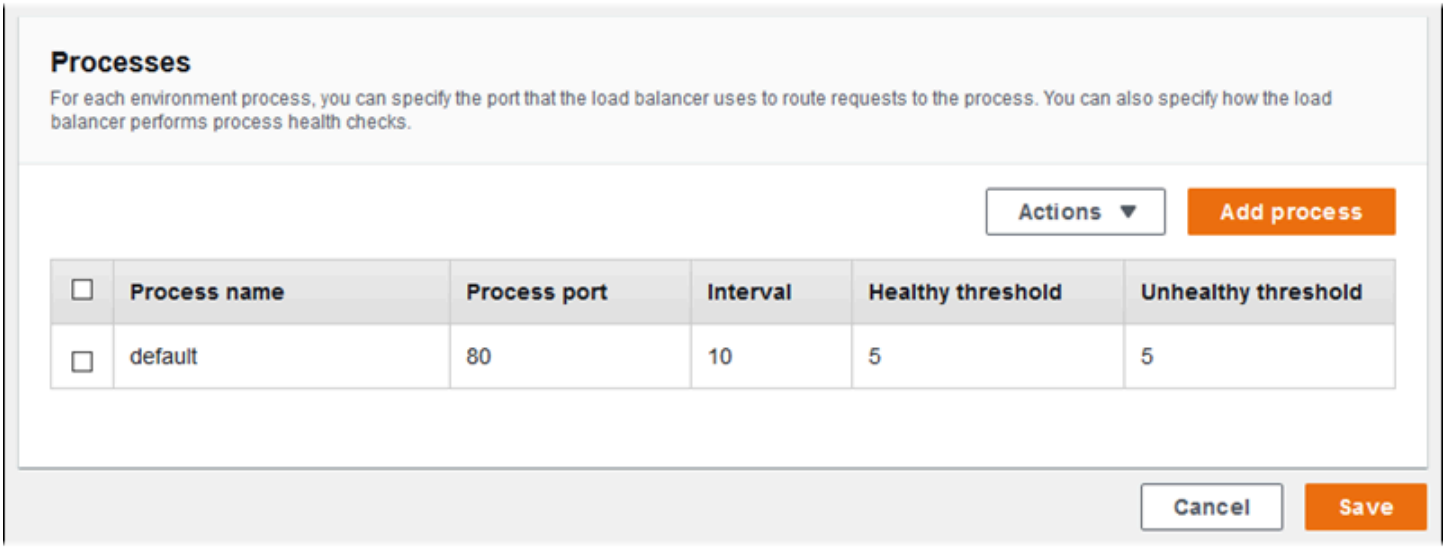
1. Choisissez Add listener (Ajouter un écouteur).
2. Dans la boîte de dialogue Network Load Balancer listener (Écouteur Network Load Balancer), configurez les paramètres requis, puis sélectionnez Ajouter.

Utilisez la boîte de dialogue Network Load Balancer listener (Écouteur Network Load Balancer) pour configurer le port sur lequel l'écouteur écoute le trafic, et choisir le processus vers lequel vous souhaitez acheminer le trafic (spécifié par le port que le processus écoute).



Processus

Utilisez cette liste pour spécifier les processus pour votre équilibreur de charge. Un processus est une cible pour le routage du trafic par les écouteurs. Chaque écouteur achemine le trafic client entrant sur un port spécifié vers un processus spécifié sur vos instances. Initialement, la liste indique le processus par défaut, qui écoute le trafic entrant sur le port 80.



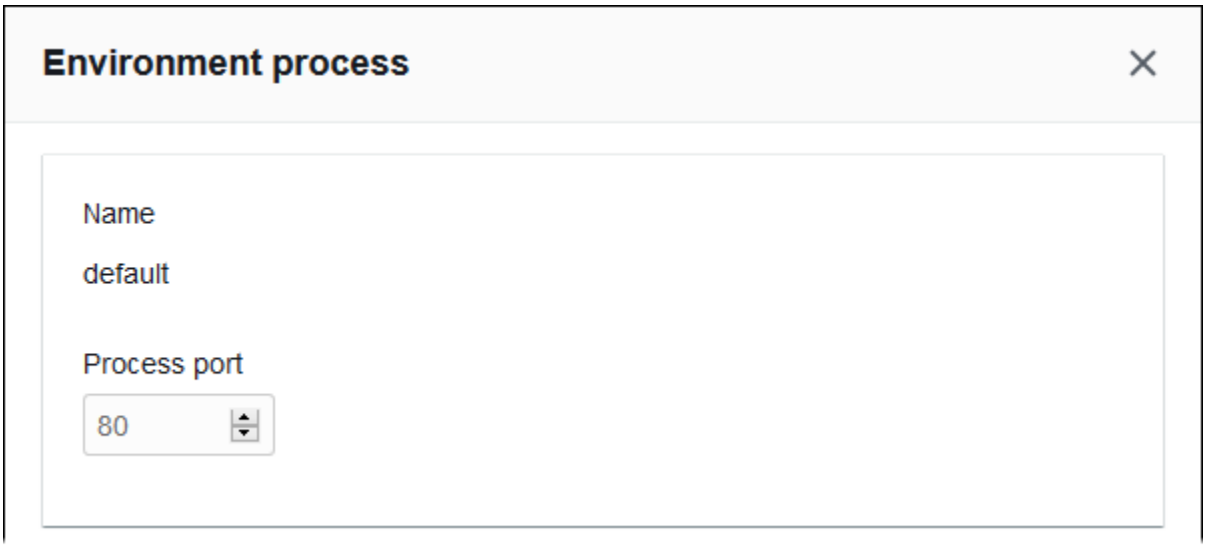
Vous pouvez modifier les paramètres d'un processus existant ou ajouter un nouveau processus. Pour commencer la modification d'un processus de la liste ou l'ajout d'un processus, utilisez la procédure définie pour la liste d'écouteurs. La boîte de dialogue Processus d'environnement s'ouvre.

Paramètres de la boîte de dialogue de processus d'environnement du Network Load Balancer

- Définition
- Surveillance de l'état

Définition

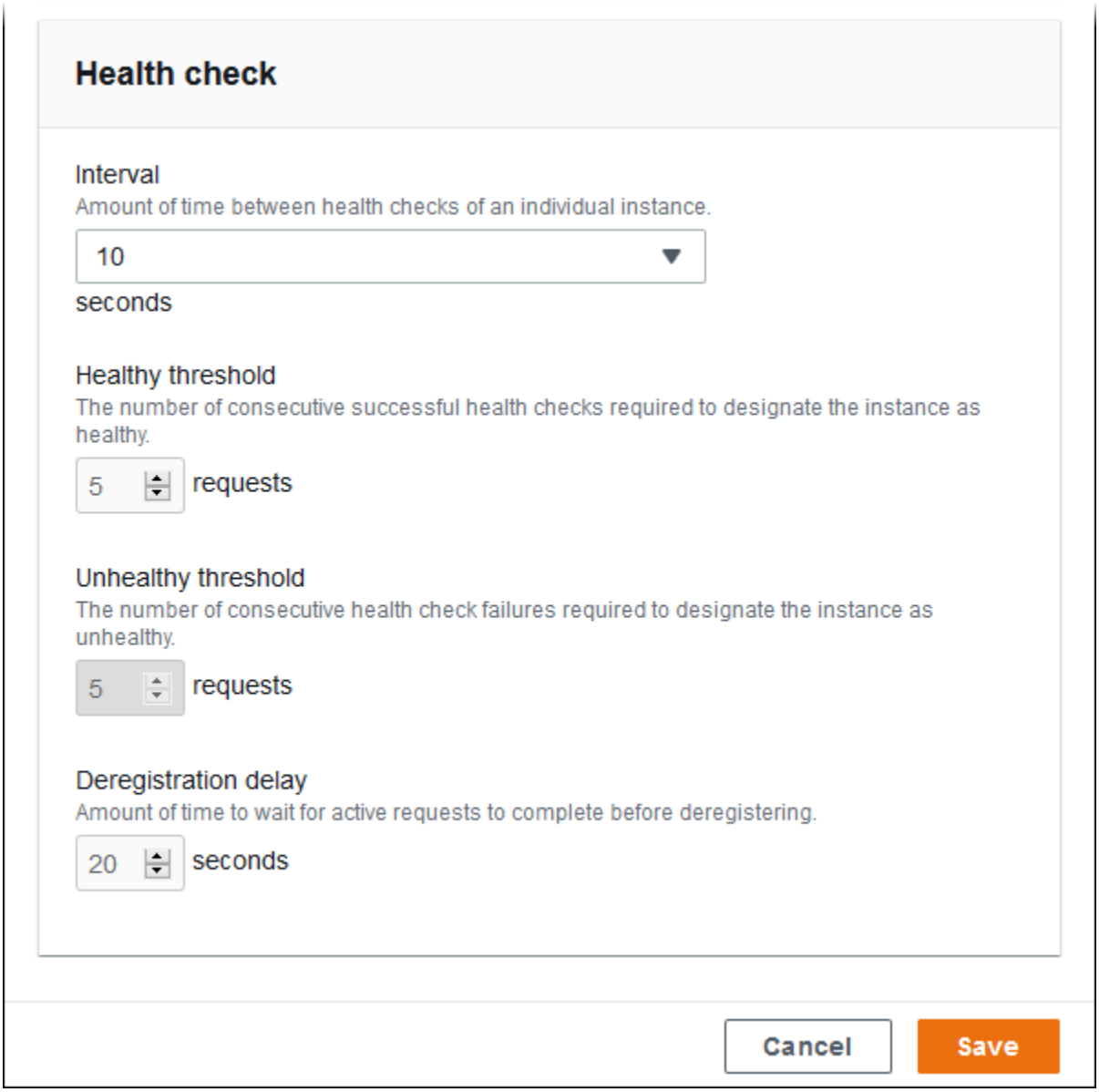
Utilisez ces paramètres pour définir le processus : son nom et le port de processus sur lequel il écoute les requêtes.



Surveillance de l'état

Utilisez les paramètres suivants pour configurer les processus des vérifications de l'état :

- Interval (Intervalle) – Durée, en secondes, entre les vérifications de l'état d'une instance.
- Healthy threshold (Seuil de bonne santé) – Nombre de vérifications de l'état qui doivent réussir avant qu'Elastic Load Balancing modifie l'état de santé d'une instance. (Pour Network Load Balancer, l'option Unhealthy threshold (Seuil de défectuosité) est un paramètre en lecture seule qui est toujours égal à la valeur du seuil de bonne santé.)
- Deregistration delay (Retard d'annulation d'enregistrement) – Délai, en secondes, d'attente de la fin des requêtes actives avant l'annulation de l'enregistrement d'une instance.



> **Note**
>
> La vérification de l'état Elastic Load Balancing n'a pas d'incidence sur le comportement de vérification de l'état du groupe Auto Scaling d'un environnement. Les instances qui échouent à un test de santé d'Elastic Load Balancing ne seront pas automatiquement remplacées par Amazon EC2 Auto Scaling, sauf si vous configurez manuellement Amazon EC2 Auto Scaling pour le faire. Consultez Paramètre de vérification de l'état Auto Scaling pour votre environnement Elastic Beanstalk pour plus de détails.

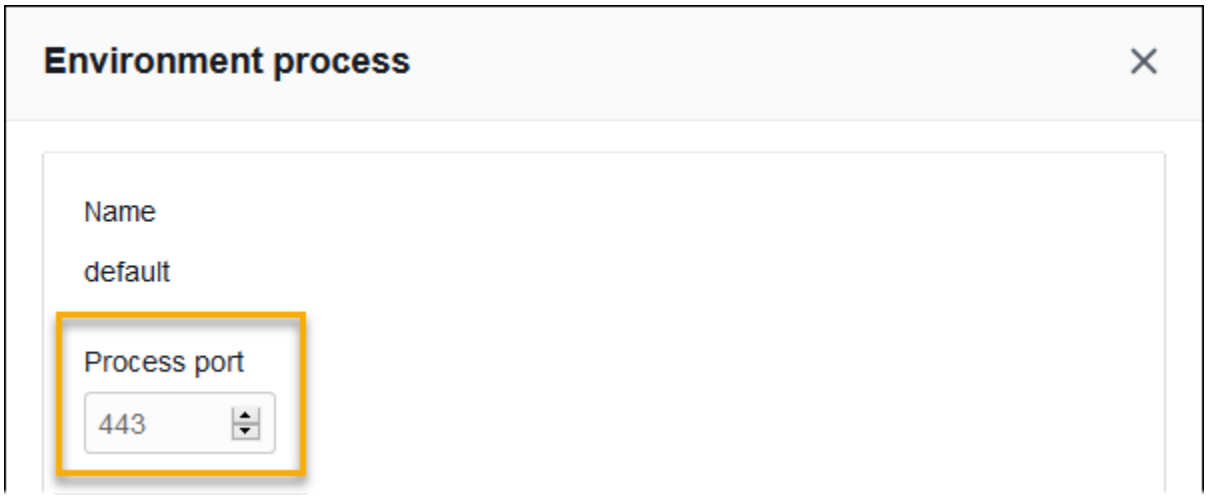
Pour plus d'informations sur les vérifications de l'état et leur influence sur l'état global de votre environnement, consultez Création de rapports d'intégrité de base.

Exemple : Network Load Balancer pour un environnement avec chiffrement end-to-end

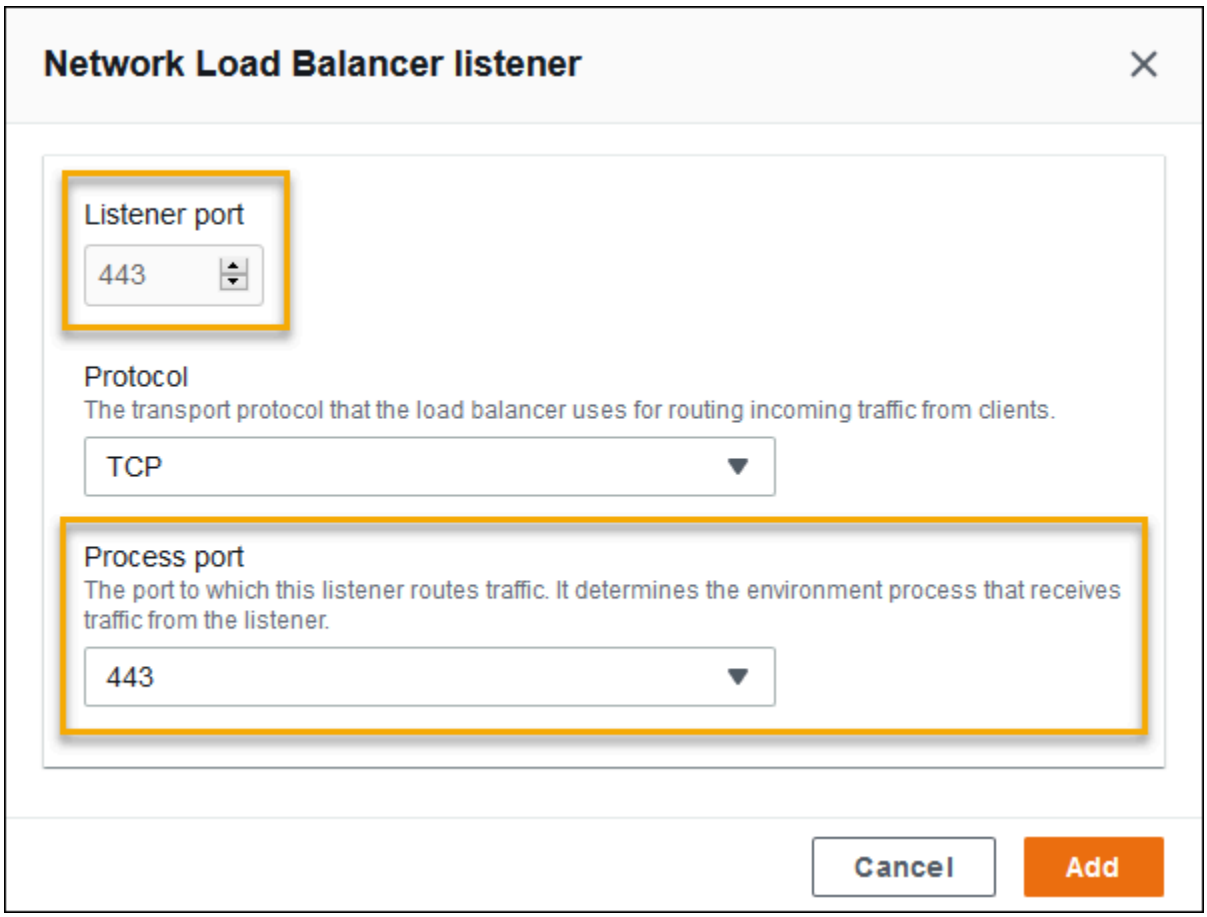
Dans cet exemple, votre application nécessite le chiffrement end-to-end du trafic. Pour configurer Network Load Balancer de votre environnement afin de répondre à ces exigences, configurez le processus par défaut pour écouter sur le port 443, ajoutez un écouteur sur le port 443 qui achemine le trafic vers le processus par défaut et désactivez l'écouteur par défaut.

Pour configurer l'équilibreur de charge de cet exemple

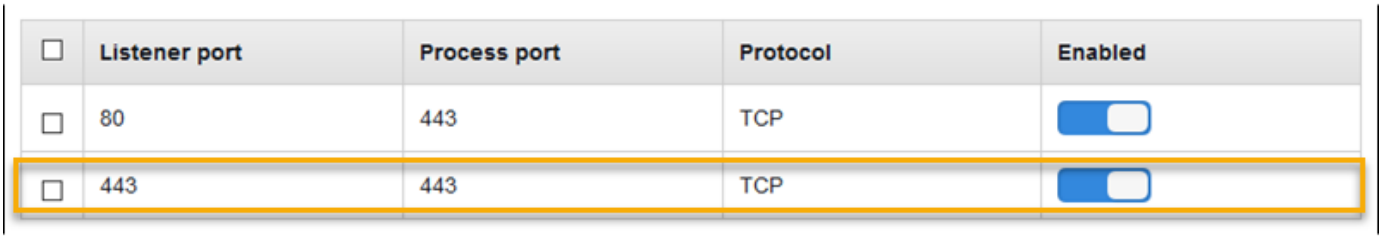
1. Configurez le processus par défaut. Sélectionnez le processus par défaut, puis, pour Actions, choisissez Edit (Modifier). Pour Process port (Port de processus), tapez `443`.



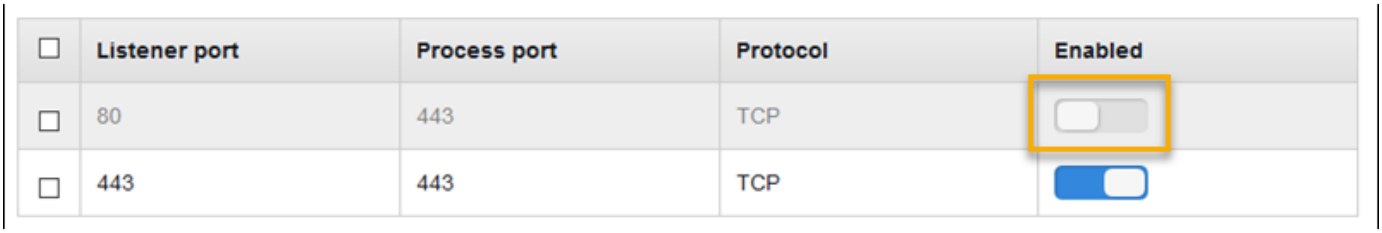
2. Ajoutez un écouteur de port 443. Ajoutez un écouteur. Pour Port d'écoute, tapez `443`. Pour Process port (Port de processus), assurez-vous que `443` est sélectionné.



Vous voyez désormais votre écouteur supplémentaires dans la liste.



3. Désactivez l'écouteur du port 80 par défaut. Pour l'écouteur par défaut, désactivez l'option Activé.



Configuration d'un Network Load Balancer à l'aide de l'interface de ligne de commande EB

L'interface de ligne de commande EB vous invite à choisir un type d'équilibreur de charge lorsque vous exécutez la commande eb create.

```
$ eb create
Enter Environment Name
(default is my-app): test-env
Enter DNS CNAME prefix
(default is my-app): test-env-DLW24ED23SF

Select a load balancer type
1) classic
2) application
3) network
(default is 1): 3
```

Vous pouvez également spécifier un type d'équilibreur de charge à l'aide de l'option `--elb-type`.

```
$ eb create test-env --elb-type network
```

Espaces de noms du Network Load Balancer

Vous trouverez les paramètres liés aux équilibreurs Network Load Balancer dans les espaces de noms suivants :

- `aws:elasticbeanstalk:environment` – Choisissez le type d'équilibreur de charge pour votre environnement. La valeur d'un équilibreur de charge Network Load Balancer est `network`.
- `aws:elbv2:listener` – Configurez les écouteurs sur l'équilibreur de charge Network Load Balancer. Ces paramètres sont mappés aux paramètres de `aws:elb:listener` pour les Classic Load Balancers.
- `aws:elasticbeanstalk:environment:process` – Configurez les vérifications de l'état et spécifiez le port et le protocole des processus qui s'exécutent sur les instances de votre environnement. Les paramètres de port et de protocole sont mappés sur ceux de l'instance dans `aws:elb:listener` pour un écouteur sur un Classic Load Balancer. Les paramètres de vérification de l'état sont mappés sur les paramètres des espaces de noms `aws:elb:healthcheck` et `aws:elasticbeanstalk:application`.

Example extensions .eb/ .config network-load-balancer

Pour vous familiariser avec un équilibreur de charge Network Load Balancer utilisez un fichier de configuration pour définir le type d'équilibreur de charge sur `network`.

```
option_settings:
  aws:elasticbeanstalk:environment:
    LoadBalancerType: network
```

> **Note**
>
> Le type d'équilibreur de charge peut uniquement être défini lors de la création de l'environnement.

Example extensions .eb/ .config nlb-default-process

Le fichier de configuration suivant modifie les paramètres de vérification de l'état sur le processus par défaut.

```
option_settings:
  aws:elasticbeanstalk:environment:process:default:
    DeregistrationDelay: '20'
    HealthCheckInterval: '10'
    HealthyThresholdCount: '5'
    UnhealthyThresholdCount: '5'
    Port: '80'
    Protocol: TCP
```

Example extensions .eb/ .config nlb-secure-listener

Le fichier de configuration suivant ajoute un écouteur pour sécuriser le trafic sur le port 443 et un processus cible correspondant qui écoute le port 443.

```
option_settings:
  aws:elbv2:listener:443:
    DefaultProcess: https
    ListenerEnabled: 'true'
  aws:elasticbeanstalk:environment:process:https:
    Port: '443'
```

L'option `DefaultProcess` est nommée ainsi en raison des Application Load Balancer, qui peuvent avoir des écouteurs autres que ceux par défaut sur le même port pour le trafic vers des chemins spécifiques (consultez Application Load Balancer pour plus de détails). Pour un équilibreur de charge Network Load Balancer, l'option spécifie le seul processus cible de cet écouteur.

Dans cet exemple, nous avons nommé le processus `https`, car il écoute le trafic sécurisé (HTTPS). L'écouteur envoie le trafic vers le processus sur le port désigné à l'aide du protocole TCP, car un équilibreur de charge Network Load Balancer fonctionne uniquement avec TCP. C'est tout à fait normal, car le trafic réseau pour HTTP et HTTPS est mis en œuvre par-dessus TCP.

Configuration des équilibreurs de charge Elastic Beanstalk à double pile

Vous pouvez permettre à vos environnements Elastic Beanstalk de IPv4 servir à la IPv6 fois des protocoles et des protocoles avec des équilibreurs de charge configurés à double pile. Lorsque vous créez un environnement d'équilibrage de charge, l'infrastructure est définie par défaut sur IPv4. Vous pouvez choisir de créer de nouveaux environnements avec une configuration à double pile ou de mettre à jour les environnements existants réservés IPv4 à une configuration à double pile.

Pour permettre aux équilibreurs de charge de votre environnement de gérer à la fois le trafic IPv4 réseau IPv6 et le trafic réseau, définissez l'*`IpAddressType`*option dualstack dans l'aws:elbv2:loadbalancerespace de noms.

> **Note**
>
> Elastic Beanstalk prend uniquement en charge les protocoles à double pile pour les équilibreurs de charge d'application et les équilibreurs de charge réseau. Il ne prend pas en charge le double stack pour les environnements utilisant des équilibreurs de charge classiques ou des environnements à instance unique.

Conditions préalables à Amazon VPC

Avant de configurer votre équilibreur de charge dans votre environnement Elastic Beanstalk, vous devez d'abord effectuer certaines étapes de configuration avec Amazon VPC :

1. Associez un bloc IPv6 CIDR au VPC de votre environnement.
2. Associez des blocs IPv6 CIDR à tous les sous-réseaux du VPC.
3. (Facultatif) Si votre environnement échange du trafic réseau avec des composants extérieurs au VPC, il dispose de tables de routage qui spécifient les réseaux avec lesquels votre VPC peut

communiquer. Dans ce cas, vous devez mettre à jour les tables de routage VPC pour activer IPv6 le trafic.

Vous pouvez effectuer ces configurations prérequises à l'aide de la console Amazon VPC ou AWS CLI de commandes. Les rubriques suivantes vous guideront et vous dirigeront vers le guide de l'utilisateur Amazon VPC et la référence de AWS CLI commande pour plus de détails.

> **Note**
>
> Une fois que vous avez terminé la configuration du VPC, attendez quelques minutes que les modifications se propagent avant de configurer Dual-Stack pour votre équilibreur de charge. Si vous rencontrez des erreurs de configuration de VPC ou de sous-réseau lors de la configuration à double pile, attendez quelques minutes que la configuration VPC se propage et réessayez la configuration à double pile.

Complétez les prérequis du VPC à l'aide de la console

Le guide de l'utilisateur Amazon VPC fournit des étapes détaillées pour effectuer ces tâches préalables.

1. Consultez l'étape 1 : associer un bloc IPv6 CIDR à votre VPC et à vos sous-réseaux dans le guide de l'utilisateur Amazon VPC.

   Cette étape propose deux procédures que vous devez effectuer :

   - Associez un bloc IPv6 CIDR à votre VPC.
   - Associez un bloc IPv6 CIDR à vos sous-réseaux VPC.
2. (Facultatif) Si votre environnement échange du trafic réseau avec des composants extérieurs au VPC, il dispose de tables de routage qui spécifient les réseaux avec lesquels votre VPC peut communiquer. Dans ce cas, vous devez mettre à jour les tables de routage VPC pour activer IPv6 le trafic. Pour terminer cette configuration, consultez l'étape 2 : Mettre à jour vos tables de routage dans le guide de l'utilisateur Amazon VPC.

Complétez les prérequis du VPC à l'aide du AWS CLI

Vous pouvez utiliser le AWS CLI pour terminer et vérifier les configurations prérequises.

1. Associez un bloc IPv6 CIDR au VPC de votre environnement.

a. Utilisez la associate-vpc-cidr-blockcommande pour associer un bloc CIDR à votre VPC.

b. Utilisez la commande describe-vpcs pour vérifier vos configurations VPC.

Example commands

```
# Associate an Amazon-provided IPv6 CIDR block with your VPC
aws ec2 associate-vpc-cidr-block \
    --vpc-id vpc-12345678 \
    --region us-east-1  \
    --amazon-provided-ipv6-cidr-block

# Verify the IPv6 CIDR block association
aws ec2 describe-vpcs \
    --vpc-ids vpc-12345678 \
    --region us-east-1  \
    --query 'Vpcs[0].Ipv6CidrBlockAssociationSet'
```

2. Associez des blocs IPv6 CIDR à tous les sous-réseaux du VPC.

a. Utilisez la associate-subnet-cidr-blockcommande pour associer un bloc CIDR à votre sous-réseau.

b. Utilisez la commande describe-subnets pour vérifier les configurations de vos sous-réseaux.

Example commands

```
# List all subnets in your VPC
aws ec2 describe-subnets \
    --region us-east-1  \
    --filters "Name=vpc-id,Values=vpc-12345678" \
    --query 'Subnets[].{SubnetId:SubnetId,AvailabilityZone:AvailabilityZone}'

# Associate IPv6 CIDR block with each subnet
aws ec2 associate-subnet-cidr-block \
    --subnet-id subnet-12345678 \
    --region us-east-1  \
    --ipv6-cidr-block 2001:db8::/64

# Verify IPv6 CIDR block association for all subnets
aws ec2 describe-subnets \
```

```
    --region us-east-1  \
    --filters "Name=vpc-id,Values=vpc-12345678" \
    --query 'Subnets[].
{SubnetId:SubnetId,Ipv6CidrBlock:Ipv6CidrBlockAssociationSet[0].Ipv6CidrBlock}'
```

3. (Facultatif) Si votre environnement échange du trafic réseau avec des composants extérieurs au VPC, il dispose de tables de routage qui spécifient les réseaux avec lesquels votre VPC peut communiquer. Dans ce cas, vous devez mettre à jour les tables de routage VPC pour activer IPv6 le trafic.

   a. Utilisez la commande create-route pour ajouter une route dans une table de routage au sein du VPC.

   b. Utilisez la describe-route-tablescommande pour vérifier vos tables de routage.

   Example commands

```
# Add IPv6 route to Internet Gateway for public subnets
aws ec2 create-route \
    --route-table-id rtb-12345678 \
    --destination-ipv6-cidr-block ::/0 \
    --gateway-id igw-12345678  \
    --region us-east-1

# Add IPv6 route to NAT Gateway for private subnets (if applicable)
aws ec2 create-route \
    --route-table-id rtb-87654321 \
    --destination-ipv6-cidr-block ::/0 \
    --nat-gateway-id nat-12345678 \
    --region us-east-1

# Verify routes
aws ec2 describe-route-tables \
    --route-table-ids rtb-12345678 \
    --region us-east-1 \
    --query 'RouteTables[0].Routes'
```

Configuration de la double pile pour votre équilibreur de charge Elastic Beanstalk

Une fois la configuration préalable de votre VPC configurée pour votre environnement, vous pouvez configurer l'équilibreur de charge avec l'option Dual-Stack, afin qu'il puisse servir à la

fois aux protocoles et. IPv4 IPv6 Vous pouvez utiliser la console Elastic Beanstalk AWS CLI, les `.ebextensions` fichiers de configuration qu'elle contient AWS et le SDK pour configurer l'équilibreur de charge afin de gérer le trafic à double pile.

Utilisation de la console

Vous pouvez utiliser la console Elastic Beanstalk pour configurer le dual stack pour l'équilibreur de charge de votre environnement.

> **Note**
>
> Cette configuration dépend de la durée de propagation des données en plusieurs points. Tenez compte des exigences temporelles suivantes lorsque vous configurez votre équilibreur de charge et testez les modifications.
> Heure de configuration du VPC. Après avoir terminé la configuration VPC décrite dansConditions préalables à Amazon VPC, attendez quelques minutes que les modifications se propagent avant de suivre ces procédures. Si vous rencontrez des erreurs de configuration de VPC ou de sous-réseau lors de la configuration à double pile, attendez quelques minutes et recommencez les étapes de configuration.
> Durée de propagation du DNS. Après avoir défini l'`IpAddressType`option sur dualstack, attendez quelques minutes que les modifications se propagent avant de procéder au test. La propagation du DNS Route 53 peut prendre 1 à 2 minutes. Pendant ce temps, vous pouvez rencontrer des problèmes lorsque vous testez la communication entre un client et votre application si vous lancez des demandes à l'aide du IPv6 protocole.

Création d'un nouvel environnement : pour configurer votre équilibreur de charge pour la prise en charge du double stack

1. Lancez la console Elastic Beanstalk et commencez les étapes de création d'un nouvel environnement. Après avoir défini les champs obligatoires Rôle de service et profil d'instance EC2 sur la page Configurer l'accès au service, suivez les étapes de cette procédure pour configurer votre équilibreur de charge sur une configuration à double pile. Pour plus d'informations sur la manière de démarrer, consultezCréation d'un environnement Elastic Beanstalk.
2. Dans Configurer l'accès au service, sélectionnez Suivant.
3. La page Configurer le réseau, la base de données et les balises s'affiche.

Si vous avez effectué les Conditions préalables à Amazon VPC étapes décrites dans la section précédente, vous avez déjà configuré le VPC et les sous-réseaux requis. Dans ce cas, ignorez cette étape ainsi que ses sous-étapes pour passer à la sélection du VPC.

a. Pour configurer le VPC et les sous-réseaux, vous pouvez sélectionner Create VPC pour accéder à la console VPC. Suivez les étapes de Complétez les prérequis du VPC à l'aide de la console.

b. Attendez quelques minutes pour que les mises à jour du VPC se propagent, puis revenez à la console Elastic Beanstalk et sélectionnez Actualiser pour passer à l'étape suivante.

   Si vous rencontrez des erreurs de configuration de VPC ou de sous-réseau au cours des étapes restantes, attendez quelques minutes pour laisser le temps à la configuration du VPC de se propager et recommencez les étapes.

4. Sur la page Configurer le réseau, la base de données et les balises, sélectionnez une valeur dans la liste déroulante VPC associée à IPv6 un bloc CIDR.

   Une fois que vous avez sélectionné un VPC, les sous-réseaux d'instance seront remplis avec les sous-réseaux VPC.

5. Sélectionnez un ou plusieurs sous-réseaux d'instance, puis sélectionnez Next.

6. La page Configurer le trafic et le dimensionnement de l'instance s'affiche.

   Dans les paramètres réseau de l'équilibreur de charge, sélectionnez Activer pour Dualstack (&) IPv4 . IPv6

7. Sélectionnez le type d'équilibreur de charge. L'équilibreur de charge d'application ou l'équilibreur de charge réseau prennent tous deux en charge le DualStack.

8. Vous pouvez continuer à configurer les autres options de l'équilibreur de charge sur la page de console actuelle. Pour plus d'informations sur les options et la configuration de l'équilibreur de charge, consultez les autres rubriques de ce chapitre.

9. Poursuivez les étapes pour terminer la configuration et la création de votre environnement. Pour de plus amples informations, veuillez consulter Création d'un environnement Elastic Beanstalk.

Environnement existant : pour configurer votre équilibreur de charge pour la prise en charge du double stack

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie Configuration du réseau et de la base de données, sélectionnez Modifier.
5. Si vous avez effectué les Conditions préalables à Amazon VPC étapes décrites dans la section précédente, vous avez déjà configuré le VPC et les sous-réseaux requis. Dans ce cas, ignorez cette étape ainsi que ses sous-étapes pour passer à la sélection du VPC.

   a. Pour configurer le VPC et les sous-réseaux, vous pouvez sélectionner Create VPC pour accéder à la console VPC. Suivez les étapes de Complétez les prérequis du VPC à l'aide de la console.

   b. Attendez quelques minutes pour que les mises à jour du VPC se propagent, puis revenez à la console Elastic Beanstalk et sélectionnez Actualiser pour passer à l'étape suivante.

      Si vous rencontrez des erreurs de configuration de VPC ou de sous-réseau au cours des étapes restantes, attendez quelques minutes pour laisser le temps à la configuration du VPC de se propager et recommencez les étapes.
6. Sur la page Réseau et base de données, sélectionnez une valeur dans la liste déroulante VPC associée à IPv6 un bloc CIDR.

   Une fois que vous avez sélectionné un VPC, les sous-réseaux d'instance seront remplis avec les sous-réseaux VPC.
7. Sélectionnez un ou plusieurs sous-réseaux d'instance.
8. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.
9. Dans la catégorie de configuration Instances (Instances), choisissez Edit (Modifier).
10. Dans les paramètres réseau de l'équilibreur de charge, sélectionnez Activer pour Dualstack (&) IPv4 . IPv6
11. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

À l'aide du AWS CLI

Vous pouvez utiliser l'interface de ligne de AWS commande (AWS CLI) pour configurer les équilibreurs de charge de votre environnement afin de gérer à la fois le trafic IPv6 et le trafic IPv4 réseau. Cette section fournit des exemples de commandes create-environment et update-environment avec l'espace de noms. aws:elbv2:loadbalancer

> **Note**
>
> Cette configuration dépend de la durée de propagation des données en plusieurs points. Tenez compte des exigences temporelles suivantes lorsque vous configurez votre équilibreur de charge et testez les modifications.
>
> Heure de configuration du VPC. Une fois que vous avez terminé la configuration VPC décrite dansConditions préalables à Amazon VPC, attendez quelques minutes que les modifications se propagent avant d'exécuter ces commandes pour configurer votre équilibreur de charge pour une double pile. Si vous rencontrez des erreurs de configuration de VPC ou de sous-réseau lors de la configuration à double pile, attendez quelques minutes et réessayez les commandes.
>
> Durée de propagation du DNS. Après avoir défini l'`IpAddressType`option sur dualstack, attendez quelques minutes que les modifications se propagent avant de procéder au test. La propagation du DNS Route 53 peut prendre 1 à 2 minutes. Pendant ce temps, vous pouvez rencontrer des problèmes lorsque vous testez la communication entre un client et votre application si vous lancez des demandes à l'aide du IPv6 protocole.

Example de create-environment avec configuration dualstack (options d'espace de noms en ligne)

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2 v3.4.0 running Python 3.8" \
--option-settings \
Namespace=aws:autoscaling:launchconfiguration,OptionName=IamInstanceProfile,Value=aws-elasticbeanstalk-ec2-role \
Namespace=aws:elbv2:loadbalancer,OptionName=IpAddressType,Value=dualstack
```

À titre de solution de rechange, utilisez un fichier `options.json` pour spécifier les options de l'espace de noms au lieu de les inclure en ligne. L'exemple de commande suivant illustre la commande update-environment.

Example de l'environnement de mise à jour avec configuration dualstack (options d'espace de noms dans le fichier) **`options.json`**

```
aws elasticbeanstalk update-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2 v3.4.0 running Python 3.8" \
--option-settings \ file://options.json
```

Example

```
### example options.json ###
[
  {
    "Namespace": "aws:elbv2:loadbalancer",
    "OptionName": "IpAddressType",
    "Value": "dualstack"
  }
]
```

L'exemple suivant met à jour un environnement existant pour définir l'`IpAddressType`option sur IPv4.

> **Note**
>
> Cet exemple de `update-environment` commande est utile si vous devez restaurer la configuration de votre environnement de Dual-Stack à IPv4.

Example de l'environnement de mise à jour à définir IpAddressType IPv4

```
aws elasticbeanstalk update-environment \
--region us-east-1 \
--application-name my-app \
```

```
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2 v3.4.0 running Python 3.8" \
--option-settings \
Namespace=aws:elbv2:loadbalancer,OptionName=IpAddressType,Value=ipv4
```

Utilisation des fichiers de configuration .ebextensions

Vous pouvez utiliser les fichiers de configuration d'Elastic Beanstalk pour permettre aux équilibreurs de charge de votre environnement de gérer à la fois le trafic et le trafic réseau. IPv6 IPv4 Définissez l'`IpAddressType`option dualstack dans l'aws:elbv2:loadbalancerespace de noms.

> **Note**
>
> Cette configuration dépend de la durée de propagation des données en plusieurs points. Tenez compte des exigences temporelles suivantes lorsque vous configurez votre équilibreur de charge et testez les modifications.
> Heure de configuration du VPC. Après avoir terminé la configuration du VPC décrite dansConditions préalables à Amazon VPC, attendez quelques minutes pour que les modifications du VPC se propagent avant d'appliquer les configurations. `.ebextensions` Si vous rencontrez des erreurs de configuration de VPC ou de sous-réseau au cours de cette configuration, attendez quelques minutes et réessayez.
> Durée de propagation du DNS. Après avoir défini l'`IpAddressType`option sur dualstack, attendez quelques minutes que les modifications se propagent avant de procéder au test. La propagation du DNS Route 53 peut prendre 1 à 2 minutes. Pendant ce temps, vous pouvez rencontrer des problèmes lorsque vous testez la communication entre un client et votre application si vous lancez des demandes à l'aide du IPv6 protocole.

Example.ebextensions/options.config pour la configuration à double pile de l'équilibreur de charge (syntaxe abrégée)

```
option_settings:
  aws:elbv2:loadbalancer:
    IpAddressType: dualstack
```

Example.ebextensions/options.config pour la configuration dualstack de l'équilibreur de charge (syntaxe standard)

```
option_settings:
```

```
  - namespace: aws:elbv2:loadbalancer
    option_name: IpAddressType
    value: dualstack
```

Utilisation du AWS SDK

Vous pouvez configurer Dual-Stack à l'aide du AWS SDKs. Comme pour les `create-environment` AWS CLI commandes `update-environment` et mentionnées dans la section précédente, vous pouvez utiliser les actions de l'UpdateEnvironmentAPI CreateEnvironmentet. Utilisez le paramètre de `OptionSettings` requête pour spécifier les options de l'espace de aws:elbv2:loadbalancer noms.

> **Note**
>
> Cette configuration dépend de la durée de propagation des données en plusieurs points. Tenez compte des exigences temporelles suivantes lorsque vous configurez votre équilibreur de charge et testez les modifications.
> Heure de configuration du VPC. Une fois que vous avez terminé la configuration VPC décrite dansConditions préalables à Amazon VPC, attendez quelques minutes pour que les modifications se propagent avant d'exécuter vos programmes afin de configurer l'équilibreur de charge pour Dual-Stack. Si vous rencontrez des erreurs de configuration de VPC ou de sous-réseau lors de la configuration à double pile, attendez quelques minutes et réessayez d'exécuter les programmes de configuration à double pile.
> Durée de propagation du DNS. Après avoir défini l'`IpAddressType`option sur dualstack, attendez quelques minutes que les modifications se propagent avant de procéder au test. La propagation du DNS Route 53 peut prendre 1 à 2 minutes. Pendant ce temps, vous pouvez rencontrer des problèmes lorsque vous testez la communication entre un client et votre application si vous lancez des demandes à l'aide du IPv6 protocole.

Résolution des problèmes

> **Essayez Amazon Q Developer CLI pour un dépannage assisté par l'IA**
>
> L'interface de ligne de commande Amazon Q Developer peut vous aider à résoudre rapidement les problèmes d'environnement. La Q CLI fournit des solutions en vérifiant l'état de l'environnement, en examinant les événements, en analysant les journaux et en posant des questions de clarification. Pour plus d'informations et des instructions détaillées,

consultez la section Résolution des problèmes liés aux environnements Elastic Beanstalk avec Amazon Q Developer CLI dans les blogs. AWS

Cette section fournit des conseils pour résoudre les problèmes liés à la configuration à double pile de votre équilibreur de charge.

Événement : aucun bloc IPv6 CIDR *vpc_id* n'est configuré pour le VPC. IPv6 Les blocs CIDR sont nécessaires pour l'équilibreur de charge à double pile. Veuillez associer un bloc IPv6 CIDR à votre VPC avant d'utiliser le mode dualstack.

Des blocs IPv6 CIDR doivent être associés à votre VPC et à tous les sous-réseaux. Il s'agit de l'une des conditions préalables au VPC que vous devez remplir avant de configurer votre équilibreur de charge pour la prise en charge du double stack. Pour plus d'informations sur l'exécution de cette tâche, voir Conditions préalables à Amazon VPC plus haut dans cette rubrique.

Événement : aucun bloc CIDR n' IPv6 est configuré pour un ou plusieurs sous-réseaux *vpc_id* VPC. IPv6 Les blocs CIDR sont requis pour les sous-réseaux utilisés avec l'équilibreur de charge à double pile. Associez les blocs IPv6 CIDR à tous les sous-réseaux requis avant d'utiliser le mode dualstack.

Des blocs CIDR doivent être IPv6 associés à tous les sous-réseaux de votre VPC. Il s'agit de l'une des conditions préalables au VPC que vous devez remplir avant de configurer votre équilibreur de charge pour la prise en charge du double stack. Pour plus d'informations sur l'exécution de cette tâche, voir Conditions préalables à Amazon VPC plus haut dans cette rubrique.

Erreur : l'*IpAddressType*option ne peut être appliquée que sur les environnements Elastic Beanstalk configurés avec un Application Load Balancer ou un Network Load Balancer.

Ce message indique que votre environnement Elastic Beanstalk est peut-être un environnement à instance unique ou qu'il utilise un Classic Load Balancer. Seuls les environnements configurés avec un Application Load Balancer ou un Network Load Balancer peuvent être configurés. `IpAddressType`

Configuration des journaux d'accès

Vous pouvez utiliser des fichiers de configuration afin de configurer l'équilibreur de charge de votre environnement pour qu'il charge les journaux d'accès dans un compartiment Amazon S3. Consultez les exemples de fichiers de configuration suivants GitHub pour obtenir des instructions :

- `loadbalancer-accesslogs-existingbucket.config` – Configurez l'équilibreur de charge pour qu'il charge les journaux d'accès dans un compartiment Amazon S3 existant.
- `loadbalancer-accesslogs-newbucket.config` – Configurez l'équilibreur de charge pour qu'il charge les journaux d'accès dans un nouveau compartiment.

Ajout d'une base de données à votre environnement Elastic Beanstalk

Elastic Beanstalk fournit une intégration de base de données couplée à Amazon Relational Database Service (Amazon RDS). Vous pouvez utiliser Elastic Beanstalk pour ajouter une base de données MySQL, PostgreSQL, Oracle ou SQL Server à un environnement existant ou à un nouvel environnement, lorsque vous le créez. Lorsque vous ajoutez une instance de base de données couplée à un environnement, Elastic Beanstalk fournit les informations de connexion à votre application. Pour ce faire, définissez les propriétés d'environnement pour le nom d'hôte de la base de données, le port, le nom d'utilisateur, le mot de passe et le nom de la base de données.

Avantages de l'utilisation d'une base de données couplée

Si vous n'avez encore jamais utilisé d'instance de base de données avec votre application, nous vous recommandons d'abord d'utiliser le processus décrit dans cette rubrique pour ajouter une base de données à un environnement de test à l'aide du service Elastic Beanstalk. Cette opération vous permet de vérifier que votre application peut lire les propriétés de l'environnement, créer une chaîne de connexion et se connecter à une instance de base de données sans le travail de configuration supplémentaire requis pour une base de données externe à Elastic Beanstalk.

Considérations à prendre en compte lors du passage en production

Après avoir vérifié que votre application fonctionne correctement avec la base de données, vous pouvez envisager de passer à un environnement de production. À ce stade, vous avez la possibilité de découpler la base de données de votre environnement Elastic Beanstalk pour passer à une configuration offrant une plus grande flexibilité. La base de données découplée peut demeurer opérationnelle en tant qu'instance de base de données Amazon RDS externe. L'état de l'environnement n'est pas affecté par le découplage de la base de données. Si vous devez résilier l'environnement, vous pouvez le faire et choisir l'option permettant de maintenir la base de données disponible et opérationnelle en dehors d'Elastic Beanstalk.

Avantages du passage à une base de données découplée

L'utilisation d'une base de données externe présente plusieurs avantages. Vous pouvez vous connecter à la base de données externe depuis plusieurs environnements, utiliser des types de base de données qui ne sont pas pris en charge par les bases de données intégrées et effectuer blue/green des déploiements. Au lieu d'utiliser une base de données découplée créée par Elastic Beanstalk, vous pouvez également créer une instance de base de données en dehors de votre environnement Elastic Beanstalk. Quelle que soit l'option privilégiée, l'instance de base de données externe à votre environnement Elastic Beanstalk générée nécessite une configuration supplémentaire pour le groupe de sécurité et la chaîne de connexion. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon RDS.

Sections

- Cycle de vie de base de données
- Ajout d'une instance de base de données Amazon RDS à votre environnement à l'aide de la console
- Connexion à la base de données
- Configuration d'une instance de base de données RDS intégrée à l'aide de la console
- Configuration d'une instance de base de données RDS intégrée à l'aide des fichiers de configuration
- Découplage d'une instance de base de données RDS à l'aide de la console
- Découplage d'une instance de base de données RDS à l'aide de fichiers de configuration

Cycle de vie de base de données

Vous pouvez déterminer que faire de la base de données une fois que vous l'avez découplée de votre environnement Elastic Beanstalk. Les options parmi lesquelles vous pouvez choisir sont collectivement appelées stratégies de suppression. Les stratégies de suppression suivantes s'appliquent à une base de données après que vous l'ayez découplée d'un environnement Elastic Beanstalk ou que vous ayez résilié l'environnement Elastic Beanstalk.

- Instantané : avant qu'Elastic Beanstalk résilie la base de données, il enregistre un instantané de celle-ci. Vous pouvez restaurer une base de données à partir d'un instantané lorsque vous ajoutez une instance de base de données à un environnement Elastic Beanstalk ou lorsque vous créez une base de données autonome. Pour plus d'informations sur la création d'une nouvelle instance de base de données autonome à partir d'un instantané, consultez Restauration à partir d'un instantané de base de données dans le Guide de l'utilisateur Amazon RDS. Le stockage

des instantanés de base de données peut entraîner des frais. Pour de plus amples informations, veuillez consulter la section Stockage de sauvegarde dans Tarification d'Amazon RDS.

- Supprimer : Elastic Beanstalk résilie la base de données. Une fois la base de données résiliée, l'instance de base de données n'est plus disponible pour aucune opération.
- Conserver : l'instance de base de données n'est pas résiliée. Elle reste disponible et opérationnelle, bien qu'elle ait été découplée d'Elastic Beanstalk. Vous pouvez ensuite configurer un ou plusieurs environnements pour connexion à la base de données en tant qu'instance de base de données Amazon RDS externe. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon RDS.

Ajout d'une instance de base de données Amazon RDS à votre environnement à l'aide de la console

Vous pouvez ajouter une instance de base de données à votre environnement à l'aide de la console Elastic Beanstalk.

Pour ajouter une instance DB à votre environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Vous pouvez configurer les options suivantes :

- Instantané – Choisissez un instantané existant de la base de données. Elastic Beanstalk restaure l'instantané et l'ajoute à votre environnement. La valeur par défaut est Aucune. Lorsque la valeur par défaut est None (Aucun), vous pouvez configurer la nouvelle base de données en utilisant les autres paramètres sur cette page.
- Moteur – Choisissez un moteur de base de données.

- Version du moteur – Choisissez une version spécifique du moteur de base de données.
- Classe d'instance – Choisissez la classe d'instance de base de données. Pour plus d'informations sur les classes d'instance de base de données, consultez https://aws.amazon.com/rds/.
- Stockage – Choisissez la quantité de stockage à allouer à votre base de données. Vous pouvez augmenter le stockage alloué ultérieurement, mais vous ne pouvez pas le réduire. Pour de plus amples informations sur l'allocation de stockage, veuillez consulter Fonctionnalités.
- Username (Nom d'utilisateur) : saisissez un nom d'utilisateur de votre choix en utilisant une combinaison de chiffres et de lettres uniquement.
- Mot de passe – Entrez un mot de passe de votre choix contenant 8–16 caractères ASCII imprimables (sauf `/`, `\` et `@`).
- Disponibilité – Choisissez Élevée (Multi-AZ) pour exécuter une sauvegarde à chaud dans une deuxième zone de disponibilité pour une haute disponibilité.
- Database deletion policy (Stratégie de suppression de base de données) : la stratégie de suppression détermine ce qu'il advient de la base de données une fois qu'elle est découplée de votre environnement. Elle peut être définie sur les valeurs suivantes : `Create Snapshot`, `Retain` ou `Delete`. Ces valeurs sont décrites dans Cycle de vie de base de données de cette rubrique.

> **Note**
>
> Elastic Beanstalk crée un utilisateur principal pour la base de données en utilisant le nom d'utilisateur et le mot de passe que vous fournissez. Pour de plus amples informations sur l'utilisateur principal et ses privilèges, veuillez consulter Privilèges du compte utilisateur principal.

L'ajout d'une instance de base de données prend environ 10 minutes. Lorsque la mise à jour est terminée, la nouvelle base de données est couplée à votre environnement. Le nom d'hôte de l'instance de base de données et les autres informations de connexion sont disponibles dans votre application via les propriétés d'environnement suivantes.

Nom de la propriété	Description	Valeur de la propriété
`RDS_HOSTNAME`	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console

Nom de la propriété	Description	Valeur de la propriété
		Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, **ebdb**.	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

Connexion à la base de données

Utilisez les informations de connectivité pour vous connecter à votre base de données à partir de votre application, par le biais des variables d'environnement. Pour plus d'informations sur l'utilisation d'Amazon RDS avec vos applications, consultez les rubriques suivantes.

- Java SE – Connexion à une base de données (plateformes Java SE)
- Java avec Tomcat – Connexion à une base de données (plateformes Tomcat)
- Node.js – Connexion à une base de données
- .NET – Connexion à une base de données
- PHP – Connexion à une base de données à l'aide d'un PDO ou d'un My SQLi
- Python – Connexion à une base de données

- Ruby – Connexion à une base de données

Configuration d'une instance de base de données RDS intégrée à l'aide de la console

Vous pouvez visualiser et modifier les paramètres de configuration de votre instance de base de données dans la section Database (Base de données) de la page Configuration de l'environnement, dans la console Elastic Beanstalk.

Pour configurer l'instance de base de données de votre environnement dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).

Vous pouvez modifier les paramètres Instance class (Classe d'instance), Storage (Stockage), Password (Mot de passe), Availability (Disponibilité) et Database deletion policy (Stratégie de suppression de base de données) après la création de la base de données. Si vous modifiez la classe d'instance, Elastic Beanstalk reprovisionne l'instance de base de données.

Si vous n'avez plus besoin d'Elastic Beanstalk pour associer la base de données à l'environnement, vous pouvez décider de la découpler en sélectionnant Decouple database (Découpler la base de données). Il importe de comprendre les options et les considérations liées à cette opération. Pour de plus amples informations, veuillez consulter the section called “Découplage d'une instance de base de données RDS à l'aide de la console”.

> **Avertissement**
>
> Ne modifiez pas les paramètres sur l'instance de base de données couplée en dehors de la fonctionnalité fournie par Elastic Beanstalk (par exemple, dans la console Amazon RDS). Si vous le faites, votre configuration de base de données Amazon RDS risque de ne pas être synchronisée avec la définition de votre environnement. Lorsque vous mettez à jour ou

redémarrez votre environnement, les paramètres spécifiés dans l'environnement remplacent ceux que vous avez définis en dehors d'Elastic Beanstalk.
Si vous devez modifier les paramètres que Elastic Beanstalk ne prend pas en charge directement, utilisez les fichiers de configuration Elastic Beanstalk.

Configuration d'une instance de base de données RDS intégrée à l'aide des fichiers de configuration

Vous pouvez configurer une instance de base de données de votre environnement à l'aide de fichiers de configuration. Utilisez les options de l'espace de noms `aws:rds:dbinstance`. L'exemple suivant remplace la taille de stockage de base de données allouée par 100 Go.

Example extensions .eb/ .config db-instance-options

```
option_settings:
  aws:rds:dbinstance:
    DBAllocatedStorage: 100
```

Si vous souhaitez configurer les propriétés d'instance de base de données qui ne sont pas prises en charge par Elastic Beanstalk, vous pouvez utiliser un fichier de configuration et spécifier vos paramètres à l'aide de la clé `resources`. L'exemple suivant définit les valeurs sur les propriétés Amazon RDS `StorageType` et `Iops`.

Example extensions .eb/ .config db-instance-properties

```
Resources:
  AWSEBRDSDatabase:
    Type: AWS::RDS::DBInstance
    Properties:
      StorageType:io1
      Iops: 1000
```

Découplage d'une instance de base de données RDS à l'aide de la console

Vous pouvez découpler votre base de données d'un environnement Elastic Beanstalk sans affecter l'état de l'environnement. Avant de découpler la base de données, tenez compte des exigences suivantes :

- Que doit-il advenir de la base de données une fois qu'elle est découplée ?

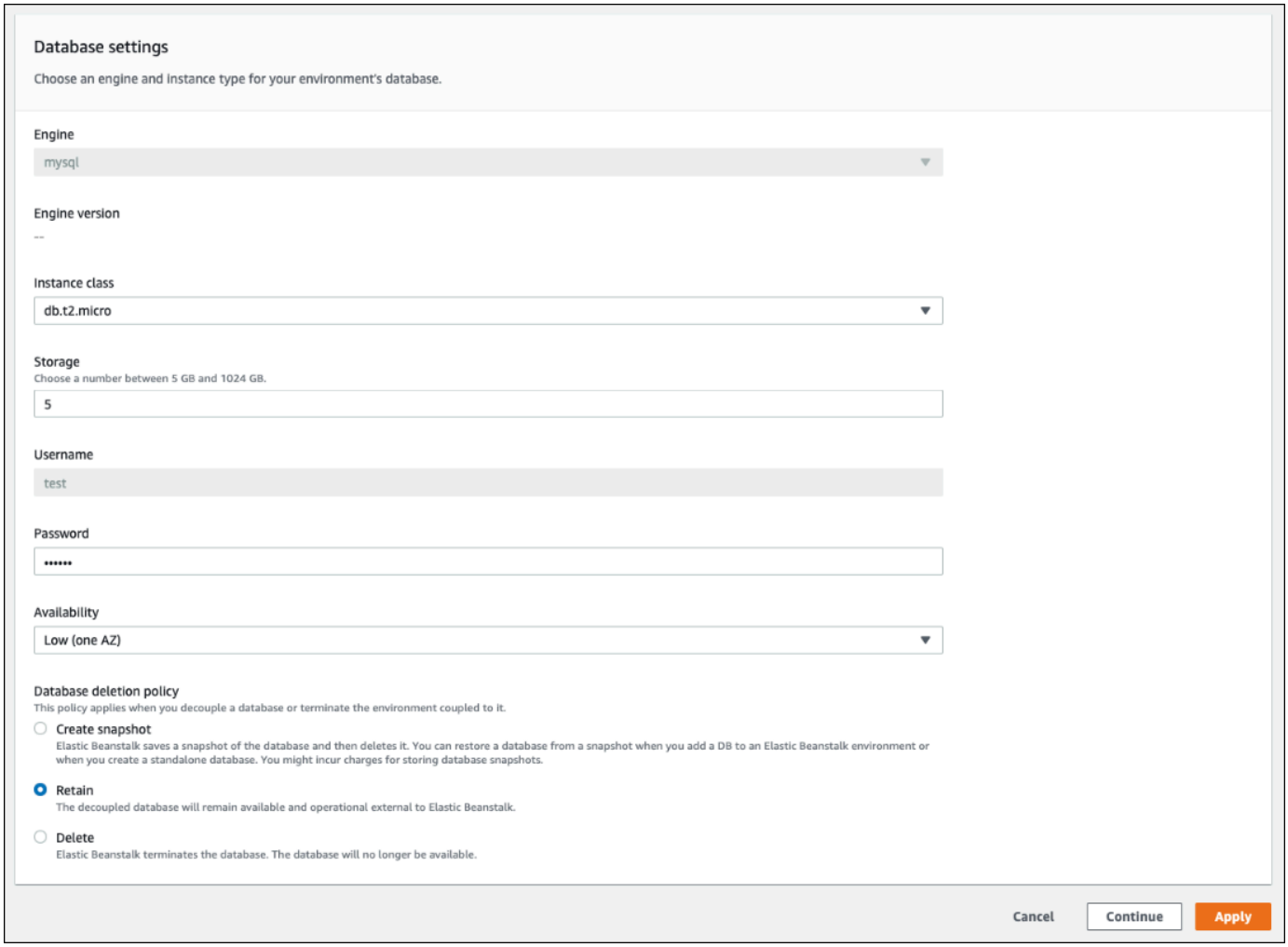
  Vous pouvez choisir de créer un instantané de la base de données, puis de la résilier, de conserver la base de données opérationnelle en tant que base de données autonome externe à Elastic Beanstalk, ou de supprimer définitivement la base de données. Le paramètre Database deletion policy (Stratégie de suppression de base de données) détermine ce résultat. Pour bénéficier d'une description détaillée des stratégies de suppression, consultez Cycle de vie de base de données dans cette rubrique.

- Avez-vous besoin d'apporter des modifications aux paramètres de configuration de la base de données avant de la découpler ?

  Si vous devez apporter des modifications de configuration à la base de données, vous devez les appliquer avant le découplage de la base de données. Cela inclut les modifications apportées au paramètre Database deletion policy (Stratégie de suppression de base de données). Toute modification en attente envoyée simultanément avec le paramètre Decouple database (Découpler la base de données) est ignorée, car seul le paramètre de découplage est appliqué.

Pour découpler une instance de base de données d'un environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Passez en revue toutes les valeurs de configuration dans la section Database settings (Paramètres de la base de données), en particulier la valeur Database deletion policy (Stratégie de suppression de base de données), qui détermine ce qu'il advient de la base de données une fois qu'elle est découplée.



Si tous les autres paramètres de configuration sont corrects, passez à l'étape 6 pour découpler la base de données.

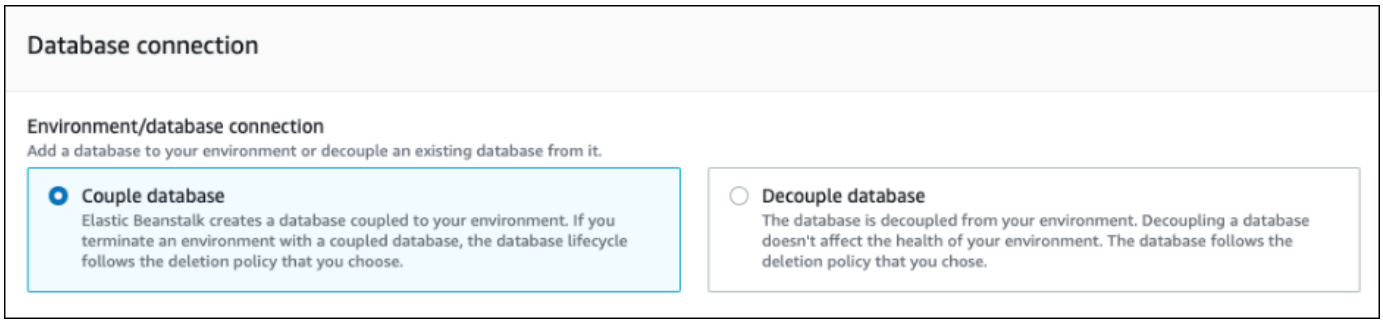
> **Warning**
>
> Il est important d'appliquer le paramètre Database deletion policy (Stratégie de suppression de base de données) séparément du paramètre Decouple database (Découpler la base de données). Si vous sélectionnez Apply (Appliquer) dans le but d'enregistrer à la fois Decouple database (Découpler la base de données) et une nouvelle Database deletion policy (Stratégie de suppression de base de données), la nouvelle stratégie de suppression que vous avez choisie est ignorée. Elastic Beanstalk découple alors la base de données en suivant la stratégie de suppression définie précédemment. Si la stratégie de suppression antérieure est `Delete` ou `Create`

`Snapshot`, vous risquez de perdre la base de données au lieu de suivre la stratégie en attente prévue.

Si l'un des paramètres de configuration nécessite des mises à jour, procédez comme suit :

1. Apportez les modifications requises dans le panneau Database settings (Paramètres de la base de données).
2. Cliquez sur Appliquer. Quelques minutes sont nécessaires à l'enregistrement des modifications de configuration de votre base de données.
3. Revenez à l'étape 3 et sélectionnez Configuration dans le panneau de navigation.

6. Accédez à la section Database connection (Connexion de la base de données) du panneau.



7. Sélectionnez Decouple database (Découpler la base de données).
8. Sélectionnez Apply (Appliquer) pour lancer l'opération de découplage de la base de données.

Le paramètre de stratégie de suppression détermine le résultat de la base de données et la durée nécessaire pour découpler la base de données.

- Si la stratégie de suppression est définie sur `Delete`, la base de données est supprimée. L'opération peut prendre environ 10 à 20 minutes selon la taille de la base de données.
- Si la stratégie de suppression est définie sur `Snapshot`, un instantané de la base de données est créé. Ensuite, la base de données est supprimée. La durée requise pour ce processus varie en fonction de la taille de la base de données.
- Si la stratégie de suppression est définie sur `Retain`, la base de données reste opérationnelle en dehors de l'environnement Elastic Beanstalk. Généralement, moins de cinq minutes sont nécessaires pour découpler une base de données.

Si vous décidez de conserver la base de données de manière externe à votre environnement Elastic Beanstalk, vous devrez prendre des mesures supplémentaires afin de la configurer. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon RDS. Si vous envisagez d'utiliser la base de données que vous découplez pour un environnement de production, vérifiez que le type de stockage utilisé par la base de données est adapté à votre charge de travail. Pour de plus amples informations, consultez Stockage d'instance de base de données et Modification d'une instance de base de données dans le Guide de l'utilisateur Amazon RDS.

Découplage d'une instance de base de données RDS à l'aide de fichiers de configuration

Vous pouvez découpler votre instance de base de données d'un environnement Elastic Beanstalk sans affecter l'état de l'environnement. L'instance de base de données suit la stratégie de suppression de base de données appliquée lorsque la base de données a été découplée.

Les deux options requises pour découpler la base de données se trouvent dans l'espace de noms the section called “aws:rds:dbinstance”. Ce sont les suivants :

- L'option `DBDeletionPolicy` définit la stratégie de suppression. Elle peut être définie sur les valeurs suivantes : `Snapshot`, `Delete` ou `Retain`. Ces valeurs sont décrites dans Cycle de vie de base de données de cette rubrique.
- L'option `HasCoupledDatabase` détermine si votre environnement possède une base de données couplée.
  - Si elle bascule sur `true`, Elastic Beanstalk crée une nouvelle instance de base de données couplée à votre environnement.
  - Si elle bascule sur `false`, Elastic Beanstalk commence à découpler l'instance de base de données de votre environnement.

Si vous souhaitez modifier la configuration de votre base de données avant de la découpler, appliquez d'abord, au cours d'une opération distincte, toutes les modifications de configuration. Cela comprend la modification de la configuration `DBDeletionPolicy`. Une fois vos modifications appliquées, exécutez une commande distincte pour définir l'option de découplage. Si, en plus du paramètre de découplage, vous soumettez d'autres paramètres de configuration, les autres paramètres d'option de configuration sont ignorés lorsque le paramètre de découplage est appliqué.

> **Warning**
>
> Il est important que vous exécutiez les commandes pour appliquer les paramètres `DBDeletionPolicy` et `HasCoupledDatabase` en deux opérations distinctes. Si la stratégie de suppression active est déjà définie sur `Delete` ou `Snapshot`, vous risquez de perdre la base de données. La base de données suit la stratégie de suppression actuellement active, plutôt que la stratégie de suppression en attente souhaitée.

Pour découpler une instance de base de données d'un environnement

Suivez ces étapes pour découpler la base de données de votre environnement Elastic Beanstalk. Vous pouvez utiliser l'interface de ligne de commande EB ou le AWS CLI pour effectuer les étapes. Pour de plus amples informations, consultez Personnalisation avancée de l'environnement avec des fichiers de configuration.

1. Si vous souhaitez modifier la stratégie de suppression, configurez un fichier de configuration au format suivant. Dans cet exemple, la stratégie de suppression est définie de manière à être conservée.

   Example

   ```
   option_settings:
     aws:rds:dbinstance:
       DBDeletionPolicy: Retain
   ```

2. Exécutez la commande à l'aide de votre outil préféré pour terminer la mise à jour de configuration.
3. Configurez un fichier de configuration afin de définir `HasCoupledDatabase` sur `false`.

   Example

   ```
   option_settings:
     aws:rds:dbinstance:
       HasCoupledDatabase: false
   ```

4. Exécutez la commande à l'aide de votre outil préféré pour terminer la mise à jour de configuration.

Le paramètre de stratégie de suppression détermine le résultat de la base de données et la durée nécessaire pour découpler la base de données.

- Si la stratégie de suppression est définie sur `Delete`, la base de données est supprimée. L'opération peut prendre environ 10 à 20 minutes selon la taille de la base de données.
- Si la stratégie de suppression est définie sur `Snapshot`, un instantané de la base de données est créé. Ensuite, la base de données est supprimée. La durée requise pour ce processus varie en fonction de la taille de la base de données.
- Si la stratégie de suppression est définie sur `Retain`, la base de données reste opérationnelle en dehors de l'environnement Elastic Beanstalk. Généralement, moins de cinq minutes sont nécessaires pour découpler une base de données.

Si vous décidez de conserver la base de données de manière externe à votre environnement Elastic Beanstalk, vous devrez prendre des mesures supplémentaires afin de la configurer. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon RDS. Si vous envisagez d'utiliser la base de données que vous découplez pour un environnement de production, vérifiez que le type de stockage utilisé par la base de données est adapté à votre charge de travail. Pour de plus amples informations, consultez Stockage d'instance de base de données et Modification d'une instance de base de données dans le Guide de l'utilisateur Amazon RDS.

La sécurité de votre AWS Elastic Beanstalk environnement

Elastic Beanstalk propose plusieurs options qui contrôlent l'accès aux services (sécurité) de votre environnement et des instances Amazon qu'il contient. EC2 Cette rubrique explique comment configurer ces options.

Sections

- Configuration de la sécurité de votre environnement
- Espaces de noms de configuration de la sécurité de l'environnement

Configuration de la sécurité de votre environnement

Vous pouvez modifier la configuration de sécurité de votre environnement Elastic Beanstalk dans la console Elastic Beanstalk.

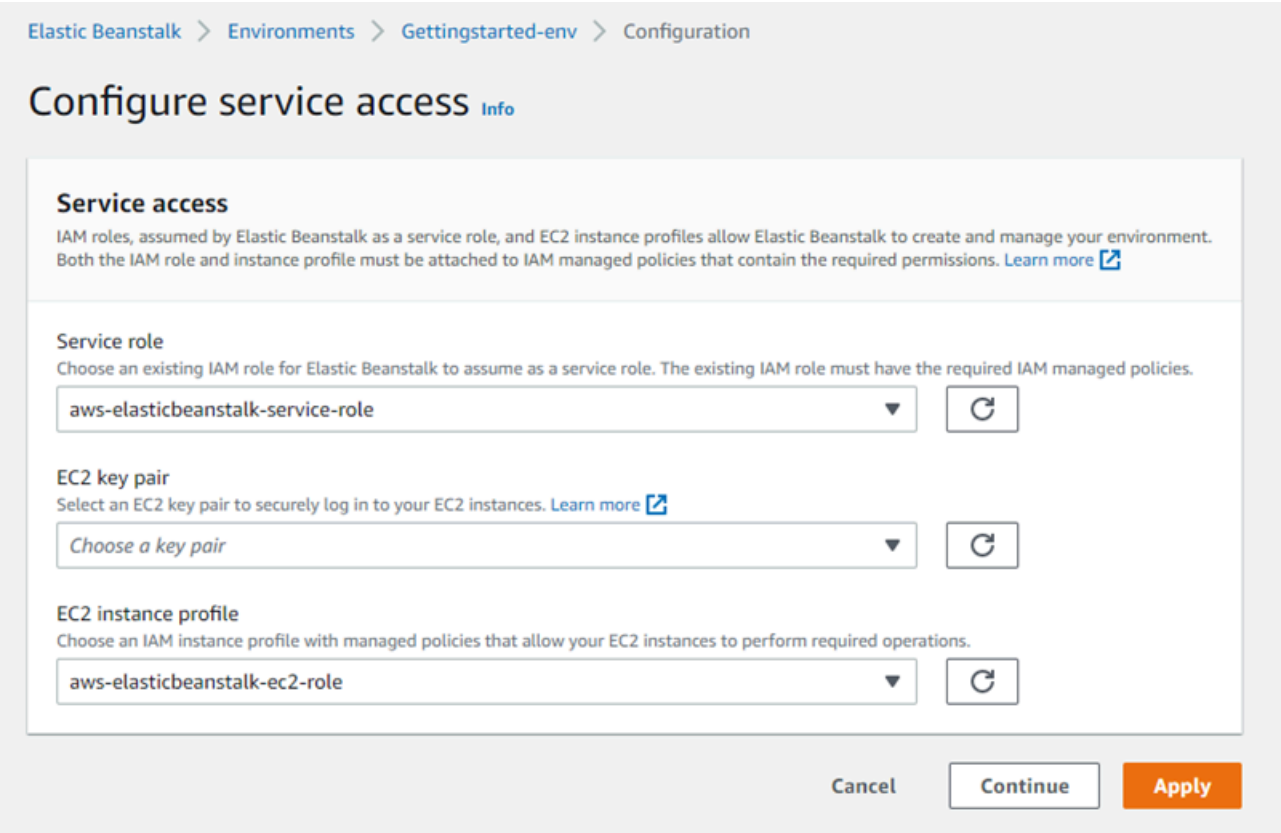
Pour configurer la (sécurité) de l'environnement dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Service access (Accès au service), choisissez Edit (Modifier).

Les paramètres suivants sont disponibles.

Settings

- Rôle de service
- EC2 paire de clés
- Profil d'instance IAM



Rôle de service

Sélectionnez un rôle de service à associer à votre environnement Elastic Beanstalk. Elastic Beanstalk assume le rôle de service lorsqu'il AWS accède à d'autres services en votre nom. Pour en savoir plus, consultez Gestion des rôles de service Elastic Beanstalk.

EC2 paire de clés

Vous pouvez vous connecter en toute sécurité aux instances Amazon Elastic Compute Cloud (Amazon EC2) fournies pour votre application Elastic Beanstalk à l'aide d'une paire de clés Amazon. EC2 Pour obtenir des instructions sur la création d'une paire de clés, consultez la section Création d'une paire de clés à l'aide d'Amazon EC2 dans le guide de EC2 l'utilisateur Amazon.

> **Note**
>
> Lorsque vous créez une paire de clés, Amazon EC2 stocke une copie de votre clé publique. Si vous n'avez plus besoin de l'utiliser pour vous connecter à des instances d'environnement, vous pouvez le supprimer d'Amazon EC2. Pour en savoir plus, consultez Supprimer votre paire de clés dans le guide de EC2 l'utilisateur Amazon.

Choisissez une paire de EC2 clés dans le menu déroulant pour l'attribuer aux instances de votre environnement. Lorsque vous assignez une paire de clés, la clé publique est stockée sur l'instance pour authentifier la clé privée, que vous stockez localement. La clé privée n'est jamais stockée sur AWS.

Pour plus d'informations sur la connexion aux EC2 instances Amazon, consultez Connect to your instance et Connecting to Linux/UNIX Instances from Windows using PuTTY dans le guide de EC2 l'utilisateur Amazon.

Profil d'instance IAM

Un profil d' EC2 instance est un rôle IAM appliqué aux instances lancées dans votre environnement Elastic Beanstalk. Les EC2 instances Amazon assument le rôle de profil d'instance pour signer les demandes AWS et y accéder APIs, par exemple pour télécharger des journaux sur Amazon S3.

La première fois que vous créez un environnement dans la console Elastic Beanstalk, Elastic Beanstalk vous invite à créer un profil d'instance avec un ensemble d'autorisations par défaut. Vous

pouvez ajouter des autorisations à ce profil pour permettre à vos instances d'accéder à d'autres AWS services. Pour en savoir plus, consultez Gestion des profils d'instance Elastic Beanstalk.

> **Note**
>
> Elastic Beanstalk avait précédemment créé EC2 un `aws-elasticbeanstalk-ec2-role` profil d'instance par défaut nommé la première AWS fois qu'un compte créait un environnement. Ce profil d'instance incluait des stratégies gérées par défaut. Si votre compte possède déjà ce profil d'instance, vous pourrez toujours l'attribuer à vos environnements. Cependant, les directives AWS de sécurité récentes n'autorisent pas un AWS service à créer automatiquement des rôles avec des politiques de confiance vis-à-vis d'autres AWS services, EC2 dans ce cas. En raison de ces directives de sécurité, Elastic Beanstalk ne crée plus de profil d'instance `aws-elasticbeanstalk-ec2-role` par défaut.

> **Note**
>
> Un autre aspect de la sécurité des EC2 instances consiste à définir des règles de pare-feu pour les EC2 instances. Ceci est contrôlé par des groupes EC2 de sécurité. Pour de plus amples informations, veuillez consulter Les EC2 instances Amazon pour votre environnement Elastic Beanstalk.

Espaces de noms de configuration de la sécurité de l'environnement

Elastic Beanstalk propose des options de configuration dans les espaces de noms suivants, vous permettant de personnaliser la sécurité de votre environnement :

- `aws:elasticbeanstalk:environment` – Configurez le rôle de service de l'environnement à l'aide de l'option `ServiceRole`.
- `aws:autoscaling:launchconfiguration`— Configurez les autorisations pour les EC2 instances Amazon de l'environnement à l'aide des `SecurityGroups` options `EC2KeyName` `IamInstanceProfile` `DisableDefaultEC2SecurityGroup`,, et.

L'interface de ligne de commande (CLI) EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous

voulez utiliser des fichiers de configuration pour configurer la même chose. Consultez Valeurs recommandées pour plus de détails.

Balisage des ressources dans vos environnements Elastic Beanstalk

Vous pouvez appliquer des balises à vos AWS Elastic Beanstalk environnements. Les balises sont des paires clé-valeur associées à des AWS ressources. Pour plus d'informations sur l'étiquette des ressources Elastic Beanstalk, les cas d'utilisation, les contraintes de clé et de valeur d'identification, et les types de ressources pris en charge, consultez Étiquette des ressources d'application Elastic Beanstalk.

Elastic Beanstalk applique des balises d'environnement à la ressource environnementale elle-même, ainsi qu'aux autres AWS ressources créées par Elastic Beanstalk pour l'environnement. Vous pouvez utiliser des balises pour gérer les autorisations au niveau de la ressource spécifique d'un environnement. Pour plus d'informations, consultez la section Marquage de vos EC2 ressources Amazon dans le guide de l' EC2 utilisateur Amazon.

Par défaut, Elastic Beanstalk applique quelques balises à votre environnement :

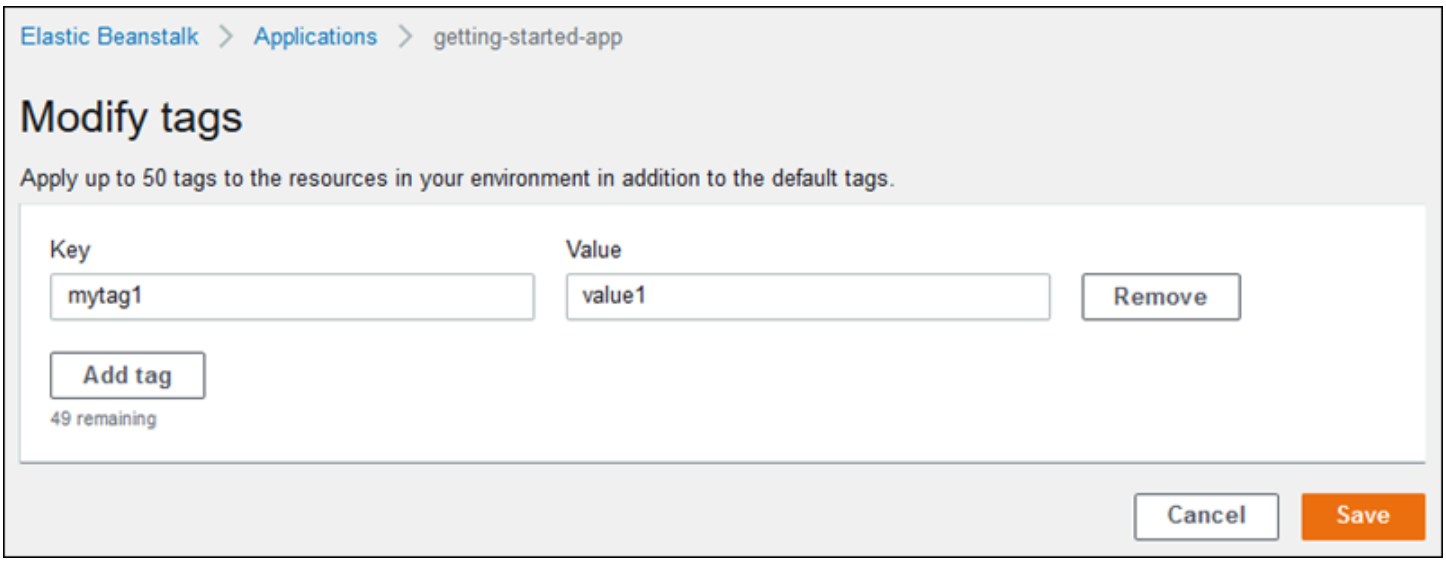
- `elasticbeanstalk:environment-name` : Nom de l'environnement.
- `elasticbeanstalk:environment-id` : ID de l'environnement.
- `Name`— Également le nom de l'environnement. `Name`est utilisé dans le tableau de EC2 bord Amazon pour identifier et trier les ressources.

Vous ne pouvez pas modifier ces balises par défaut.

Vous pouvez spécifier des balises lorsque vous créez l'environnement Elastic Beanstalk. Dans un environnement existant, vous pouvez ajouter ou supprimer des balises, ainsi que mettre à jour les valeurs des balises existantes. Un environnement peut avoir jusqu'à 50 balises, y compris les balises par défaut.

Ajout de balises lors de la création de l'environnement

Lorsque vous utilisez la console Elastic Beanstalk pour créer un environnement, vous pouvez spécifier des clés et valeurs de balise sur la page de configuration Modify tags (Modifier les balises) de l'assistant Create New Environment (Créer un nouvel environnement).



Si vous utilisez l'interface de ligne de commande EB pour créer des environnements, utilisez l'option `--tags` avec eb create pour ajouter des balises.

```
~/workspace/my-app$ eb create --tags mytag1=value1,mytag2=value2
```

Avec le AWS CLI ou d'autres clients basés sur une API, utilisez le `--tags` paramètre de la commande. create-environment

```
$ aws elasticbeanstalk create-environment \
      --tags Key=mytag1,Value=value1 Key=mytag2,Value=value2 \
      --application-name my-app --environment-name my-env --cname-prefix my-app --
version-label v1 --template-name my-saved-config
```

Les configurations enregistrées incluent des balises définies par l'utilisateur. Lorsque vous appliquez une configuration enregistrée contenant des balises lors de la création de l'environnement, ces balises s'appliquent au nouvel environnement tant que vous ne spécifiez pas de nouvelles balises. Si vous ajoutez des balises à un environnement via l'une des méthodes précédentes, les balises définies dans la configuration enregistrée sont ignorées.

Gestion des balises d'un environnement existant

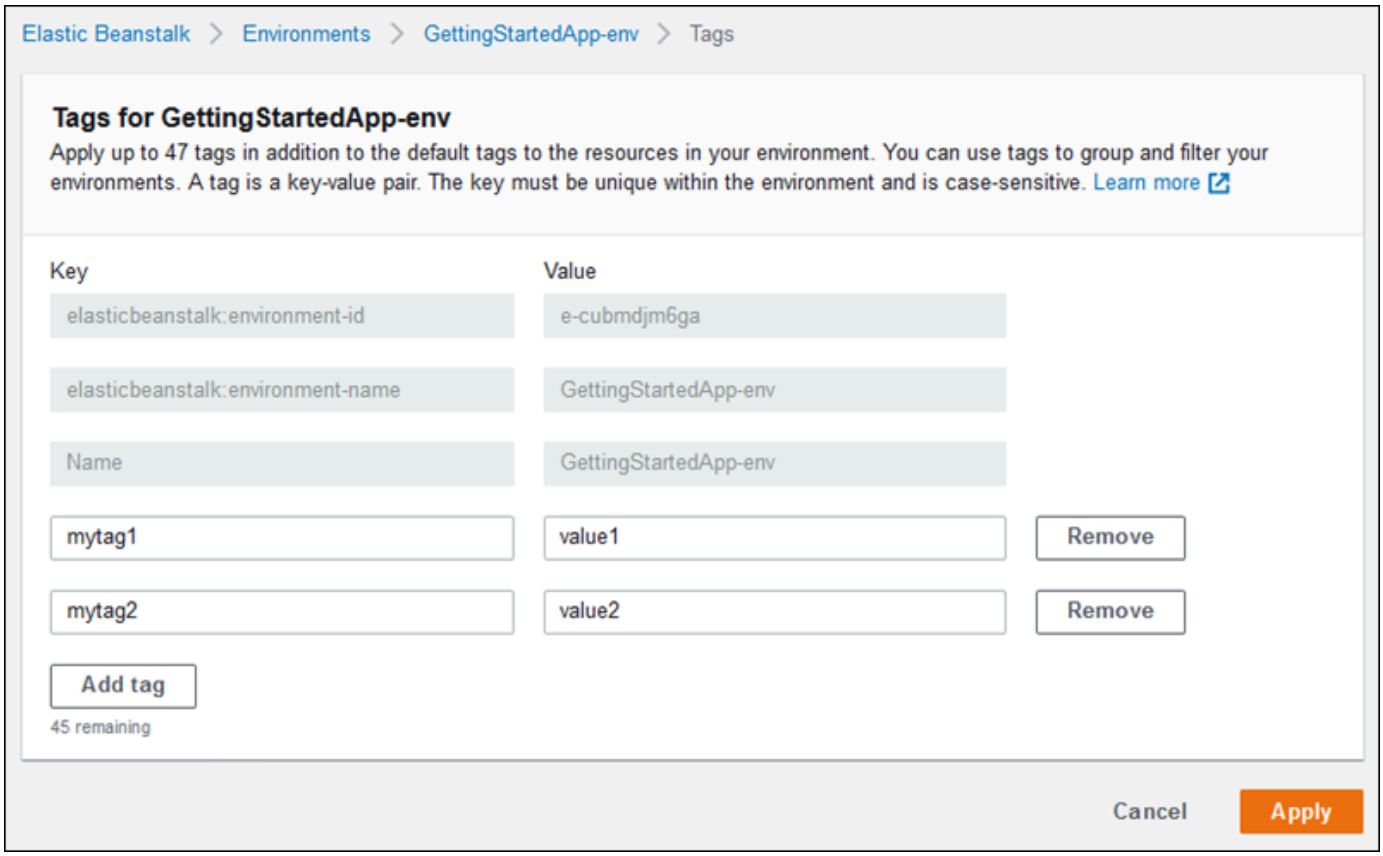
Vous pouvez ajouter, mettre à jour et supprimer des balises dans un environnement Elastic Beanstalk existant. Elastic Beanstalk applique les modifications aux ressources de votre environnement.

Vous ne pouvez cependant pas modifier les balises par défaut qu'Elastic Beanstalk applique à votre environnement.

Pour gérer les balises d'un environnement dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, sélectionnez Tags.

   La page de gestion des balises affiche la liste des balises qui existent actuellement dans l'environnement.



4. Ajoutez, modifiez ou supprimez les balises :

   - Pour ajouter une balise, tapez-la dans les zones vides en bas de la liste. Pour ajouter une autre balise, choisissez Add tag (Ajouter une balise) et Elastic Beanstalk ajoute une autre paire de cases vides.
   - Pour mettre à jour la valeur ou la clé d'une balise, modifiez la zone correspondante dans la ligne de la balise.

- Pour supprimer une balise, choisissez Remove (Retirer) en regard de la zone de valeur de la balise.

5. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Si vous utilisez l'interface de ligne de commande EB pour mettre à jour les environnements, utilisez eb tags pour ajouter, mettre à jour, supprimer ou répertorier des balises.

Par exemple, la commande suivante répertorie les balises de votre environnement par défaut.

```
~/workspace/my-app$ eb tags --list
```

La commande suivante met à jour l'identification `mytag1` et supprime l'identification `mytag2`.

```
~/workspace/my-app$ eb tags --update mytag1=newvalue --delete mytag2
```

Pour obtenir une liste complète des options et d'autres exemples, consultez `eb tags`.

Avec le AWS CLI ou d'autres clients basés sur une API, utilisez la list-tags-for-resource commande pour répertorier les balises d'un environnement.

```
$ aws elasticbeanstalk list-tags-for-resource --resource-arn
 "arn:aws:elasticbeanstalk:us-east-2:my-account-id:environment/my-app/my-env"
```

Utilisez la commande update-tags-for-resource pour ajouter, mettre à jour ou supprimer des balises dans un environnement.

```
$ aws elasticbeanstalk update-tags-for-resource \
      --tags-to-add Key=mytag1,Value=newvalue --tags-to-remove mytag2 \
      --resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:environment/my-
app/my-env"
```

Spécifiez les identifications à ajouter et les identifications à mettre à jour dans le paramètre `--tags-to-add` de update-tags-for-resource. Une identification inexistante est ajoutée et la valeur d'une identification existante est mise à jour.

> **Note**
>
> Pour utiliser ces deux AWS CLI commandes dans un environnement Elastic Beanstalk, vous avez besoin de l'ARN de l'environnement. Vous pouvez extraire l'ARN à l'aide de la commande suivante.
>
> ```
> $ aws elasticbeanstalk describe-environments
> ```

Variables d'environnement et autres paramètres du logiciel

La page de configuration des mises à jour, de la surveillance et de la journalisation vous permet de configurer le logiciel sur les instances Amazon Elastic Compute Cloud (Amazon EC2) qui exécutent votre application. Vous pouvez configurer les variables d'environnement, le AWS X-Ray débogage, le stockage et le streaming des journaux d'instance, ainsi que les paramètres spécifiques à la plate-forme.

Rubriques

- Configurer les paramètres spécifiques à la plateforme
- Configuration des propriétés de l'environnement (variables d'environnement)
- Espaces de noms des paramètres de logiciel
- Accès aux propriétés de l'environnement
- Configuration du AWS X-Ray débogage
- Affichage de vos journaux d'environnement Elastic Beanstalk

Configurer les paramètres spécifiques à la plateforme

Outre l'ensemble d'options standard disponibles pour tous les environnements, la plupart des plateformes Elastic Beanstalk vous permettent de spécifier des paramètres propres à une langue ou à l'infrastructure. Ils apparaissent dans la section Logiciel de plateforme de la page Configurer les mises à jour, la surveillance et la journalisation et peuvent prendre les formes suivantes.

- Propriétés de l'environnement prédéfinies – La plateforme Ruby utilise des propriétés d'environnement pour les paramètres d'infrastructure comme `RACK_ENV` et `BUNDLE_WITHOUT`.

- Propriétés de l'environnement d'espace réservé – La plateforme Tomcat définit une propriété d'environnement nommée `JDBC_CONNECTION_STRING` qui n'a aucune valeur définie. Ce type de paramètre était plus fréquent sur les anciennes versions de la plateforme.
- Options de configuration – La plupart des plateformes définissent des options de configuration dans des espaces de noms propres à la plateforme ou partagés tels que `aws:elasticbeanstalk:xray` ou `aws:elasticbeanstalk:container:python`.

Pour configurer les paramètres spécifiques à la plateforme dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Sous Logiciel de la plateforme, apportez les modifications nécessaires aux paramètres des options.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Pour plus d'informations sur les options propres aux plateformes, et sur l'obtention des valeurs de propriété d'environnement dans votre code, reportez-vous à la rubrique relative à la plateforme pour votre langue ou votre infrastructure :

- Docker – the section called “Configuration de l'environnement”
- Go – Utilisation de la plateforme Go Elastic Beanstalk
- Java SE – Utilisation de la plateforme Java SE Elastic Beanstalk
- Tomcat – Utilisation de la plateforme Elastic Beanstalk Tomcat
- .NET Core sous Linux – Utilisation du noyau d'Elastic Beanstalk .NET sur la plateforme Linux
- .NET – Utilisation de la plateforme Windows Elastic Beanstalk .NET
- Node.js – Utilisation de la plateforme Elastic Beanstalk Node.js
- PHP – Utilisation de la plateforme PHP Elastic Beanstalk
- Python – Utilisation de la plateforme Python Elastic Beanstalk

- Ruby – Utilisation de la plateforme Elastic Beanstalk Ruby

Configuration des propriétés de l'environnement (variables d'environnement)

Vous pouvez utiliser les propriétés d'environnement (également appelées variables d'environnement) pour transmettre des points de terminaison, des paramètres de débogage et d'autres informations à votre application. Les variables d'environnement vous aident à exécuter votre application dans plusieurs environnements à des fins différentes, telles que le développement, les tests, la préparation et la production.

En outre, lorsque vous ajoutez une base de données à votre environnement, Elastic Beanstalk définit des variables d'environnement, `RDS_HOSTNAME` telles que celles que vous pouvez lire dans le code de votre application pour créer un objet ou une chaîne de connexion.

Pour configurer les variables d'environnement dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Faites défiler la page jusqu'à Variables d'environnement d'exécution.
6. Sélectionnez Ajouter une variable d'environnement.
7. Pour Source, sélectionnez Texte brut.

> **Note**
>
> Les valeurs Secrets Manager et SSM Parameter Store de la liste déroulante permettent de configurer les variables d'environnement en tant que secrets pour stocker des données sensibles, telles que les informations d'identification et les clés d'API. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic AWS Secrets Manager AWS Systems Manager Beanstalk avec et Parameter Store.

8. Entrez le nom de la variable d'environnement et les paires de valeurs de la variable d'environnement.
9. Si vous devez ajouter d'autres variables, répétez les étapes 6 à 8.
10. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Limites des propriétés de l'environnement

- Les clés peuvent contenir des caractères alphanumériques et les symboles suivants : `_ . : / + \ - @`

  Les symboles répertoriés sont valides pour les clés de propriété d'environnement, mais peuvent ne pas être valides pour les noms de variable d'environnement sur la plateforme de votre environnement. Pour assurer une compatibilité avec toutes les plateformes, limitez les propriétés d'environnement au schéma suivant : `[A-Z_][A-Z0-9_]*`
- Les valeurs peuvent contenir des caractères alphanumériques, des espaces et les symboles suivants : `_ . : / = + \ - @ ' "`

  > **Note**
  >
  > Certains caractères dans les valeurs de propriété d'environnement doivent être échappés. Utilisez le caractère barre oblique inverse (\) pour représenter certains caractères spéciaux et de contrôle. La liste suivante inclut des exemples de représentation de certains caractères devant être échappés :
  >
  > - barre oblique inverse (\) – pour représenter l'utilisation `\\`
  > - guillemet simple (') – pour représenter l'utilisation `\'`
  > - guillemet double (") – pour représenter l'utilisation `\"`
- Les clés et les valeurs sont sensibles à la casse.
- La taille combinée de toutes les propriétés d'environnement ne peut pas dépasser 4 096 octets lorsqu'elles sont stockées sous forme de chaînes au format *key* =. *value*

Espaces de noms des paramètres de logiciel

Vous pouvez utiliser un fichier de configuration pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration

peuvent être spécifiques à la plate-forme ou s'appliquer à toutes les plateformes du service Elastic Beanstalk dans son ensemble. Les options de configuration sont organisées en espaces de noms.

Vous pouvez utiliser des fichiers de configuration Elastic Beanstalk pour définir les propriétés de l'environnement et les options de configuration dans votre code source. Utilisez l'espace de noms `aws:elasticbeanstalk:application:environment` pour définir les propriétés de l'environnement.

Example.ebextensions/options.config

```
option_settings:
  aws:elasticbeanstalk:application:environment:
    API_ENDPOINT: www.example.com/api
```

Si vous utilisez des fichiers de configuration ou des CloudFormation modèles pour créer des ressources personnalisées, vous pouvez utiliser une CloudFormation fonction pour obtenir des informations sur la ressource et l'attribuer dynamiquement à une propriété d'environnement lors du déploiement. L'exemple suivant, tiré du elastic-beanstalk-samples GitHub référentiel, utilise la fonction Ref pour obtenir l'ARN d'une rubrique Amazon SNS créée et l'affecte à une propriété d'environnement nommée. `NOTIFICATION_TOPIC`

> **Remarques**
>
> - Si vous utilisez une CloudFormation fonction pour définir une propriété d'environnement, la console Elastic Beanstalk affiche la valeur de la propriété avant que la fonction ne soit évaluée. Vous pouvez utiliser le script de plateforme `get-config` pour confirmer les valeurs des propriétés d'environnement disponibles pour votre application.
> - La plateforme Docker Multicontainer n'est pas utilisée CloudFormation pour créer des ressources de conteneur. Par conséquent, cette plate-forme ne prend pas en charge la définition des propriétés de l'environnement à l'aide de CloudFormation fonctions.

Example.Ebextensions/sns-topic.config

```
Resources:
  NotificationTopic:
    Type: AWS::SNS::Topic

option_settings:
```

```
  aws:elasticbeanstalk:application:environment:
    NOTIFICATION_TOPIC: ``{"Ref" : "NotificationTopic"}``
```

Vous pouvez également utiliser cette fonctionnalité pour propager les informations à partir des pseudo-paramètres CloudFormation. Cet exemple obtient la région actuelle et l'affecte à une propriété nommée `AWS_REGION`.

Example.Ebextensions/env-regionname.config

```
option_settings:
  aws:elasticbeanstalk:application:environment:
    AWS_REGION: ``{"Ref" : "AWS::Region"}``
```

La plupart des plateformes Elastic Beanstalk définissent des espaces de noms supplémentaires avec des options pour configurer le logiciel qui s'exécute sur l'instance, tel que le proxy inversé qui relaie des requêtes pour votre application. Pour plus d'informations sur les espaces de noms disponibles pour votre plateforme, consultez les ressources suivantes :

- Go – Espaces de noms de la configuration Go
- Java SE – Espaces de noms de la configuration Java SE
- Tomcat – Espaces de noms de la configuration Tomcat
- .NET Core sous Linux – Espace de noms de la configuration .NET Core sous Linux
- .NET – Espace de noms aws:elasticbeanstalk:container:dotnet:apppool
- Node.js – Espaces de noms de la configuration Node.js
- PHP – Espaces de noms pour la configuration
- Python – Espaces de noms de la configuration Python
- Ruby – Espaces de noms de configuration Ruby

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d' AWS CLI. Pour plus d’informations, consultez Options de configuration.

Accès aux propriétés de l'environnement

Dans la plupart des cas, vous accédez aux propriétés de l'environnement dans le code de votre application sous forme d'une variable d'environnement. Toutefois, les propriétés d'environnement

sont généralement transmises uniquement à l'application et ne peuvent pas être affichées via une connexion à une instance dans votre environnement et l'exécution de `env`.

- Go – `os.Getenv`

```
endpoint := os.Getenv("API_ENDPOINT")
```

- Java SE – `System.getenv`

```
String endpoint = System.getenv("API_ENDPOINT");
```

- Tomcat — et `System.getProperty` `System.getenv`

  Les versions de la plateforme Tomcat publiées le 26 mars 2025 ou après cette date peuvent également être utilisées `System.getenv` pour accéder aux variables d'environnement en texte brut. Vous pouvez continuer à utiliser pour accéder `System.getProperty` aux variables d'environnement en texte brut. Cependant, les variables d'environnement stockées sous forme de secrets ne sont disponibles qu'à l'aide de`System.getenv`.

```
String endpoint = System.getProperty("API_ENDPOINT");
```

```
String endpoint = System.getenv("API_ENDPOINT");
```

  > **Important**
  >
  > L'ajout de l'`System.getenv`accès aux variables d'environnement dans les versions de la plateforme Tomcat publiées le 26 mars 2025 ou après cette date peut provoquer un comportement inattendu dans les applications qui accordent la priorité aux variables d'environnement sur les propriétés du système Java ou lors du passage explicite de à`System.getProperty`. `System.getenv` Pour plus d'informations et les actions recommandées, consultez la section Utilisation de la plateforme Elastic Beanstalk Tomcat.

- .NET Core sous Linux – `Environment.GetEnvironmentVariable`

```
string endpoint = Environment.GetEnvironmentVariable("API_ENDPOINT");
```

- .NET – `appConfig`

```
NameValueCollection appConfig = ConfigurationManager.AppSettings;
```

```
string endpoint = appConfig["API_ENDPOINT"];
```

- Node.js : `process.env`

```
var endpoint = process.env.API_ENDPOINT
```

- PHP – `$_SERVER`

```
$endpoint = $_SERVER['API_ENDPOINT'];
```

- Python : `os.environ`

```
import os
endpoint = os.environ['API_ENDPOINT']
```

- Ruby – ENV

```
endpoint = ENV['API_ENDPOINT']
```

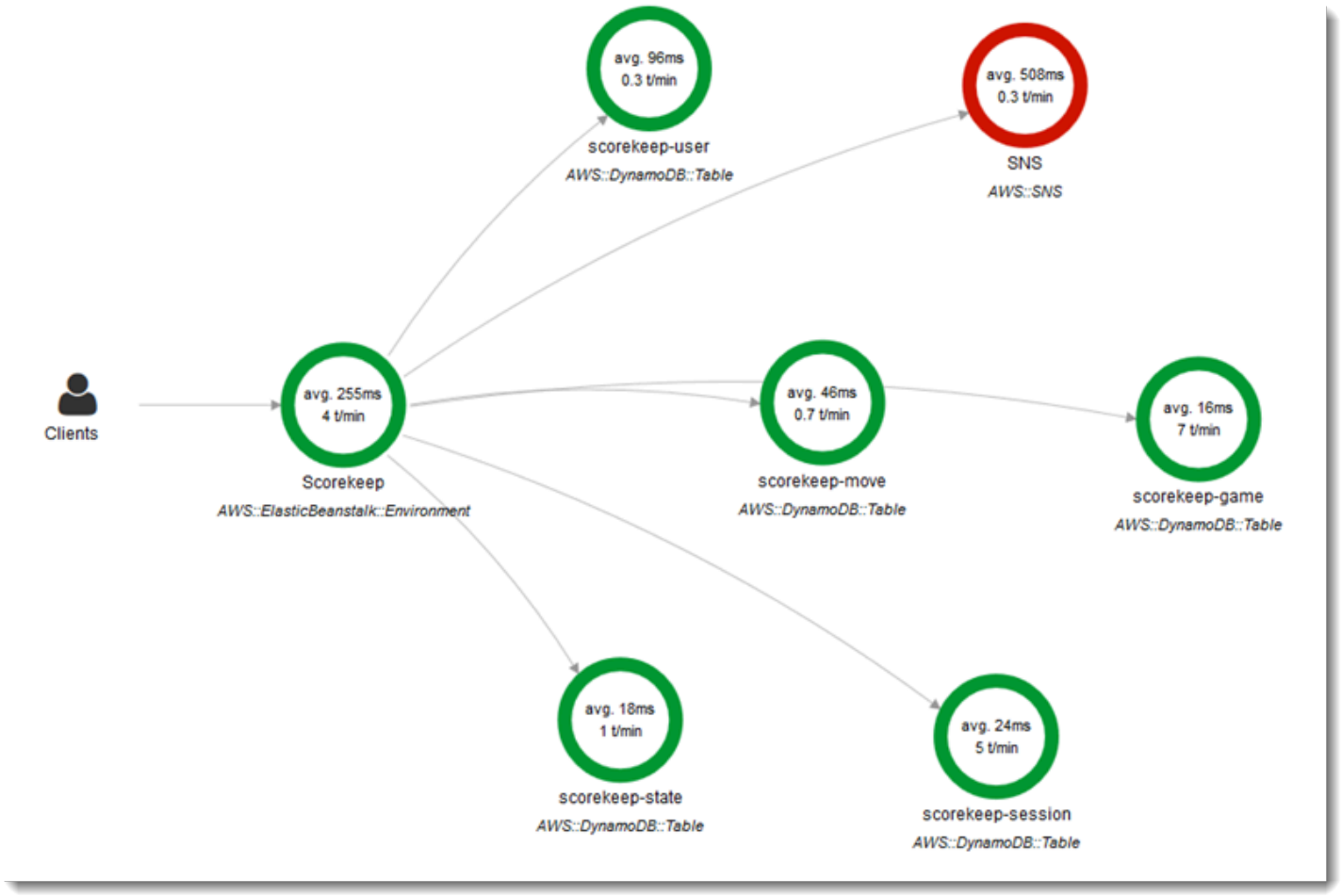
En dehors du code de l'application, par exemple dans un script exécuté pendant le déploiement, vous pouvez accéder aux propriétés de l'environnement avec le script de plateforme `get-config`. Consultez le elastic-beanstalk-samples GitHub référentiel pour des exemples de configurations qui utilisent`get-config`.

Configuration du AWS X-Ray débogage

Vous pouvez utiliser la AWS Elastic Beanstalk console ou un fichier de configuration pour exécuter le AWS X-Ray démon sur les instances de votre environnement. X-Ray est un AWS service qui collecte des données sur les demandes traitées par votre application et les utilise pour créer une carte des services que vous pouvez utiliser pour identifier les problèmes liés à votre application et les opportunités d'optimisation.

> **Note**
>
> Certaines régions ne proposent pas X-Ray. Si vous créez un environnement dans l'une de ces régions, vous ne pouvez pas exécuter le démon X-Ray sur les instances dans votre environnement.

Pour plus d'informations sur les AWS services offerts dans chaque région, consultez le tableau des régions.



X-Ray fournit un kit de développement logiciel (SDK) que vous pouvez utiliser pour instrumentaliser votre code d'application et une application démon qui transmet les informations de débogage du SDK vers l'API X-Ray.

Plateformes prises en charge

Vous pouvez utiliser le kit SDK X-Ray avec les plateformes Elastic Beanstalk suivantes :

- Go - Version 2.9.1 et ultérieure
- Java 8 - version 2.3.0 et ultérieures
- Java 8 avec Tomcat 8 - version 2.4.0 et ultérieures
- Node.js - version 3.2.0 et ultérieures

- Windows Server - toutes les versions de plateforme publiées le 18 décembre 2016 ou après cette date
- Python version 2.5.0 et ultérieures

Sur les plateformes prises en charge, vous pouvez utiliser une option de configuration pour exécuter le démon X-Ray sur les instances de votre environnement. Vous pouvez activer le démon dans la console Elastic Beanstalk ou à l'aide d'un fichier de configuration.

Pour télécharger des données vers X-Ray, le daemon X-Ray nécessite des autorisations IAM dans la politique AWSXrayWriteOnlyAccessgérée. Ces autorisations sont incluses dans le profil d'instance Elastic Beanstalk. Si vous n'utilisez pas le profil d'instance par défaut, veuillez consulter Autorisation du démon à envoyer des données à X-Ray dans le Guide du développeur AWS X-Ray .

Le débogage avec X-Ray nécessite l'utilisation du kit SDK X-Ray. Pour obtenir des instructions et des exemples d'applications, veuillez consulter Mise en route avec AWS X-Ray dans le Guide du développeur AWS X-Ray .

Si vous utilisez une version de plateforme qui n'inclut pas le démon, vous pouvez malgré tout l'exécuter avec un script dans un fichier de configuration. Pour de plus amples informations, veuillez consulter Téléchargement et exécution du Démon X-Ray manuellement (avancé) dans le Guide du développeur AWS X-Ray .

Sections

- Configuration du débogage
- Espace de noms aws:elasticbeanstalk:xray

Configuration du débogage

Vous pouvez activer le démon X-Ray sur un environnement en cours d'exécution dans la console Elastic Beanstalk.

Pour activer le débogage dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Dans la section Amazon X-Ray, sélectionnez Activé.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Vous pouvez également activer cette option lors de la création de l'environnement. Pour de plus amples informations, veuillez consulter Assistant de création d'un environnement.

Espace de noms aws:elasticbeanstalk:xray

Vous pouvez utiliser l'option `XRayEnabled` dans l'espace de noms `aws:elasticbeanstalk:xray` pour activer le débogage.

Pour activer automatiquement le débogage lorsque vous déployez votre application, définissez l'option dans un fichier de configuration dans votre code source, comme suit.

Example.ebextensions/debugging.config

```
option_settings:
  aws:elasticbeanstalk:xray:
    XRayEnabled: true
```

Affichage de vos journaux d'environnement Elastic Beanstalk

AWS Elastic Beanstalk propose deux méthodes pour consulter régulièrement les journaux des EC2 instances Amazon qui exécutent votre application :

- Configurez votre environnement Elastic Beanstalk pour charger les journaux d'instance ayant subi une rotation dans le compartiment Amazon S3 de l'environnement.
- Configurez l'environnement pour diffuser les journaux d'instance vers Amazon CloudWatch Logs.

Lorsque vous configurez le streaming des CloudWatch journaux d'instance vers Logs, Elastic CloudWatch Beanstalk crée des groupes de journaux pour les journaux de proxy et de déploiement sur les instances EC2 Amazon, et transfère ces CloudWatch fichiers journaux vers Logs en temps réel. Pour plus d'informations sur les journaux d'instance, consultez Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk.

Outre les journaux d'instance, si vous activez l'amélioration de l'état de santé de votre environnement, vous pouvez configurer l'environnement pour qu'il diffuse les informations de santé vers CloudWatch les journaux. Lorsque l'état d'intégrité de l'environnement change, Elastic Beanstalk ajoute un enregistrement à un groupe de journaux d'intégrité, avec le nouvel état et une description de la cause du changement. Pour plus d'informations sur le streaming de l'intégrité d'environnement, consultez Diffusion des informations relatives à l'état de santé de l'environnement Elastic Beanstalk vers Amazon Logs CloudWatch.

Configuration de l'affichage des journaux d'instance

Pour afficher les journaux d'instance, vous pouvez activer la rotation des journaux d'instance et le streaming des journaux dans la console Elastic Beanstalk.

Pour configurer la rotation du journal d'instance et le streaming de journaux dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Dans la section Stockage des journaux S3 sélectionnez Activé sous Rotation des journaux pour activer le chargement des journaux ayant subi une rotation vers Amazon S3.
6. dans la section Transmission du journal d'instance vers CloudWatch Logs, configurez les paramètres suivants :
   - Flux de journaux – Sélectionnez Activé pour activer le flux de journaux.
   - Conservation — Spécifiez le nombre de jours pendant lesquels les journaux sont conservés dans CloudWatch les journaux.
   - Cycle de vie : paramétré sur Supprimer les journaux lors de la fermeture pour supprimer immédiatement les CloudWatch journaux des journaux en cas de fermeture de l'environnement, au lieu d'attendre leur expiration.
7. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Après avoir activé le streaming de journaux, vous pouvez revenir à la page ou à la catégorie de configuration Logiciels pour trouver le lien Groupes de journaux. Cliquez sur ce lien pour voir les journaux de votre instance dans la CloudWatch console.

Configuration de l'affichage des journaux d'intégrité d'environnement

Pour afficher les journaux d'intégrité d'environnement, vous pouvez activer le streaming des journaux d'intégrité d'environnement sur la console Elastic Beanstalk.

Pour configurer le streaming du journal d'intégrité de l'environnement dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Passez à la section Surveillance.
6. Sous Health event streaming vers CloudWatch Logs, configurez les paramètres suivants :
   - Flux de journaux – Choisissez Activé pour activer le flux de journaux.
   - Conservation — Spécifiez le nombre de jours pendant lesquels les journaux sont conservés dans CloudWatch les journaux.
   - Cycle de vie : paramétré sur Supprimer les journaux lors de la fermeture pour supprimer immédiatement les CloudWatch journaux des journaux en cas de fermeture de l'environnement, au lieu d'attendre leur expiration.
7. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Espaces de noms de l'affichage des journaux

Les paramètres liés à l'affichage des journaux se trouvent dans les espaces de noms suivants :

- `aws:elasticbeanstalk:hostmanager` – Configurer le téléchargement de journaux tournés vers Amazon S3.

- `aws:elasticbeanstalk:cloudwatch:logs`— Configurez le streaming du journal d'instance sur CloudWatch.
- `aws:elasticbeanstalk:cloudwatch:logs:health`— Configurez le streaming de l'état de l'environnement sur CloudWatch.

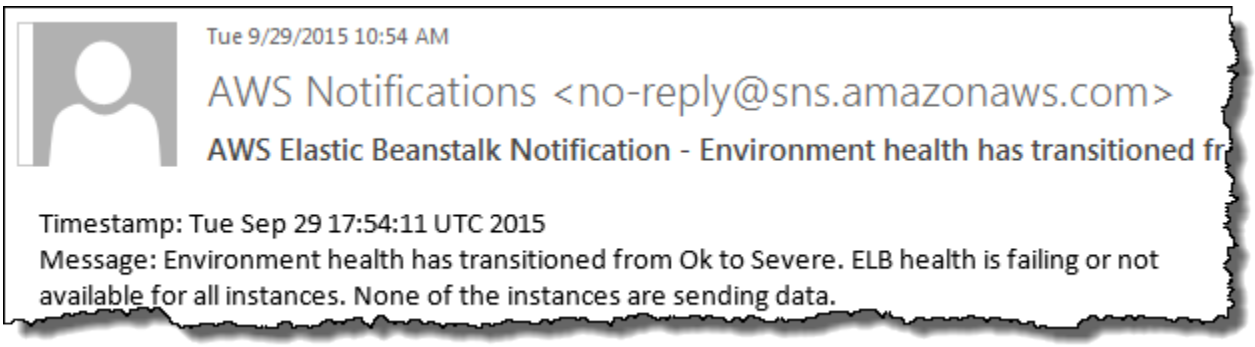
Notifications d'environnement Elastic Beanstalk avec Amazon SNS

Vous pouvez configurer votre AWS Elastic Beanstalk environnement pour utiliser Amazon Simple Notification Service (Amazon SNS) afin de vous informer des événements importants qui affectent votre application. Pour recevoir des e-mails AWS chaque fois qu'une erreur se produit ou que l'état de votre environnement change, spécifiez une adresse e-mail lors de la création d'un environnement ou ultérieurement.

> **Note**
>
> Elastic Beanstalk utilise Amazon SNS pour les notifications. Pour plus d'informations sur la tarification d'Amazon SNS, consultez https://aws.amazon.com/sns/ la section pricing/.

Lorsque vous configurez des notifications pour votre environnement, Elastic Beanstalk crée une rubrique Amazon SNS pour votre environnement en votre nom. Pour envoyer des messages à une rubrique Amazon SNS, Elastic Beanstalk doit disposer de l'autorisation requise. Pour de plus amples informations, veuillez consulter Configuration des autorisations d'envoi de notifications.

Lorsqu'un événement notable se produit, Elastic Beanstalk envoie un message à la rubrique. Ensuite, Amazon SNS transmet les messages qu'il reçoit aux abonnés à la rubrique. Les événements importants incluent les erreurs de création d'environnement et toutes les modifications de l'état d'un environnement ou d'une instance. Les événements relatifs à des opérations Amazon EC2 Auto Scaling (comme l'ajout et la suppression d'instances dans l'environnement) et les autres événements informatifs ne déclenchent pas l'envoi de notifications.



Vous pouvez entrer une adresse e-mail dans la console Elastic Beanstalk lorsque vous créez un environnement ou quelque temps après. Cela créera une rubrique Amazon SNS et vous y abonnera. Elastic Beanstalk gère le cycle de vie de la rubrique et la supprime lorsque votre environnement est arrêté ou lorsque vous supprimez votre adresse e-mail de la console de gestion de l'environnement.

L'espace de noms `aws:elasticbeanstalk:sns:topics` fournit des options permettant de configurer une rubrique Amazon SNS à l'aide de fichiers de configuration, d'une interface de ligne de commande (CLI) ou d'un kit SDK. En employant l'une de ces méthodes, vous pouvez configurer le type d'abonné et le point de terminaison. Pour le type d'abonné, vous pouvez choisir une file d'attente Amazon SQS ou une URL HTTP.

Vous pouvez seulement activer ou désactiver les notifications Amazon SNS. Selon la taille et la composition de votre environnement, la fréquence des notifications envoyées à la rubrique peut être élevée. Pour configurer les notifications à envoyer dans des cas précis, vous disposez d'autres options. Vous pouvez configurer des règles basées sur les événements avec Amazon EventBridge pour vous avertir lorsqu'Elastic Beanstalk émet des événements répondant à des critères spécifiques. Vous pouvez également configurer votre environnement pour publier des métriques personnalisées et configurer des CloudWatch alarmes Amazon pour vous avertir lorsque ces métriques atteignent un seuil non satisfaisant.

Configuration des notifications à l'aide de la console Elastic Beanstalk

Vous pouvez entrer une adresse e-mail dans la console Elastic Beanstalk pour créer une rubrique Amazon SNS pour votre environnement.

Pour configurer les notifications dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Faites défiler jusqu'à la section Notifications par e-mail.
6. Entrez une adresse e-mail.
7. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Si vous saisissez une adresse e-mail pour les notifications, Elastic Beanstalk crée une rubrique Amazon SNS pour votre environnement et ajoute un abonnement. Amazon SNS envoie un e-mail à l'adresse indiquée afin de confirmer l'abonnement. Vous devez cliquer sur le lien figurant dans l'e-mail de confirmation afin d'activer l'abonnement et de recevoir les notifications.

Configuration des notifications à l'aide des options de configuration

Utilisez les options de l'espace de noms `aws:elasticbeanstalk:sns:topics` afin de configurer les notifications Amazon SNS pour votre environnement. Vous pouvez définir ces options à l'aide de fichiers de configuration, d'une CLI ou d'un kit SDK.

- Point de terminaison de notification : l'adresse e-mail, la file d'attente Amazon SQS ou l'URL à laquelle les notifications doivent être envoyées. Si vous définissez cette option, une file d'attente SQS et un abonnement pour le point de terminaison spécifié sont créés. Si le point de terminaison n'est pas une adresse e-mail, vous devez également définir l'option `Notification Protocol`. SNS valide la valeur de `Notification Endpoint` en fonction de la valeur de `Notification Protocol`. Si vous configurez plusieurs fois cette option, vous créez des abonnements supplémentaires à la rubrique. Si vous supprimez cette option, la rubrique est supprimée.
- Protocole de notification : le protocole qui est utilisé pour envoyer les notifications au `Notification Endpoint`. La valeur par défaut de cette option est `email`. Définissez cette option sur `email-json` pour envoyer des e-mails au format JSON, sur `http` ou `https` pour publier les notifications au format JSON à un point de terminaison HTTP, ou sur `sqs` pour envoyer les notifications à une file d'attente SQS.

> **Note**
>
> AWS Lambda les notifications ne sont pas prises en charge.

- ARN de rubrique de notification : une fois que vous avez défini un point de terminaison de notification pour votre environnement, consultez ce paramètre afin d'obtenir l'ARN de la rubrique SNS. Vous pouvez également définir cette option afin d'utiliser une rubrique SNS existante pour les notifications. La rubrique que vous associez à votre environnement via cette option n'est pas supprimée lorsque l'option est modifiée ou que l'environnement est arrêté.

  Pour configurer les notifications Amazon SNS, vous devez disposer des autorisations requises. Si votre utilisateur IAM utilise la politique relative aux utilisateurs gérés par AdministratorAccessElastic AWSElastic Beanstalk - Beanstalk, vous devriez déjà disposer des autorisations requises pour configurer la rubrique Amazon SNS par défaut créée par Elastic Beanstalk pour votre environnement. Toutefois, si vous configurez une rubrique Amazon SNS qu'Elastic Beanstalk ne gère pas, vous devez ajoutez la stratégie suivante à votre rôle utilisateur.

  JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sns:SetTopicAttributes",
        "sns:GetTopicAttributes",
        "sns:Subscribe",
        "sns:Unsubscribe",
        "sns:Publish"
      ],
      "Resource": [
        "arn:aws:sns:us-east-2:123456789012:sns_topic_name"
      ]
    }
  ]
}
```

- Nom de rubrique de notification : définissez cette option pour personnaliser le nom de la rubrique Amazon SNS utilisée pour les notifications relatives à l'environnement. Si une rubrique portant le même nom existe déjà, Elastic Beanstalk associe cette rubrique à l'environnement.

  > **Warning**
  >
  > Si vous associez une rubrique SNS existante à un environnement avec `Notification Topic Name`, Elastic Beanstalk supprimera la rubrique si vous arrêtez l'environnement ou modifiez ce paramètre ultérieurement.

  Si vous modifiez cette option, l'`Notification Topic ARN` est également modifié. Si une rubrique est déjà associée à l'environnement, Elastic Beanstalk supprime l'ancienne rubrique et crée une rubrique et un abonnement.

  En utilisant un nom de rubrique personnalisé, vous devez également fournir l'ARN d'une rubrique personnalisée créée en externe. Comme la stratégie utilisateur géré ne détecte pas automatiquement une rubrique avec un nom personnalisé, vous devez fournir des autorisations Amazon SNS personnalisées à vos utilisateurs IAM. Utilisez une stratégie semblable à celle utilisée pour un ARN de rubrique personnalisée, mais incluez les ajouts suivants :

  - Incluez deux autres actions dans la liste `Actions`, en particulier : `sns:CreateTopic` et `sns:DeleteTopic`
  - Si vous remplacez le nom `Notification Topic Name` d'un sujet personnalisé par un autre, vous devez également inclure ARNs les deux sujets dans la `Resource` liste. En variante, incluez une expression régulière qui couvre les deux rubriques. Ainsi, Elastic Beanstalk dispose des autorisations pour supprimer l'ancienne rubrique et en créer une nouvelle.

La CLI EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Consultez Valeurs recommandées pour plus de détails.

Configuration des autorisations d'envoi de notifications

Cette section présente les considérations de sécurité liées aux notifications envoyées avec Amazon SNS. Il existe deux cas distincts :

- Utilisez la rubrique Amazon SNS par défaut qu'Elastic Beanstalk crée pour votre environnement.

- Fournissez une rubrique Amazon SNS externe en utilisant les options de configuration.

La stratégie d'accès par défaut pour une rubrique Amazon SNS permet au seul propriétaire de la rubrique de la publier ou de s'y abonner. Toutefois, en utilisant la configuration de stratégie appropriée, Elastic Beanstalk peut être autorisé à publier sur une rubrique Amazon SNS dans l'un des deux cas décrits dans cette section. Vous trouverez davantage d'informations dans les sous-sections suivantes.

Autorisations pour une rubrique par défaut

Lorsque vous configurez des notifications pour votre environnement, Elastic Beanstalk crée une rubrique Amazon SNS pour votre environnement. Pour envoyer des messages à une rubrique Amazon SNS, Elastic Beanstalk doit disposer de l'autorisation requise. Si votre environnement utilise le rôle de service généré pour lui par la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB, ou le rôle lié au service de surveillance de votre compte, vous n'avez pas besoin de faire quoi que ce soit d'autre. Ces rôles gérés contiennent l'autorisation nécessaire qui permet à Elastic Beanstalk d'envoyer des messages à la rubrique Amazon SNS.

Toutefois, si vous avez fourni un rôle de service personnalisé lorsque vous avez créé votre environnement, vérifiez que ce rôle de service personnalisé inclut bien la stratégie suivante.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": [
        "arn:aws:sns:us-east-2:123456789012:ElasticBeanstalkNotifications*"
      ]
    }
  ]
}
```

Autorisations pour une rubrique externe

Configuration des notifications à l'aide des options de configuration explique la façon dont vous pouvez remplacer la rubrique Amazon SNS qu'Elastic Beanstalk fournit avec une autre rubrique Amazon SNS. Si vous avez remplacé la rubrique, Elastic Beanstalk doit vérifier que vous êtes autorisé à publier dans cette rubrique SNS afin que vous puissiez associer la rubrique SNS à l'environnement. Vous devez disposer de `sns:Publish`. Le rôle de service utilise la même autorisation. Pour vérifier que c'est le cas, Elastic Beanstalk envoie une notification test à SNS dans le cadre de votre action de création ou de mise à jour de l'environnement. En cas d'échec de ce test, votre tentative de création ou de mise à jour de l'environnement échoue également. Elastic Beanstalk affiche un message détaillant la raison de cet échec.

Si vous fournissez un rôle de service personnalisé pour votre environnement, vérifiez que votre rôle de service personnalisé inclut bien la stratégie suivante pour permettre à Elastic Beanstalk d'envoyer des messages à la rubrique Amazon SNS. Dans le code suivant, remplacez *sns_topic_name* par le nom de la rubrique Amazon SNS que vous avez fourni dans les options de configuration.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": [
        "arn:aws:sns:us-east-2:123456789012:sns_topic_name"
      ]
    }
  ]
}
```

Pour plus d'informations sur le contrôle d'accès Amazon SNS, veuillez consulter Exemples de cas pour le contrôle d'accès Amazon SNS dans le Guide du développeur Amazon Simple Notification Service.

Configuration d'Amazon Virtual Private Cloud (Amazon VPC) avec Elastic Beanstalk

Amazon Virtual Private Cloud (Amazon VPC) est le service réseau qui achemine le trafic en toute sécurité vers les EC2 instances qui exécutent votre application dans Elastic Beanstalk. Si vous ne configurez pas de VPC lorsque vous lancez votre environnement, Elastic Beanstalk utilise le VPC par défaut.

Vous pouvez lancer votre environnement dans un VPC personnalisé afin de personnaliser les paramètres de mise en réseau et de sécurité. Elastic Beanstalk vous permet de choisir quels sous-réseaux utiliser pour vos ressources et comment configurer les adresses IP des instances et de l'équilibreur de charge dans votre environnement. Un environnement est lié à un VPC lorsque vous le créez, mais vous pouvez modifier les paramètres de sous-réseau et d'adresse IP dans un environnement d'exécution.

Configuration des paramètres VPC dans la console Elastic Beanstalk

Si vous avez sélectionné un VPC personnalisé lors de la création de votre environnement, vous pouvez modifier ses paramètres VPC dans la console Elastic Beanstalk.

Pour configurer les paramètres VPC de votre environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Network (Réseau), choisissez Edit (Modifier).

Les paramètres suivants sont disponibles.

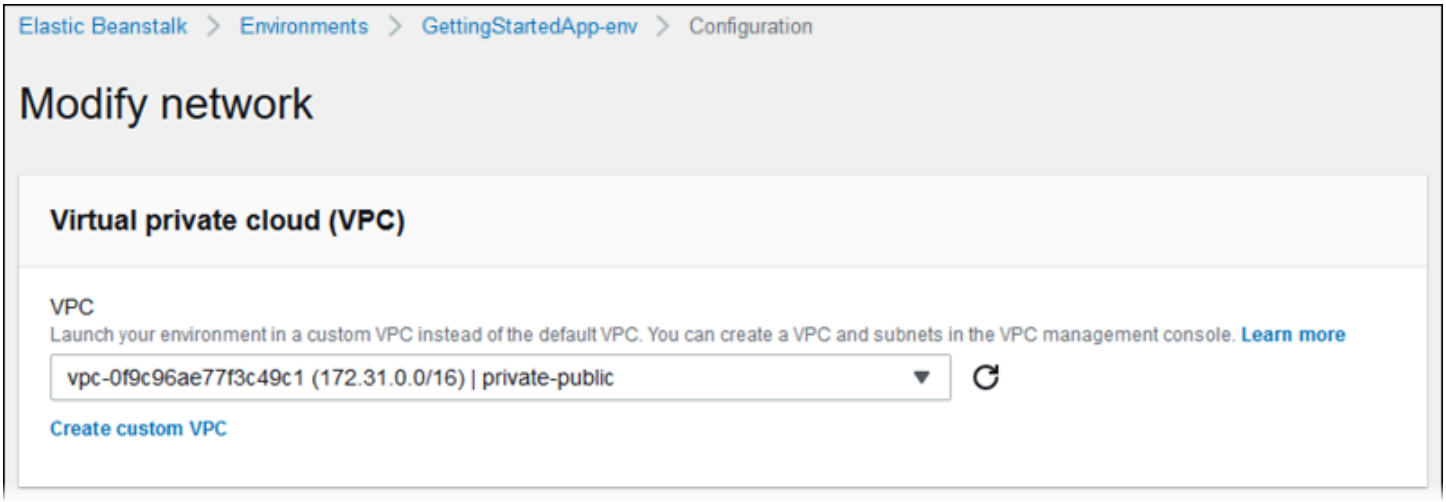
Options

- VPC
- Visibilité de l'équilibreur de charge
- Sous-réseaux de l'équilibreur de charge
- Adresse IP publique d'instance

- Sous-réseaux d'instance
- Sous-réseaux de la base de données

VPC

Sélectionnez un VPC pour votre environnement. Vous ne pouvez modifier ce paramètre que lors de la création de l'environnement.

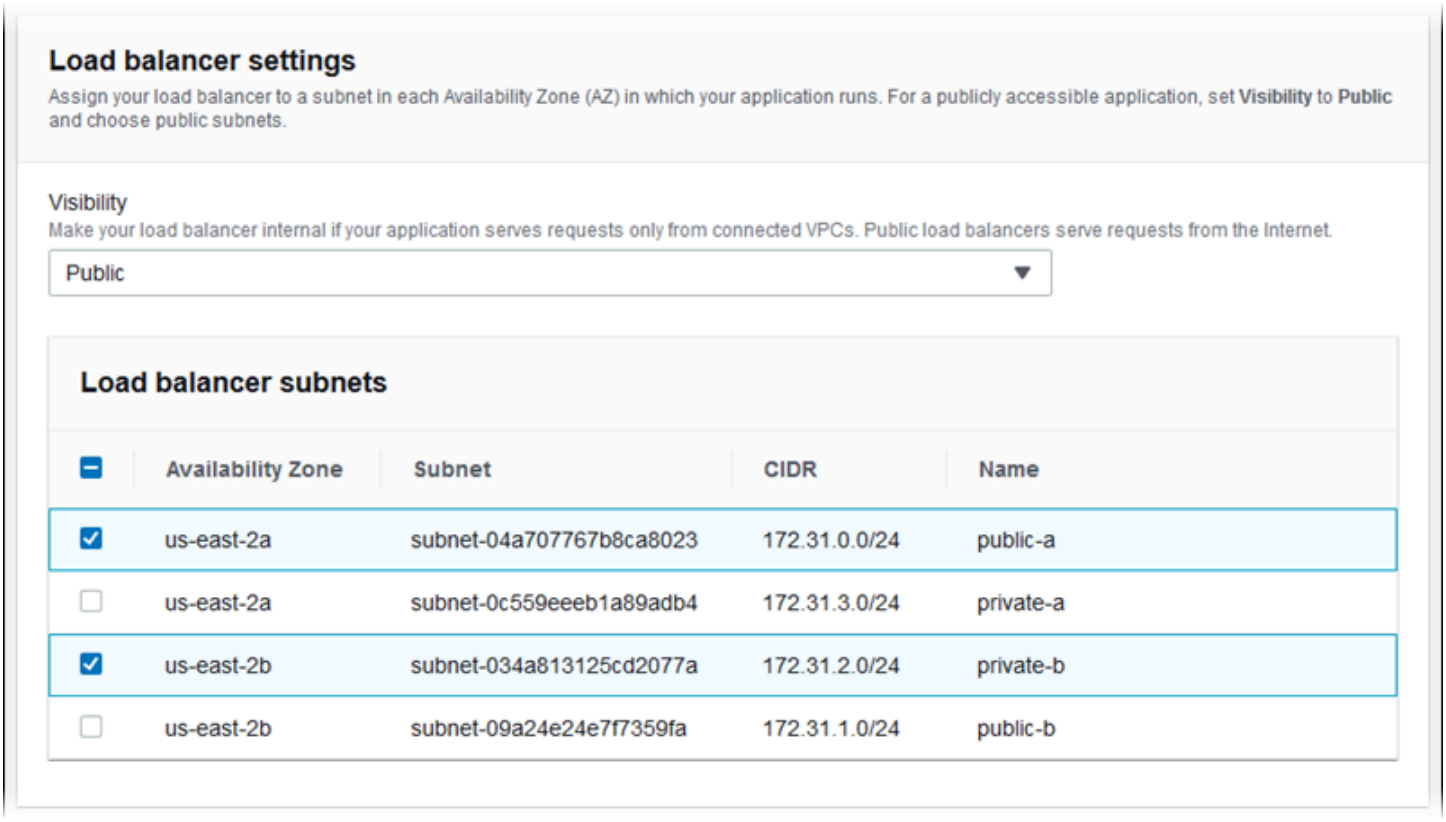


Visibilité de l'équilibreur de charge

Dans un environnement à charge équilibrée, choisissez le schéma de l'équilibreur de charge. Par défaut, l'équilibreur de charge est public, avec une adresse IP et un nom de domaine publics. Si votre application ne gère que du trafic provenant de votre VPC ou d'un VPN connecté, désélectionnez cette option et choisissez des sous-réseaux privés pour votre équilibreur de charge afin de rendre l'équilibreur de charge interne et de désactiver l'accès depuis Internet.

Sous-réseaux de l'équilibreur de charge

Dans un environnement à charge équilibrée, choisissez les sous-réseaux que votre équilibreur de charge utilise pour gérer le trafic. Dans le cas d'une application publique, choisissez des sous-réseaux publics. Utilisez des sous-réseaux dans plusieurs zones de disponibilité pour une haute disponibilité. Dans le cas d'une application interne, choisissez des sous-réseaux privés et désactivez la visibilité de l'équilibreur de charge.

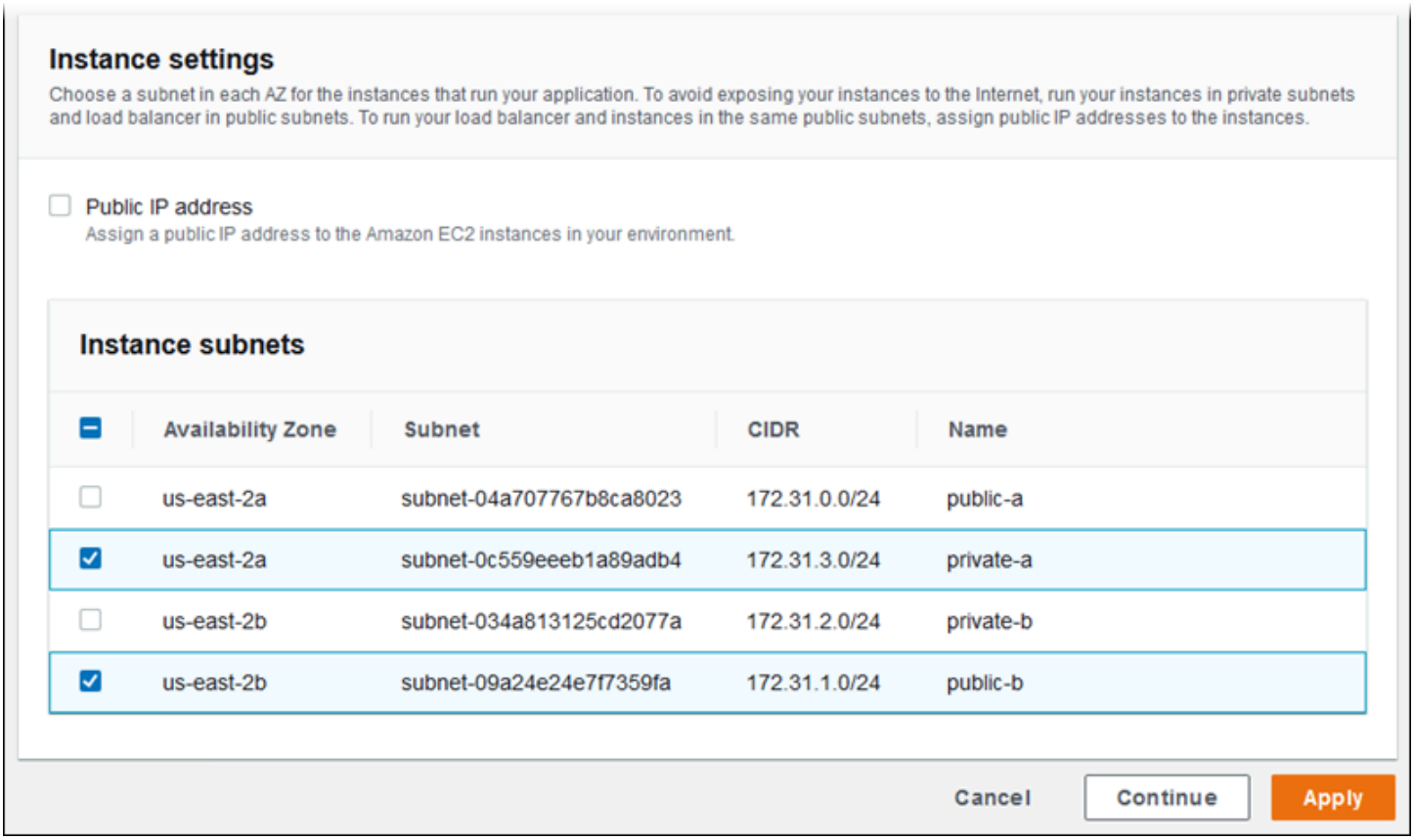


Adresse IP publique d'instance

Si vous choisissez des sous-réseaux publics pour les instances de votre application, activez des adresses IP publiques pour les rendre routables à partir d'Internet.

Sous-réseaux d'instance

Choisissez des sous-réseaux pour les instances de votre application. Choisissez au moins un sous-réseau pour chaque zone de disponibilité utilisée par votre équilibreur de charge. Si vous choisissez des sous-réseaux privés pour vos instances, votre VPC doit avoir une passerelle NAT dans un sous-réseau public que les instances peuvent utiliser pour accéder à Internet.



Sous-réseaux de la base de données

Lorsque vous exécutez une base de données Amazon RDS attachée à votre environnement Elastic Beanstalk, choisissez des sous-réseaux pour vos instances de base de données. Pour une haute disponibilité, rendez la base de données Multi-AZ et choisissez un sous-réseau pour chaque zone de disponibilité. Pour vous assurer que votre application puisse se connecter à votre base de données, exécutez les deux dans les mêmes sous-réseaux.

Espace de noms aws:ec2:vpc

Vous pouvez utiliser les options de configuration de l'espace de noms `aws:ec2:vpc` pour configurer les paramètres réseau de votre environnement.

Le fichier de configuration suivant utilise des options dans cet espace de noms pour définir le VPC et les sous-réseaux de l'environnement pour une configuration publique-privée. Pour définir l'ID de VPC dans un fichier de configuration, le fichier doit être inclus dans le bundle de sources de l'application lors de la création de l'environnement. Consultez Définition des options de configuration lors de la création de l'environnement pour d'autres méthodes de configuration de ces paramètres lors de la création de l'environnement.

Example.ebextensions/vpc.config – Public-private

```
option_settings:
   aws:ec2:vpc:
      VPCId: vpc-087a68c03b9c50c84
      AssociatePublicIpAddress: 'false'
      ELBScheme: public
      ELBSubnets: subnet-0fe6b36bcb0ffc462,subnet-032fe3068297ac5b2
      Subnets: subnet-026c6117b178a9c45,subnet-0839e902f656e8bd1
```

Cet exemple montre une configuration publique-publique, dans laquelle l'équilibreur de charge et les EC2 instances s'exécutent dans les mêmes sous-réseaux publics.

Example.ebextensions/vpc.config – Public-public

```
option_settings:
   aws:ec2:vpc:
      VPCId: vpc-087a68c03b9c50c84
      AssociatePublicIpAddress: 'true'
      ELBScheme: public
      ELBSubnets: subnet-0fe6b36bcb0ffc462,subnet-032fe3068297ac5b2
      Subnets: subnet-0fe6b36bcb0ffc462,subnet-032fe3068297ac5b2
```

Migration des environnements Elastic EC2 Beanstalk de -Classic vers un VPC

Cette rubrique décrit les différentes options permettant de migrer vos environnements Elastic Beanstalk d'une plateforme réseau -Classic EC2 vers un réseau Amazon Virtual Private Cloud (Amazon VPC).

Si vous avez créé votre AWS compte avant le 4 décembre 2013, certains Régions AWS environnements utilisent peut-être la configuration réseau EC2 -Classic. Tous les AWS comptes créés le 4 décembre 2013 ou après cette date sont déjà réservés aux VPN dans toutes les AWS régions. Les seules exceptions concernent les cas où Amazon EC2 -Classic a été activé à la suite d'une demande d'assistance.

> **Note**
>
> Vous pouvez afficher les paramètres de configuration réseau de votre environnement dans la catégorie Network configuration (Configuration réseau) de la page Configuration overview (Aperçu de la configuration) de la console Elastic Beanstalk.

Pourquoi migrer

Amazon EC2 -Classic atteindra la fin du support standard le 15 août 2022. Pour éviter toute interruption de vos charges de travail, nous vous recommandons de migrer d'Amazon EC2 -Classic vers un VPC avant le 15 août 2022. Nous vous demandons également de ne lancer aucune AWS ressource sur Amazon EC2 -Classic à l'avenir et d'utiliser Amazon VPC à la place.

Lorsque vous migrez vos environnements Elastic Beanstalk d' EC2Amazon -Classic vers Amazon VPC, vous devez créer un nouveau compte. AWS Vous devez également recréer vos environnements AWS EC2 -Classic dans votre nouveau AWS compte. Aucun travail de configuration supplémentaire pour vos environnements n'est requis afin d'utiliser le VPC par défaut. Si le VPC par défaut ne répond pas à vos exigences, créez manuellement un VPC personnalisé et associez-le à vos environnements.

Sinon, si votre AWS compte existant possède des ressources que vous ne pouvez pas migrer vers un nouveau AWS compte, ajoutez un VPC à votre compte actuel. Configurez ensuite vos environnements pour qu'ils utilisent le VPC.

Pour plus d'informations, consultez le billet de blog EC2-Classic Networking is Retiring - Here's How to Prepare.

Migrer un environnement de EC2 -Classic vers un nouveau AWS compte (recommandé)

Si vous ne possédez pas encore de AWS compte créé le 4 décembre 2013 ou après cette date, créez-en un nouveau. Vous allez procéder à la migration de vos environnements vers ce nouveau compte.

1. Votre nouveau AWS compte fournit un VPC par défaut à ses environnements. Si vous n'avez pas besoin de créer un VPC personnalisé, passez à l'étape 2.

   Vous pouvez créer un VPC personnalisé de l'une des manières suivantes :

- Créez rapidement un VPC à l'aide de l'assistant de console Amazon VPC et de l'une des options de configuration disponibles. Pour de plus amples informations, veuillez consulter Configurations de l'assistant de la console Amazon VPC.
- Créez un VPC personnalisé sur la console Amazon VPC si vous disposez d'exigences plus spécifiques pour votre VPC. Nous vous recommandons cette opération, par exemple, si votre cas d'utilisation nécessite un nombre spécifique de sous-réseaux. Pour plus d'informations, reportez-vous à la section VPCs et sous-réseaux.
- Créez un VPC à l'aide du elastic-beanstalk-samplesréférentiel sur le GitHub site Web si vous préférez utiliser des AWS CloudFormation modèles avec vos environnements Elastic Beanstalk. Ce référentiel inclut des CloudFormation modèles. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon VPC.

> **Note**
>
> Vous pouvez également créer un VPC personnalisé en même temps que vous recréez l'environnement dans votre nouveau AWS compte à l'aide de l'assistant de création d'un nouvel environnement. Si vous utilisez l'assistant et choisissez de créer un VPC personnalisé, l'assistant vous redirige vers la console Amazon VPC.

2. Dans votre nouveau AWS compte, créez un nouvel environnement. Nous recommandons que l'environnement inclue la même configuration que votre environnement existant dans le AWS compte à partir duquel vous effectuez la migration. Pour ce faire, utilisez l'une des méthodes suivantes :

> **Note**
>
> Si votre nouvel environnement doit utiliser le même CNAME après la migration, mettez fin à l'environnement d'origine sur la plateforme EC2 -Classic. En conséquence, CNAME est libéré et prêt à être utilisé. Cependant, cela peut entraîner des interruptions de service pour cet environnement et le risque qu'un autre client sélectionne votre CNAME entre la fin de votre environnement EC2 -Classic et la création du nouveau. Pour de plus amples informations, veuillez consulter Arrêt d'un environnement Elastic Beanstalk. Pour les environnements qui ont leur propre nom de domaine propriétaire, le CNAME n'a pas ce problème. Vous pouvez simplement mettre à jour votre système de noms de domaine (DNS) pour transférer les demandes à votre nouveau CNAME.

- Utilisez l' assistant de création d'environnement sur la console Elastic Beanstalk. L'Assistant fournit une option pour créer un VPC personnalisé. Si vous ne choisissez pas de créer un VPC personnalisé, un VPC par défaut est affecté.
- Utilisez l'interface de ligne de commande Elastic Beanstalk (EB CLI) pour recréer votre environnement dans votre nouveau compte. AWS L'un des exemples de la description de la commande eb create illustre la création d'un environnement dans un VPC personnalisé. Si vous ne fournissez pas d'ID de VPC, l'environnement utilise le VPC par défaut.

  En utilisant cette approche, vous pouvez utiliser un fichier de configuration enregistré sur les deux AWS comptes. Par conséquent, vous ne devez pas saisir manuellement toutes les informations de configuration. Cependant, vous devez enregistrer les paramètres de configuration de l'environnement EC2 -Classic que vous migrez à l'aide de la commande eb config save. Copiez le fichier de configuration enregistré dans un nouveau répertoire pour le nouvel environnement de compte.

  > **Note**
  >
  > Vous devez modifier certaines données du fichier de configuration enregistré avant de pouvoir les utiliser dans le nouveau compte. Vous devez également mettre à jour les informations relatives à votre compte précédent avec les bonnes données pour votre nouveau compte. Par exemple, vous devez remplacer l'Amazon Resource Name (ARN) du rôle Gestion des identités et des accès AWS (IAM) par l'ARN du rôle IAM pour le nouveau compte.

  Si vous utilisez la commande eb create avec l'option `cfg`, le nouvel environnement est créé à l'aide du fichier de configuration enregistré spécifié. Pour de plus amples informations, veuillez consulter Utilisation des configurations enregistrées par Elastic Beanstalk.

Migrer un environnement depuis EC2 -Classic au sein de votre même compte AWS

Votre AWS compte existant contient peut-être des ressources que vous ne pouvez pas migrer vers un nouveau AWS compte. Dans ce cas, vous devez recréer vos environnements et configurer manuellement un VPC pour chaque environnement que vous créez.

Migrer vos environnements vers un VPC personnalisé

Conditions préalables

Avant de commencer, vous devez disposer d'un VPC. Vous pouvez créer un VPC non par défaut (personnalisé) de l'une des manières suivantes :

- Créez rapidement un VPC à l'aide de l'assistant de console Amazon VPC et de l'une des options de configuration disponibles. Pour de plus amples informations, veuillez consulter Configurations de l'assistant de la console Amazon VPC.
- Créez un VPC personnalisé sur la console Amazon VPC si vous disposez d'exigences plus spécifiques pour votre VPC. Nous vous recommandons cette opération, par exemple, si votre cas d'utilisation nécessite un nombre spécifique de sous-réseaux. Pour plus d'informations, reportez-vous à la section VPCs et sous-réseaux.
- Créez un VPC à l'aide du elastic-beanstalk-samplesréférentiel sur le GitHub site Web si vous préférez utiliser des AWS CloudFormation modèles avec vos environnements Elastic Beanstalk. Ce référentiel inclut des CloudFormation modèles. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon VPC.

Dans les étapes suivantes, vous utiliserez l'ID VPC et le sous-réseau générés IDs lorsque vous configurez le VPC dans le nouvel environnement.

1. Créez un nouvel environnement qui inclut la même configuration que votre environnement existant. Pour ce faire, utilisez l'une des méthodes suivantes :

   > **Note**
   >
   > La fonctionnalité Configurations enregistrées peut vous aider à recréer vos environnements dans le nouveau compte. Cette fonction peut enregistrer la configuration d'un environnement de manière à ce que vous puissiez l'appliquer lorsque vous créez ou mettez à jour d'autres environnements. Pour de plus amples informations, veuillez consulter Utilisation des configurations enregistrées par Elastic Beanstalk.

   - À l'aide de la console Elastic Beanstalk, appliquez une configuration enregistrée EC2 depuis votre environnement -Classic lorsque vous configurez le nouvel environnement. Cette configuration utilise le VPC. Pour de plus amples informations, veuillez consulter Utilisation des configurations enregistrées par Elastic Beanstalk.

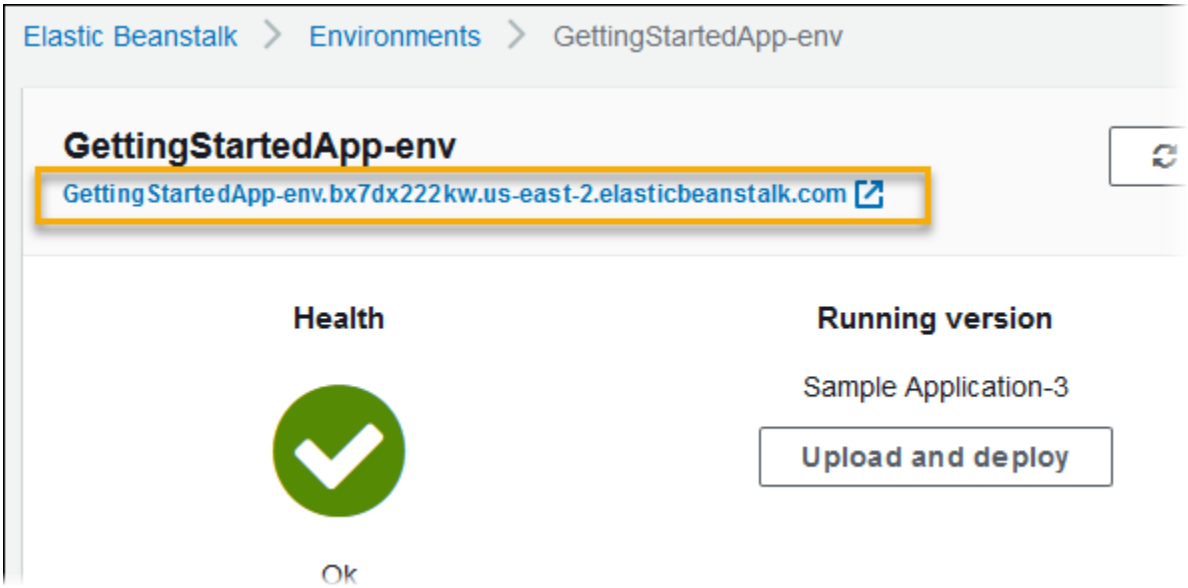
- À l'aide de l'interface de ligne de commande Elastic Beanstalk (EB CLI), exécutez la commande eb create pour recréer votre environnement. Fournissez les paramètres de votre environnement d'origine et l'identifiant du VPC. L'un des exemples de la description de la commande eb create illustre la création d'un environnement dans un VPC personnalisé.
- Utilisez le AWS Command Line Interface (AWS CLI) et recréez votre environnement à l'aide de la elasticbeanstalk create-environment commande. Fournissez les paramètres de votre environnement d'origine avec l'identifiant VPC. Pour obtenir des instructions, veuillez consulter Création d'environnements Elastic Beanstalk avec la CLI AWS.

2. Remplacez CNAMEs l'environnement existant par le nouvel environnement. De cette façon, le nouvel environnement que vous avez créé peut être référencé avec l'adresse familière. Vous pouvez utiliser l'interface de ligne de commande (CLI) EB ou le AWS CLI.
   - À l'aide de la CLI EB, changez d'environnement en CNAMEs exécutant la eb swap commande. Pour de plus amples informations, veuillez consulter Configuration de l'interface de ligne de commande EB (EB CLI) pour gérer Elastic Beanstalk.
   - À l'aide de AWS CLI, remplacez l'environnement CNAMEs par la commande swap-environment-cnameselasticbeanstalk. Pour plus d'informations, consultez la référence de la commande AWS CLI.

Nom de domaine de votre environnement Elastic Beanstalk

Par défaut, votre environnement est mis à la disposition des utilisateurs à un sous-domaine de `elasticbeanstalk.com`. Lorsque vous créez un environnement, vous pouvez choisir un nom d'hôte pour votre application. Le sous-domaine et le domaine sont autorenseignés sur *`region`*`.elasticbeanstalk.com`.

Pour acheminer les utilisateurs vers votre environnement, Elastic Beanstalk effectue un enregistrement CNAME qui pointe vers l'équilibreur de charge de votre environnement. Vous pouvez voir l'URL de l'application de votre environnement avec la valeur actuelle du CNAME dans la page de présentation de l'environnement de la console Elastic Beanstalk.



Choisissez l'URL sur la page de présentation ou choisissez Go to environment (Aller à l'environnement) dans le volet de navigation pour accéder à la page web de votre application.

Vous pouvez changer le CNAME sur votre environnement en l'échangeant avec le CNAME d'un autre environnement. Pour obtenir des instructions, consultez Déploiements bleu/vert avec Elastic Beanstalk.

Si vous possédez un nom de domaine, vous pouvez utiliser Amazon Route 53 pour le résoudre dans votre environnement. Vous pouvez acheter un nom de domaine avec Amazon Route 53, ou en utiliser un que vous achetez auprès d'un autre fournisseur.

Pour acheter un nom de domaine avec Route 53, veuillez consulter Enregistrement d'un nouveau domaine dans le Guide du développeur Amazon Route 53.

Pour en savoir plus sur l'utilisation d'un domaine personnalisé, veuillez consulter Routage du trafic vers un environnement AWS Elastic Beanstalk dans le Guide du développeur Amazon Route 53.

> **Important**
>
> Si vous résiliez un environnement, vous devez également supprimer tous les mappages CNAME que vous avez créés, car d'autres clients peuvent réutiliser un nom d'hôte disponible. Veillez à supprimer les enregistrements DNS qui pointent vers votre environnement résilié, afin d'éviter toute entrée DNS en suspens. Une entrée DNS bloquée peut exposer le trafic Internet destiné à votre domaine à des failles de sécurité. Elle peut également présenter d'autres risques.

Pour de plus amples informations, veuillez consulter Protection contre les enregistrements de délégation suspendus dans Route 53 dans le manuel du développeur Amazon Route 53. Vous pouvez également en savoir plus sur le blocage des entrées DNS dans Enhanced Domain Protections for Amazon CloudFront Requests sur le blog sur la AWS sécurité.

Configuration d'environnements Elastic Beanstalk (niveau avancé)

Lorsque vous créez un AWS Elastic Beanstalk environnement, Elastic Beanstalk fournit et configure toutes les ressources nécessaires à l'exécution et au AWS support de votre application. En plus de configurer le comportement de mise à jour et les métadonnées de votre environnement, vous pouvez personnaliser ces ressources en fournissant des valeurs pour les options de configuration. Par exemple, vous pouvez ajouter une file d'attente Amazon SQS et une alarme indiquant la profondeur de la file d'attente, ou vous pouvez ajouter un cluster Amazon ElastiCache .

La plupart des options de configuration sont associées à des valeurs par défaut qui sont automatiquement appliquées par Elastic Beanstalk. Vous pouvez remplacer ces valeurs par défaut via des fichiers de configuration, des configurations enregistrées, des options de ligne de commande, ou en appelant directement l'API Elastic Beanstalk. L'interface de ligne de commande EB et la console Elastic Beanstalk appliquent également les valeurs recommandées pour certaines options.

Vous pouvez facilement personnaliser votre environnement en même temps que vous déployez votre version d'application, en incluant un fichier de configuration dans votre bundle source. Lorsque vous personnalisez le logiciel sur votre instance, il est plus avantageux d'utiliser un fichier de configuration que de créer une AMI personnalisée, car il n'est pas nécessaire de gérer un ensemble de AMIs.

Lorsque vous déployez vos applications, vous pouvez personnaliser et configurer le logiciel dont dépend votre application. Ces fichiers peuvent être des dépendances requises par l'application (par exemple, des packages supplémentaires provenant du référentiel yum) ou des fichiers de configuration (par exemple, pour que httpd.conf remplace des paramètres spécifiques qui sont définis par défaut par AWS Elastic Beanstalk).

Rubriques

- Options de configuration
- Personnalisation d'environnement avancée avec fichiers de configuration (.ebextensions)
- Utilisation des configurations enregistrées par Elastic Beanstalk
- Manifeste d'environnement (env.yaml)
- Utilisation d'une image machine Amazon (AMI) personnalisée dans votre environnement Elastic Beanstalk
- Service de fichiers statiques

- Configuration de HTTPS pour votre environnement Elastic Beanstalk

Options de configuration

Elastic Beanstalk définit un grand nombre d'options de configuration que vous pouvez utiliser pour configurer le comportement de votre environnement et les ressources qu'il contient. Les options de configuration sont organisées en espaces de noms comme `aws:autoscaling:asg`, qui définit les options de groupe Auto Scaling d'un environnement.

La console Elastic Beanstalk et l'interface de ligne de commande (CLI) EB définissent les options de configuration lorsque vous créez un environnement, y compris les options que vous définissez explicitement, et les valeurs recommandées définies par le client. Vous pouvez également définir les options de configuration dans les configurations enregistrées et les fichiers de configuration. Si la même option figure dans plusieurs emplacements, la valeur utilisée est déterminée par l'ordre de priorité.

Paramètres d'option de configuration peuvent être composés au format texte et enregistrés avant la phase de création de l'environnement, appliqués au cours de la création de l'environnement à l'aide de n'importe quel client pris en charge et ajoutés, modifiés ou retirés après la création de l'environnement. Pour une description détaillée de toutes les méthodes disponibles pour l'utilisation des options de configuration à chacun de ces trois étapes, consultez les rubriques suivantes :

- Définition d'options de configuration avant la création de l'environnement
- Définition des options de configuration lors de la création de l'environnement
- Définition des options de configuration après la création de l'environnement

Pour obtenir une liste complète des espaces de noms et des options, y compris les valeurs par défaut et celles prises en charge pour chacun, veuillez consulter Options générales pour tous les environnements et Options spécifiques à une plateforme.

Priorité

Au cours de la création de l'environnement, des options de configuration sont appliquées à partir de plusieurs sources avec la priorité suivante, de la plus élevée à la plus faible :

- Paramètres appliqués directement à l'environnement : paramètres spécifiés lors d'une opération de création ou de mise à jour d'un environnement sur l'API Elastic Beanstalk par n'importe quel client, y compris la console Elastic Beanstalk, l'EB CLI et. AWS CLI SDKs La console Elastic Beanstalk et

l'interface de ligne de commande (CLI) EB appliquent également des valeurs recommandées pour certaines options qui correspondent à ce niveau, sauf en cas de remplacement.

- Configurations sauvegardées – Les paramètres pour toutes les options qui ne sont pas appliquées directement à l'environnement sont chargés à partir d'une configuration enregistrée, si spécifié.
- Fichiers de configuration (.ebextensions) – Les paramètres pour toutes les options qui ne sont pas appliquées directement à l'environnement et ne sont pas non plus précisées dans une configuration enregistrée, sont chargés à partir de fichiers de configuration dans le dossier `.ebextensions` à la racine du bundle de fichiers source de l'application.

  Les fichiers de configuration sont exécutés dans l'ordre alphabétique. Par exemple, `.ebextensions/01run.config` est exécuté avant `.ebextensions/02do.config`.
- Valeurs par défaut – Si une option de configuration a une valeur par défaut, elle ne s'applique que lorsque l'option n'est pas définie sur un des niveaux ci-dessus.

Si la même option de configuration est définie dans plusieurs emplacements, le paramètre ayant la plus haute priorité est appliqué. Lorsqu'un paramètre est appliqué à partir d'une configuration enregistrée ou de paramètres appliqués directement à l'environnement, le paramètre est stocké dans le cadre de la configuration de l'environnement. Ces paramètres peuvent être supprimés avec l' AWS CLI ou avec l'interface de ligne de commande (CLI) EB.

Les paramètres dans les fichiers de configuration ne sont pas appliqués directement à l'environnement et ne peuvent pas être supprimés sans modifier les fichiers de configuration ni déployer une nouvelle version de l'application. Si un paramètre appliqué avec l'une ou l'autre des méthodes est supprimé, ce même paramètre sera chargé à partir des fichiers de configuration dans le groupe source.

Imaginons par exemple que vous définissiez le nombre minimum d'instances dans votre environnement sur 5 lors de la création de l'environnement, à l'aide de la console Elastic Beanstalk, d'une option de ligne de commande ou d'une configuration sauvegardée. Le groupe source de votre application inclut également un fichier de configuration qui définit le nombre minimum d'instances sur 2.

Lorsque vous créez l'environnement, Elastic Beanstalk définit l'option `MinSize` dans l'espace de noms `aws:autoscaling:asg` sur 5. Si vous supprimez ensuite l'option de la configuration de l'environnement, la valeur dans le fichier de configuration est chargée, et le nombre minimum d'instances est défini sur 2. Si vous supprimez ensuite le fichier de configuration du bundle de fichiers source et que vous redéployez, Elastic Beanstalk utilise 1 comme paramètre par défaut.

Valeurs recommandées

L'interface de ligne de commande Elastic Beanstalk (EB CLI) et la console Elastic Beanstalk fournissent des valeurs recommandées pour certaines options de configuration. Ces valeurs peuvent être différentes des valeurs par défaut et sont définies au niveau de l'API lors de la création de votre environnement. Les valeurs recommandées permettent à Elastic Beanstalk d'améliorer la configuration de l'environnement par défaut sans procéder à des modifications précédentes incompatibles au niveau de l'API.

Par exemple, l'EB CLI et la console Elastic Beanstalk définissent l' EC2 option de configuration pour le `InstanceType` type d'instance (dans l'espace de noms). `aws:autoscaling:launchconfiguration` Chaque client fournit une manière différente de remplacer le paramètre par défaut. Dans la console, vous pouvez choisir un autre type d'instance à partir d'un menu déroulant sur la page Détails de configuration de l'assistant Créer un nouvel environnement. Avec l'interface de ligne de commande (CLI) EB, vous pouvez utiliser le paramètre `--instance_type` pour eb create.

Etant donné que les valeurs recommandées sont définies au niveau de l'API, elles remplaceront les valeurs pour les mêmes options que vous définissez dans les fichiers de configuration ou les configurations enregistrées. Les options suivantes sont définies :

Console Elastic Beanstalk

- Espace de nom : `aws:autoscaling:launchconfiguration`

  Noms d'options : `IamInstanceProfile`, `EC2KeyName`, `InstanceType`
- Espace de nom : `aws:autoscaling:updatepolicy:rollingupdate`

  Noms d'options : `RollingUpdateType` et `RollingUpdateEnabled`
- Espace de nom : `aws:elasticbeanstalk:application`

  Nom d'option: `Application Healthcheck URL`
- Espace de nom : `aws:elasticbeanstalk:command`

  Nom d'option : `DeploymentPolicy`, `BatchSize` et `BatchSizeType`
- Espace de nom : `aws:elasticbeanstalk:environment`

  Nom d'option: `ServiceRole`
- Espace de nom : `aws:elasticbeanstalk:healthreporting:system`

Nom d'option : `SystemType` et `HealthCheckSuccessThreshold`

- Espace de nom : `aws:elasticbeanstalk:sns:topics`

  Nom d'option: `Notification Endpoint`

- Espace de nom : `aws:elasticbeanstalk:sqsd`

  Nom d'option: `HttpConnections`

- Espace de nom : `aws:elb:loadbalancer`

  Nom d'option: `CrossZone`

- Espace de nom : `aws:elb:policies`

  Noms d'options : `ConnectionDrainingTimeout` et `ConnectionDrainingEnabled`

INTERFACE DE LIGNE DE COMMANDE (CLI) EB

- Espace de nom : `aws:autoscaling:launchconfiguration`

  Noms d'options : `IamInstanceProfile`, `InstanceType`

- Espace de nom : `aws:autoscaling:updatepolicy:rollingupdate`

  Noms d'options : `RollingUpdateType` et `RollingUpdateEnabled`

- Espace de nom : `aws:elasticbeanstalk:command`

  Nom d'option : `BatchSize` et `BatchSizeType`

- Espace de nom : `aws:elasticbeanstalk:environment`

  Nom d'option: `ServiceRole`

- Espace de nom : `aws:elasticbeanstalk:healthreporting:system`

  Nom d'option: `SystemType`

- Espace de nom : `aws:elb:loadbalancer`

  Nom d'option: `CrossZone`

- Espace de nom : `aws:elb:policies`

  Noms d'options : `ConnectionDrainingEnabled`

Définition d'options de configuration avant la création de l'environnement

AWS Elastic Beanstalk prend en charge un grand nombre d'options de configuration qui vous permettent de modifier les paramètres appliqués aux ressources de votre environnement. Plusieurs de ces options ont des valeurs par défaut qui peuvent être ignorées pour personnaliser votre environnement. D'autres options peuvent être configurées pour activer des fonctionnalités supplémentaires.

Elastic Beanstalk prend en charge deux méthodes d'enregistrement des paramètres des options de configuration. Les fichiers de configuration au format YAML ou JSON peuvent être inclus dans le code source de votre application dans un répertoire nommé `.ebextensions` et déployés dans le cadre du bundle de fichiers source de votre application. Vous créez et gérez des fichiers de configuration localement.

Les configurations enregistrées sont des modèles que vous créez à partir d'un environnement en cours d'exécution ou du fichier d'options JSON et que vous stockez dans Elastic Beanstalk. Les configurations existantes enregistrées peuvent être étendues pour créer une nouvelle configuration.

> **Note**
>
> Les paramètres définis dans les fichiers de configuration et les configurations enregistrées ont une priorité moindre que les paramètres configurés pendant ou après la création de l'environnement, y compris les valeurs recommandées appliquées par la console Elastic Beanstalk et l'interface de ligne de commande (CLI) EB. Consultez Priorité pour plus de détails.

Les options peuvent aussi être spécifiées dans un document JSON et transmises directement à Elastic Beanstalk lorsque vous créez ou mettez à jour un environnement avec l'interface de ligne de commande (CLI) EB ou l' AWS CLI. Les options fournies directement à Elastic Beanstalk de cette manière remplacent toutes les autres méthodes.

Pour obtenir une liste complète des options disponibles, veuillez consulter Options de configuration.

Méthodes

- Fichiers de configuration (.ebextensions)
- Configurations enregistrées
- Document JSON

- Configuration de l'interface de ligne de commande (CLI) EB

Fichiers de configuration (`.ebextensions`)

Utilisez `.ebextensions` pour configurer les options qui sont requises pour que votre application fonctionne et fournissez des valeurs par défaut pour d'autres options qui peuvent être remplacées à un niveau supérieur de priorité. Les options spécifiées dans `.ebextensions` ont le niveau de priorité le plus bas et sont remplacées par des paramètres de n'importe quel autre niveau.

Pour utiliser des fichiers de configuration, créez un dossier nommé `.ebextensions` au niveau supérieur du code source de votre projet. Ajoutez un fichier avec l'extension `.config` et spécifiez les options de la façon suivante :

```
option_settings:
  - namespace:  namespace
    option_name:  option name
    value:  option value
  - namespace:  namespace
    option_name:  option name
    value:  option value
```

Par exemple, le fichier de configuration suivant définit l'URL de vérification de l'état de l'application sur `/health`:

`healthcheckurl.config`

```
option_settings:
  - namespace:  aws:elasticbeanstalk:application
    option_name:  Application Healthcheck URL
    value:  /health
```

Dans JSON :

```
{
 "option_settings" :
    [
      {
        "namespace" : "aws:elasticbeanstalk:application",
        "option_name" : "Application Healthcheck URL",
```

```
        "value" : "/health"
      }
    ]
}
```

Cela configure l'équilibreur de charge Elastic Load Balancing dans votre environnement Elastic Beanstalk pour effectuer une requête HTTP vers le chemin d'accès `/health` à chaque instance EC2 afin de déterminer si elle est saine ou non.

> **Note**
>
> YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Incluez le répertoire `.ebextensions` dans le bundle de fichiers source de l'application et déployez-le dans un environnement Elastic Beanstalk nouveau ou existant.

Les fichiers de configuration prennent en charge plusieurs sections en plus des `option_settings` pour personnaliser le logiciel et les fichiers qui s'exécutent sur les serveurs dans votre environnement. Pour plus d'informations, consultez .Ebextensions.

Configurations enregistrées

Créez une configuration enregistrée pour enregistrer des paramètres que vous avez appliqués à un environnement existant pendant ou après la création de l'environnement à l'aide de la console Elastic Beanstalk, de l'interface de ligne de commande (CLI) EB ou de l' AWS CLI. Les configurations enregistrées appartiennent à une application et peuvent être appliquées aux environnements nouveaux ou existants pour cette application.

Clients

- Console Elastic Beanstalk
- INTERFACE DE LIGNE DE COMMANDE (CLI) EB
- AWS CLI

Console Elastic Beanstalk

Pour créer une configuration enregistrée (console Elastic Beanstalk)

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions, (Actions) puis Save configuration (Enregistrer la configuration).
4. Utilisez la boîte de dialogue à l'écran pour terminer l'action.

Les configurations enregistrées sont stockées dans le compartiment S3 Elastic Beanstalk, dans un dossier nommé en fonction de votre application. Par exemple, les configurations pour une application nommée `my-app` dans la région us-west-2 pour le compte n° 123456789012 sont disponibles à l'emplacement `s3://elasticbeanstalk-us-west-2-123456789012/resources/templates/my-app`.

INTERFACE DE LIGNE DE COMMANDE (CLI) EB

L'interface de ligne de commande (CLI) EB fournit également des sous-commandes permettant d'interagir avec des configurations enregistrées sous eb config :

Pour créer une configuration enregistrée (CLI EB)

1. Enregistrez la configuration actuelle de l'environnement attaché :

   ```
   ~/project$ eb config save --cfg my-app-v1
   ```

   L'interface de ligne de commande (CLI) EB enregistre la configuration dans `~/project/.elasticbeanstalk/saved_configs/my-app-v1.cfg.yml`
2. Modifiez la configuration sauvegardée localement si nécessaire.
3. Chargez la configuration enregistrée dans S3 :

   ```
   ~/project$ eb config put my-app-v1
   ```

AWS CLI

Créez une configuration enregistrée à partir d'un environnement en cours d'exécution avec `aws elasticbeanstalk create-configuration-template`

Pour créer une configuration enregistrée (AWS CLI)

1. Identifiez l'ID d'environnement de votre environnement Elastic Beanstalk avec `describe-environments`:

```
$ aws elasticbeanstalk describe-environments --environment-name my-env
{
    "Environments": [
        {
            "ApplicationName": "my-env",
            "EnvironmentName": "my-env",
            "VersionLabel": "89df",
            "Status": "Ready",
            "Description": "Environment created from the EB CLI using \"eb create
\"",
            "EnvironmentId": "e-vcghmm2zwk",
            "EndpointURL": "awseb-e-v-AWSEBLoa-1JUM8159RA11M-43V6ZI1194.us-
west-2.elb.amazonaws.com",
            "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.2 running Multi-
container Docker 1.7.1 (Generic)",
            "CNAME": "my-env-nfptuqaper.elasticbeanstalk.com",
            "Health": "Green",
            "AbortableOperationInProgress": false,
            "Tier": {
                "Version": " ",
                "Type": "Standard",
                "Name": "WebServer"
            },
            "HealthStatus": "Ok",
            "DateUpdated": "2015-10-01T00:24:04.045Z",
            "DateCreated": "2015-09-30T23:27:55.768Z"
        }
    ]
}
```

2. Enregistrez la configuration actuelle de l'environnement avec `create-configuration-template`:

```
$ aws elasticbeanstalk create-configuration-template --environment-id e-vcghmm2zwk
 --application-name my-app --template-name v1
```

Elastic Beanstalk enregistre la configuration dans votre compartiment Elastic Beanstalk dans Amazon S3.

Document JSON

Si vous utilisez le AWS CLI pour créer et mettre à jour des environnements, vous pouvez également fournir des options de configuration au format JSON. Une bibliothèque de fichiers de configuration au format JSON est utile si vous utilisez le AWS CLI pour créer et gérer des environnements.

Par exemple, les ensembles de documents JSON suivants définissent l'URL d'intégrité de l'application sur `/health`:

~/ebconfigs/healthcheckurl.json

```
[
  {
    "Namespace": "aws:elasticbeanstalk:application",
    "OptionName": "Application Healthcheck URL",
    "Value": "/health"
  }
]
```

Configuration de l'interface de ligne de commande (CLI) EB

En plus de prendre en charge les configurations enregistrées et la configuration de l'environnement direct avec les commandes eb config, l'interface de ligne de commande (CLI) EB a un fichier de configuration avec une option nommée `default_ec2_keyname` que vous pouvez utiliser pour spécifier une paire de clés Amazon EC2 pour l'accès SSH aux instances dans votre environnement. L'interface de ligne de commande (CLI) EB utilise cette option pour définir l'option de configuration `EC2KeyName` dans l'espace de noms `aws:autoscaling:launchconfiguration`.

~/ .yml workspace/my-app/.elasticbeanstalk/config

```
branch-defaults:
  master:
```

```
    environment: my-env
  develop:
    environment: my-env-dev
deploy:
  artifact: ROOT.war
global:
  application_name: my-app
  default_ec2_keyname: my-keypair
  default_platform: Tomcat 8 Java 8
  default_region: us-west-2
  profile: null
  sc: git
```

Définition des options de configuration lors de la création de l'environnement

Lorsque vous créez un AWS Elastic Beanstalk environnement à l'aide de la console Elastic Beanstalk, de l'EB AWS CLI CLI, d'un SDK ou de l'API Elastic Beanstalk, vous pouvez fournir des valeurs pour les options de configuration afin de personnaliser votre environnement et les ressources qui y sont lancées. AWS

Pour toute autre action qu'une modification ponctuelle de la configuration, vous pouvez stocker les fichiers de configuration localement, dans votre bundle de fichiers source ou dans Amazon S3.

Cette rubrique inclut les procédures relatives à toutes les méthodes de définition des options de configuration lors de la création de l'environnement.

Clients

- Dans la console Elastic Beanstalk.
- Utilisation de l'interface de ligne de commande (CLI) EB
- À l'aide du AWS CLI

Dans la console Elastic Beanstalk.

Lorsque vous créez un environnement Elastic Beanstalk dans la console Elastic Beanstalk, vous pouvez fournir des options de configuration à l'aide de fichiers de configuration, de configurations enregistrées et de formulaires dans l'assistant Create New Environment (Créer un nouvel environnement).

Méthodes

- Utilisation des fichiers de configuration (.ebextensions)
- Utilisation d'une configuration enregistrée
- Utilisation de l'assistant de création d'un environnement

Utilisation des fichiers de configuration (`.ebextensions`)

Incluez les fichiers `.config` dans le bundle source de votre application dans un dossier nommé `.ebextensions`.

Pour de plus amples informations sur les fichiers de configuration, veuillez consulter .Ebextensions.

```
~/workspace/my-app-v1.zip
|-- .ebextensions
|   |-- environmentvariables.config
|   `-- healthcheckurl.config
|-- index.php
`-- styles.css
```

Téléchargez le bundle de fichiers source vers Elastic Beanstalk normalement, lors de la création de l'environnement.

La console Elastic Beanstalk applique des valeurs recommandées pour certaines options de configuration et inclut des champs de formulaire pour les autres. Les options configurées par la console Elastic Beanstalk sont appliquées directement à l'environnement et remplacent les paramètres dans les fichiers de configuration.

Utilisation d'une configuration enregistrée

Lorsque vous créez un environnement à l'aide de la console Elastic Beanstalk, l'une des premières étapes consiste à choisir une configuration. Il peut s'agir d'une configuration prédéfinie, qui correspond généralement à la dernière version d'une plateforme telle que PHP ou Tomcat, ou d'une configuration enregistrée.

Pour appliquer une configuration enregistrée lors de la création de l'environnement (console Elastic Beanstalk)

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS

2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.
3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Saved configurations (Configurations enregistrées).
4. Sélectionnez la configuration enregistrée que vous souhaitez appliquer, puis choisissez Launch environment (Lancer l'environnement).
5. Suivez les étapes de l'assistant pour créer votre environnement.

Les configurations enregistrées sont spécifiques aux applications. Pour de plus amples informations sur la création de configurations enregistrées, veuillez consulter Configurations enregistrées.

Utilisation de l'assistant de création d'un environnement

La plupart des options de configuration standards sont présentées dans les pages Configurer plus d'options de l'assistant Création d'un environnement. Si vous créez une base de données Amazon RDS ou configurez un VPC pour votre environnement, des options de configuration supplémentaires vous sont proposées pour ces ressources.

Pour définir les options de configuration lors de la création de l'environnement (console Elastic Beanstalk)

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le volet de navigation, choisissez Applications.
3. Choisissez ou créez une application.
4. Choisissez Actions, puis Create environment (Créer un environnement).
5. Suivez les étapes de l'assistant, puis choisissez Configurer plus d'options.
6. Choisissez l'un des configuration presets (préréglages de la configuration), puis choisissez Edit (Modifier) dans une ou plusieurs catégories de configuration pour modifier un groupe d'options de configuration associées.
7. Une fois que vous avez sélectionné toutes les options souhaitées, choisissez Créer un environnement.

Toutes les options que vous définissez dans l'assistant de création d'un environnement sont directement définies dans l'environnement. Elles remplacent les paramètres d'option figurant dans les configurations enregistrées ou dans les fichiers de configuration (`.ebextensions`) que

vous appliquez. Vous pouvez supprimer des paramètres après la création de l'environnement, via l'interface de ligne de commande (CLI) EB ou l'AWS CLI afin d'autoriser l'affichage des paramètres dans les configurations enregistrées ou les fichiers de configuration.

Pour plus d'informations sur l'assistant de création d'environnement, reportez-vous à la section Assistant de création d'un environnement.

Utilisation de l'interface de ligne de commande (CLI) EB

Méthodes

- Utilisation des fichiers de configuration (.ebextensions)
- Utilisations de configurations enregistrées
- Utilisation d'options de ligne de commande

Utilisation des fichiers de configuration (`.ebextensions`)

Incluez les fichiers `.config` dans votre dossier de projet sous `.ebextensions` afin de les déployer avec votre code d'application.

Pour de plus amples informations sur les fichiers de configuration, veuillez consulter .Ebextensions.

```
~/workspace/my-app/
|-- .ebextensions
|   |-- environmentvariables.config
|   `-- healthcheckurl.config
|-- .elasticbeanstalk
|   `-- config.yml
|-- index.php
`-- styles.css
```

Créez votre environnement et déployez votre code source dans ce dernier à l'aide de la commande eb create.

```
~/workspace/my-app$ eb create my-env
```

Utilisations de configurations enregistrées

Pour appliquer une configuration enregistrée lorsque vous créez un environnement via eb create, utilisez l'option `--cfg`.

```
~/workspace/my-app$ eb create --cfg savedconfig
```

Vous pouvez stocker la configuration enregistrée dans votre dossier de projet ou dans votre emplacement de stockage Elastic Beanstalk sur Amazon S3. Dans l'exemple ci-avant, l'interface de ligne de commande (CLI) EB commence par rechercher un fichier de configuration enregistré nommé `savedconfig.cfg.yml` dans le dossier `.elasticbeanstalk/saved_configs/`. N'incluez pas les extensions des noms de fichier (`.cfg.yml`) lorsque vous appliquez une configuration enregistrée avec `--cfg`.

```
~/workspace/my-app/
|-- .ebextensions
|   `-- healthcheckurl.config
|-- .elasticbeanstalk
|   |-- saved_configs
|   |   `-- savedconfig.cfg.yml
|   `-- config.yml
|-- index.php
`-- styles.css
```

Si l'interface de ligne de commande (CLI) EB ne trouve pas la configuration localement, elle cherche dans l'emplacement de stockage Elastic Beanstalk dans Amazon S3. Pour de plus amples informations sur la création, la modification et le téléchargement des configurations enregistrées, veuillez consulter Configurations enregistrées.

Utilisation d'options de ligne de commande

La commande eb create de l'interface de ligne de commande (CLI) EB comporte plusieurs options, qui vous permettent de définir des options de configuration lors de la création de l'environnement. Vous pouvez utiliser ces options pour ajouter une base de données RDS à votre environnement, configurer un VPC ou remplacer des valeurs recommandées.

Par exemple, l'interface de ligne de commande (CLI) EB utilise le type d'instance `t2.micro` par défaut. Pour choisir un autre type d'instance, utilisez l'option `--instance_type`.

```
$ eb create my-env --instance_type t2.medium
```

Pour créer une instance de base de données Amazon RDS et l'associer à votre environnement, utilisez les options `--database` :

```
$ eb create --database.engine postgres --database.username dbuser
```

Si vous n'indiquez pas le nom de l'environnement, le mot de passe de la base de données ou tout autre paramètre requis pour créer votre environnement, l'interface de ligne de commande (CLI) EB vous invite à saisir ces données.

Consultez eb create pour obtenir la liste complète des options disponibles et des exemples d'utilisation.

À l'aide du AWS CLI

Lorsque vous utilisez la `create-environment` commande pour créer un environnement Elastic Beanstalk AWS CLI avec le AWS CLI , aucune valeur recommandée n'est appliquée. Toutes les options de configuration sont définies dans les fichiers de configuration du bundle source que vous spécifiez.

Méthodes

- Utilisation des fichiers de configuration (.ebextensions)
- Utilisation d'une configuration enregistrée
- Utilisation d'options de ligne de commande

Utilisation des fichiers de configuration (**`.ebextensions`**)

Pour appliquer des fichiers de configuration à un environnement que vous créez avec le AWS CLI, incluez-les dans le bundle de sources d'applications que vous chargez sur Amazon S3.

Pour de plus amples informations sur les fichiers de configuration, veuillez consulter .Ebextensions.

```
~/workspace/my-app-v1.zip
|-- .ebextensions
|   |-- environmentvariables.config
|   `-- healthcheckurl.config
|-- index.php
`-- styles.css
```

Pour télécharger un bundle de sources d'applications et créer un environnement avec AWS CLI

1. Si vous n'avez pas encore de compartiment Elastic Beanstalk dans Amazon S3, créez-en un avec `create-storage-location`.

```
$ aws elasticbeanstalk create-storage-location
{
    "S3Bucket": "elasticbeanstalk-us-west-2-123456789012"
}
```

2. Téléchargez votre bundle de fichiers source d'application sur Amazon S3.

```
$ aws s3 cp sourcebundle.zip s3://elasticbeanstalk-us-west-2-123456789012/my-app/
sourcebundle.zip
```

3. Créez la version de l'application.

```
$ aws elasticbeanstalk create-application-version --application-name my-app --
version-label v1 --description MyAppv1 --source-bundle S3Bucket="elasticbeanstalk-
us-west-2-123456789012",S3Key="my-app/sourcebundle.zip" --auto-create-application
```

4. Création de l'environnement

```
$ aws elasticbeanstalk create-environment --application-name my-app --environment-
name my-env --version-label v1 --solution-stack-name "64bit Amazon Linux 2015.03
 v2.0.0 running Tomcat 8 Java 8"
```

Utilisation d'une configuration enregistrée

Pour appliquer une configuration enregistrée à un environnement lors de la création, utilisez le paramètre `--template-name`.

```
$ aws elasticbeanstalk create-environment --application-name my-app --environment-name
 my-env --template-name savedconfig --version-label v1
```

Lorsque vous spécifiez une configuration enregistrée, ne spécifiez pas un nom de pile de solutions en plus. Les configurations enregistrées spécifient déjà une pile de solutions, et Elastic Beanstalk affichera une erreur si vous essayez d'utiliser les deux options.

Utilisation d'options de ligne de commande

Utilisez le paramètre `--option-settings` pour spécifier les options de configuration au format JSON.

```
$ aws elasticbeanstalk create-environment --application-name my-app --environment-name
 my-env --version-label v1 --template-name savedconfig --option-settings '[
  {
    "Namespace": "aws:elasticbeanstalk:application",
    "OptionName": "Application Healthcheck URL",
    "Value": "/health"
  }
]
```

Pour charger le JSON à partir d'un fichier, utilisez le préfixe `file://`.

```
$ aws elasticbeanstalk create-environment --application-name my-app --environment-
name my-env --version-label v1 --template-name savedconfig --option-settings file://
healthcheckurl.json
```

Elastic Beanstalk applique directement les paramètres d'option que vous spécifiez via l'option `--option-settings` dans votre environnement. Si les mêmes options sont spécifiées dans une configuration enregistrée ou un fichier de configuration, `--option-settings` remplace ces valeurs.

Définition des options de configuration après la création de l'environnement

Vous pouvez modifier les paramètres d'option dans un environnement en cours d'exécution en appliquant des configurations enregistrées, en chargeant un nouveau bundle de fichiers source à l'aide des fichiers de configuration (`.ebextensions`) ou un document JSON. L'interface de ligne de commande (CLI) EB et la console Elastic Beanstalk ont également des fonctionnalités spécifiques client pour définir et mettre à jour des options de configuration.

Lorsque vous définissez ou modifiez une option de configuration, vous pouvez déclencher une mise à jour environnement complet, selon la gravité du changement. Par exemple, des modifications apportées aux options dans `aws:autoscaling:launchconfiguration`, comme `InstanceType`, nécessitent que les instances Amazon EC2 dans votre environnement soient réapprovisionnées. Cela déclenche une mise à jour propagée. D'autres changements de configuration peuvent être appliqués sans interruption ou réapprovisionnement.

Vous pouvez supprimer les paramètres des options d'un environnement à l'aide de la CLI ou de AWS CLI commandes EB. La suppression d'une option qui a été définie directement sur un environnement à un niveau d'API autorise des paramètres dans des fichiers de configuration, qui sont autrement masqués par des paramètres appliqués directement dans un environnement, à s'exposer et à prendre effet.

Vous pouvez remplacer les paramètres des configurations enregistrées et des fichiers de configuration en définissant la même option directement au niveau de l'environnement à l'aide de l'une des autres méthodes de configuration. Toutefois, le seul moyen de supprimer complètement ces paramètres consiste à appliquer une nouvelle configuration enregistrée ou un nouveau fichier de configuration. Lorsqu'une option n'est pas définie dans une configuration enregistrée, dans le fichier de configuration, ou directement sur un environnement, la valeur par défaut s'applique, s'il y en a une. Consultez Priorité pour plus de détails.

Clients

- La console Elastic Beanstalk.
- Interface de ligne de commande (CLI) EB
- Le AWS CLI

La console Elastic Beanstalk.

Vous pouvez mettre à jour des paramètres d'option de configuration dans la console Elastic Beanstalk en déployant un bundle de fichiers source d'application qui contient des fichiers de configuration, en appliquant une configuration enregistrée ou en modifiant l'environnement directement avec la page Configuration dans la console de gestion de l'environnement.

Méthodes

- Utilisation des fichiers de configuration (.ebextensions)
- Utilisation d'une configuration enregistrée
- Utilisation de la console Elastic Beanstalk

Utilisation des fichiers de configuration (`.ebextensions`)

Mettez à jour les fichiers de configuration dans votre répertoire source, créez un bundle de fichiers source et déployez la nouvelle version dans votre environnement Elastic Beanstalk pour appliquer les modifications.

Pour de plus amples informations sur les fichiers de configuration, veuillez consulter .Ebextensions.

Pour déployer un groupe source

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

Les modifications apportées aux fichiers de configuration ne remplaceront pas les paramètres d'options dans les configurations enregistrées ou les paramètres appliqués directement à l'environnement au niveau de l'API. Pour plus d'informations, consultez Precedence.

Utilisation d'une configuration enregistrée

Appliquez une configuration enregistrée à un environnement en cours d'exécution pour appliquer des paramètres d'option qu'il définit.

Pour appliquer une configuration enregistrée à un environnement en cours d'exécution (console Elastic Beanstalk)

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.
3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Saved configurations (Configurations enregistrées).
4. Sélectionnez la configuration enregistrée à appliquer, puis choisissez Load (Charger).
5. Sélectionnez un environnement, puis Charge.

Les paramètres définis dans une configuration enregistrée remplacent les paramètres dans les fichiers de configuration et sont remplacés par les paramètres configurés à l'aide de la console de gestion d'environnement.

Pour de plus amples informations sur la création de configurations enregistrées, veuillez consulter Configurations enregistrées.

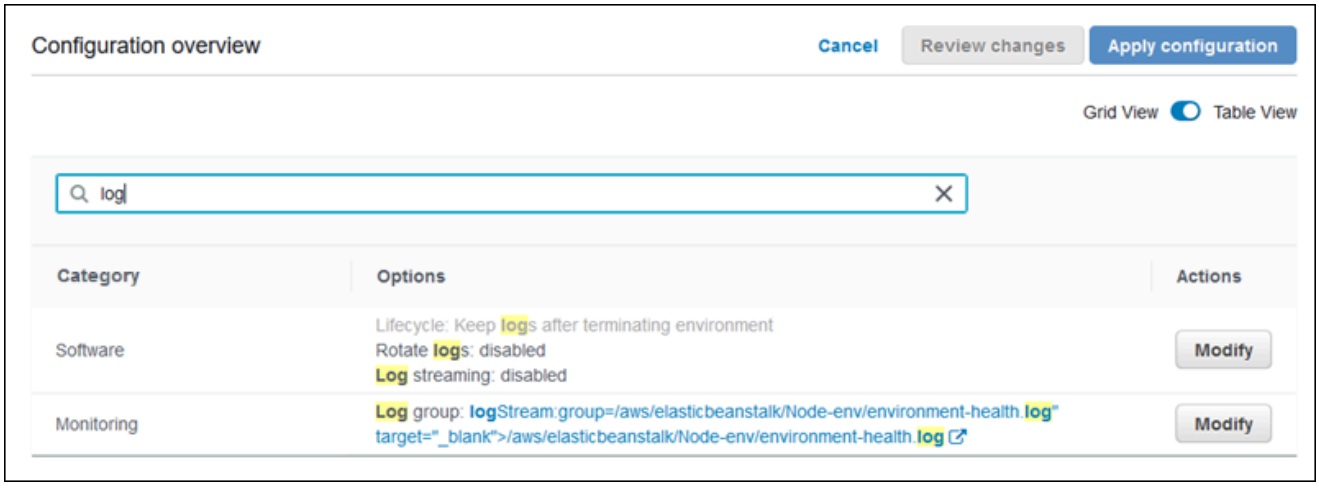
Utilisation de la console Elastic Beanstalk

La console Elastic Beanstalk présente beaucoup d'options de configuration sur la page Configuration (Configuration) de chaque environnement.

Pour modifier les options de configuration dans un environnement en cours d'exécution (console Elastic Beanstalk)

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Recherchez la page de configuration que vous souhaitez modifier :
   - Si vous voyez l'option qui vous intéresse, ou que vous savez à quelle catégorie de configuration elle appartient, choisissez Edit (Modifier) dans sa catégorie de configuration.
   - Pour rechercher une option, activez la Table View (Vue Table), puis saisissez vos termes de recherche dans la zone de recherche. À mesure que vous saisissez du texte, la liste se réduit et affiche uniquement les options qui correspondent à vos termes de recherche.

     Lorsque vous voyez l'option que vous recherchez, choisissez Edit (Modifier) dans la catégorie de configuration qui la contient.



5. Modifiez les paramètres, puis choisissez Enregistrer.
6. Si vous souhaitez modifier d'autres catégories de configuration, répétez les deux étapes précédentes.
7. Choisissez Apply.

Les modifications apportées aux options de configuration dans la console de gestion de l'environnement sont directement appliquées à l'environnement. Ces modifications remplacent les paramètres pour les mêmes options dans les fichiers de configuration ou dans les configurations enregistrées. Pour plus d'informations, consultez Precedence (Priorité).

Pour de plus amples informations sur la modification des options de configuration au niveau d'un environnement en cours d'exécution à l'aide de la console Elastic Beanstalk, veuillez consulter les rubriques sous Configuration d'environnements Elastic Beanstalk.

Interface de ligne de commande (CLI) EB

Vous pouvez mettre à jour les paramètres d'option de configuration avec l'interface de ligne de commande (CLI) EB en déployant le code source qui contient des fichiers de configuration, en appliquant des paramètres à partir d'une configuration enregistrée ou en modifiant la configuration de l'environnement directement avec la commande eb config.

Méthodes

- Utilisation des fichiers de configuration (.ebextensions)
- Utilisation d'une configuration enregistrée
- Utiliser eb config
- Utiliser eb setenv

Utilisation des fichiers de configuration (`.ebextensions`)

Incluez les fichiers `.config` dans votre dossier de projet sous `.ebextensions` afin de les déployer avec votre code d'application.

Pour de plus amples informations sur les fichiers de configuration, veuillez consulter .Ebextensions.

```
~/workspace/my-app/
|-- .ebextensions
|   |-- environmentvariables.config
|   `-- healthcheckurl.config
|-- .elasticbeanstalk
|   `-- config.yml
|-- index.php
`-- styles.css
```

Déployez votre code source avec eb deploy.

```
~/workspace/my-app$ eb deploy
```

Utilisation d'une configuration enregistrée

Vous pouvez utiliser la commande eb config pour appliquer une configuration enregistrée à un environnement en cours d'exécution. Utilisez l'option `--cfg` avec le nom de la configuration enregistrée pour appliquer ses paramètres à votre environnement.

```
$ eb config --cfg v1
```

Dans cet exemple, v1 est le nom d'un fichier de configuration enregistré précédemment créé.

Les paramètres appliqués à un environnement avec cette commande remplacent les paramètres qui ont été appliqués au cours de la création de l'environnement et les paramètres définis dans les fichiers de configuration dans le groupe source de votre application.

Utiliser eb config

La commande eb config de l'interface de ligne de commande (CLI) EB vous permet de définir et de supprimer des paramètres d'option directement dans un environnement à l'aide d'un éditeur de texte.

Lorsque vous exécutez eb config, l'interface de ligne de commande (CLI) EB affiche les paramètres appliqués à votre environnement depuis toutes les sources, y compris les fichiers de configuration, les configurations enregistrées, les valeurs recommandées, le jeu d'options directement sur l'environnement et les valeurs par défaut d'API.

> **Note**
>
> eb config n'affiche pas les propriétés de l'environnement. Pour définir les propriétés de l'environnement que vous pouvez lire à partir de votre application, utilisez eb setenv.

L'exemple suivant montre les paramètres appliqués dans l'espace de noms `aws:autoscaling:launchconfiguration`. Ces paramètres sont les suivants :

- Deux valeurs recommandées, pour `IamInstanceProfile` et `InstanceType`, appliquées par l'interface de ligne de commande (CLI) EB lors de la création de l'environnement.
- L'option `EC2KeyName`, définie directement sur l'environnement lors de la création selon la configuration du référentiel.

- Les valeurs d'API par défaut pour les autres options.

```
ApplicationName: tomcat
DateUpdated: 2015-09-30 22:51:07+00:00
EnvironmentName: tomcat
SolutionStackName: 64bit Amazon Linux 2015.03 v2.0.1 running Tomcat 8 Java 8
settings:
...
aws:autoscaling:launchconfiguration:
    BlockDeviceMappings: null
    EC2KeyName: my-key
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role
    ImageId: ami-1f316660
    InstanceType: t2.micro
...
```

Pour définir ou modifier les options de configuration avec eb config

1. Exécutez eb config pour afficher la configuration de votre environnement.

   ```
   ~/workspace/my-app/$ eb config
   ```

2. Modifiez toutes valeurs de paramètre à l'aide de l'éditeur de texte par défaut.

   ```
   aws:autoscaling:launchconfiguration:
       BlockDeviceMappings: null
       EC2KeyName: my-key
       IamInstanceProfile: aws-elasticbeanstalk-ec2-role
       ImageId: ami-1f316660
       InstanceType: t2.medium
   ```

3. Enregistrez le fichier de configuration temporaire et quittez.
4. L'interface de ligne de commande (CLI) EB met à jour la configuration de votre environnement.

Définir des options de configuration avec eb config remplace les paramètres de toutes les autres sources.

Vous pouvez également supprimer des options à partir de votre environnement avec eb config.

Pour supprimer des options de configuration (CLI EB)

1. Exécutez eb config pour afficher la configuration de votre environnement.

```
~/workspace/my-app/$ eb config
```

2. Remplacez toute valeur indiquée par la chaîne `null`. Vous pouvez aussi supprimer la ligne complète contenant l'option que vous souhaitez supprimer.

```
aws:autoscaling:launchconfiguration:
    BlockDeviceMappings: null
    EC2KeyName: my-key
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role
    ImageId: ami-1f316660
    InstanceType: null
```

3. Enregistrez le fichier de configuration temporaire et quittez.
4. L'interface de ligne de commande (CLI) EB met à jour la configuration de votre environnement.

Supprimer des options de votre environnement avec eb config permet aux paramètres pour les mêmes options de s'exposer depuis les fichiers de configuration dans le groupe source de votre application. Pour plus d'informations, consultez Precedence.

Utiliser eb setenv

Pour définir des propriétés de l'environnement avec l'interface de ligne de commande (CLI) EB, utilisez eb setenv.

```
~/workspace/my-app/$ eb setenv ENVVAR=TEST
INFO: Environment update is starting.
INFO: Updating environment my-env's configuration settings.
INFO: Environment health has transitioned from Ok to Info. Command is executing on all
 instances.
INFO: Successfully deployed new configuration to environment.
```

Cette commande définit les propriétés de l'environnement dans l'`aws:elasticbeanstalk:application:environment`espace de noms. Les propriétés d'environnement définies avec eb setenv sont disponibles pour votre application après un court processus de mise à jour.

Affichez les propriétés de l'environnement définies sur votre environnement avec eb printenv.

```
~/workspace/my-app/$ eb printenv
 Environment Variables:
     ENVVAR = TEST
```

Le AWS CLI

Vous pouvez mettre à jour les paramètres des options de configuration avec le AWS CLI en déployant un bundle source contenant des fichiers de configuration, en appliquant une configuration enregistrée à distance ou en modifiant l'environnement directement à l'aide de la `aws elasticbeanstalk update-environment` commande.

Méthodes

- Utilisation des fichiers de configuration (.ebextensions)
- Utilisation d'une configuration enregistrée
- Utilisation d'options de ligne de commande

Utilisation des fichiers de configuration (`.ebextensions`)

Pour appliquer des fichiers de configuration à un environnement d'exécution avec le AWS CLI, incluez-les dans le bundle de sources d'applications que vous chargez sur Amazon S3.

Pour de plus amples informations sur les fichiers de configuration, veuillez consulter .Ebextensions.

```
~/workspace/my-app-v1.zip
|-- .ebextensions
|   |-- environmentvariables.config
|   `-- healthcheckurl.config
|-- index.php
`-- styles.css
```

Pour télécharger un bundle source d'application et l'appliquer à un environnement en cours d'exécution (AWS CLI)

1. Si vous n'avez pas encore de compartiment Elastic Beanstalk dans Amazon S3, créez-en un avec `create-storage-location`:

   ```
   $ aws elasticbeanstalk create-storage-location
   {
   ```

```
    "S3Bucket": "elasticbeanstalk-us-west-2-123456789012"
}
```

2. Téléchargez votre bundle de fichiers source d'application sur Amazon S3.

```
$ aws s3 cp sourcebundlev2.zip s3://elasticbeanstalk-us-west-2-123456789012/my-app/
sourcebundlev2.zip
```

3. Créez la version de l'application.

```
$ aws elasticbeanstalk create-application-version --application-
name my-app --version-label v2 --description MyAppv2 --source-bundle
 S3Bucket="elasticbeanstalk-us-west-2-123456789012",S3Key="my-app/
sourcebundlev2.zip"
```

4. Mettez à jour l'environnement.

```
$ aws elasticbeanstalk update-environment --environment-name my-env --version-label
 v2
```

Utilisation d'une configuration enregistrée

Vous pouvez appliquer une configuration enregistrée dans un environnement en cours d'exécution avec l'option `--template-name` sur la commande `aws elasticbeanstalk update-environment`.

La configuration enregistrée doit être dans votre compartiment Elastic Beanstalk dans un chemin d'accès nommé en fonction de votre application sous `resources/templates`. Par exemple, le modèle `v1` de l'application `my-app` dans la région USA Ouest (Oregon) (us-west-2) pour le compte n° 123456789012 est disponible à l'emplacement `s3://elasticbeanstalk-us-west-2-123456789012/resources/templates/my-app/v1`.

Pour appliquer une configuration enregistrée à un environnement en cours d'exécution (AWS CLI)

- Spécifiez la configuration enregistrée dans un appel `update-environment` avec l'option `--template-name`.

```
$ aws elasticbeanstalk update-environment --environment-name my-env --template-
name v1
```

Elastic Beanstalk place les configurations enregistrées dans cet emplacement lorsque vous les créez avec `aws elasticbeanstalk create-configuration-template`. Vous pouvez également modifier les configurations enregistrées localement et les mettre à cet emplacement vous-même.

Utilisation d'options de ligne de commande

Pour modifier des options de configuration avec un document JSON (AWS CLI)

1. Définissez vos paramètres d'options au format JSON dans un fichier local.
2. Exécutez `update-environment` avec l'option `--option-settings`.

```
$ aws elasticbeanstalk update-environment --environment-name my-env --option-
settings file://~/ebconfigs/as-zero.json
```

Dans cet exemple, `as-zero.json` définit des options qui configurent l'environnement avec un minimum et un maximum de zéro instance. Cela interrompt les instances dans l'environnement sans mise hors service de ce dernier.

~/ebconfigs/as-zero.json

```
[
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "MinSize",
        "Value": "0"
    },
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "MaxSize",
        "Value": "0"
    },
    {
        "Namespace": "aws:autoscaling:updatepolicy:rollingupdate",
        "OptionName": "RollingUpdateEnabled",
        "Value": "false"
    }
]
```

> **Note**
>
> Définir des options de configuration avec `update-environment` remplace les paramètres de toutes les autres sources.

Vous pouvez également supprimer des options à partir de votre environnement avec `update-environment`.

Pour supprimer des options de configuration (AWS CLI)

- Exécutez la commande `update-environment` avec l'option `--options-to-remove`.

```
$ aws elasticbeanstalk update-environment --environment-name my-env --options-to-
remove Namespace=aws:autoscaling:launchconfiguration,OptionName=InstanceType
```

Supprimer des options de votre environnement avec `update-environment` permet aux paramètres pour les mêmes options de s'exposer depuis les fichiers de configuration dans le groupe source de votre application. Si une option n'est pas configurée à l'aide d'une de ces méthodes, la valeur par défaut de l'API s'applique, le cas échéant. Pour plus d'informations, consultez Precedence.

Options générales pour tous les environnements

Espaces de noms

- aws:autoscaling:asg
- aws:autoscaling:launchconfiguration
- aws:autoscaling:scheduledaction
- aws:autoscaling:trigger
- aws:autoscaling:updatepolicy:rollingupdate
- aws:ec2:instances
- aws:ec2:vpc
- aws:elasticbeanstalk:application
- aws:elasticbeanstalk:application:environment
- patte : tige de haricot élastique : application : secrets environnementaux
- aws:elasticbeanstalk:cloudwatch:logs

- aws:elasticbeanstalk:cloudwatch:logs:health
- aws:elasticbeanstalk:command
- aws:elasticbeanstalk:environment
- aws:elasticbeanstalk:environment:process:default
- aws:elasticbeanstalk:environment:process:process_name
- aws:elasticbeanstalk:environment:proxy:staticfiles
- aws:elasticbeanstalk:healthreporting:system
- aws:elasticbeanstalk:hostmanager
- aws:elasticbeanstalk:managedactions
- aws:elasticbeanstalk:managedactions:platformupdate
- aws:elasticbeanstalk:monitoring
- aws:elasticbeanstalk:sns:topics
- aws:elasticbeanstalk:sqsd
- aws:elasticbeanstalk:trafficsplitting
- aws:elasticbeanstalk:xray
- aws:elb:healthcheck
- aws:elb:loadbalancer
- aws:elb:listener
- aws:elb:listener:listener_port
- aws:elb:policies
- aws:elb:policies:policy_name
- aws:elbv2:listener:default
- aws:elbv2:listener:listener_port
- aws:elbv2:listenerrule:rule_name
- aws:elbv2:loadbalancer
- aws:rds:dbinstance

aws:autoscaling:asg

Configurez le groupe Auto Scaling de votre environnement. Pour plus d'informations, consultez the section called “Groupe Auto Scaling”.

Espace de noms : **`aws:autoscaling:asg`**

Nom	Description	Par défaut	Valeurs valides

Availability Zones	Les zones de disponibilité (AZs) sont des emplacements distincts au sein d'une AWS région conçus pour être isolés des défaillances survenant dans d'autres régions AZs. Ils fournissent une connectivité réseau peu coûteuse et à faible latence AZs à d'autres utilisateurs de la même région. Choisissez le nombre de AZs pour vos instances.	`Any`	`Any`
			`Any 1`
			`Any 2`
			`Any 3`

Cooldown	Le temps de stabilisation permet d'empêcher Amazon EC2 Auto Scaling de lancer d'autres activités de mise à l'échelle avant que les effets des activités précédentes soient visibles. Un temps de stabilisation est la durée, en secondes, qui doit s'écouler entre la fin d'une activité de dimensionnement et le début d'une autre.	`360`	`0` sur `10000`

Custom Availability Zones	Définissez le AZs pour vos instances.	Aucune	`us-east-1a`
			`us-east-1b`
			`us-east-1c`
			`us-east-1d`
			`us-east-1e`
			`eu-central-1`

Nom	Description	Par défaut	Valeurs valides

EnableCapacityRebalancing	Indique s'il faut activer la fonction de rééquilibrage de capacité pour les instances Spot dans votre groupe Auto Scaling. Pour plus d'informations, consultez Rééquilibrage de la capacité dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.
	
	Cette option n'est pertinente que lorsque `EnableSpot` a la valeur `true` dans `aws:ec2:instances` l'espace de noms et quand votre groupe Auto Scaling comprend au moins une instance Spot.
		`false`	`true`
			
			`false`

MinSize	Nombre minimal d'instances souhaitées dans votre groupe Auto Scaling.	`1`	1 sur `10000`
MaxSize	Nombre maximal d'instances souhaitées dans votre groupe Auto Scaling.	`4`	1 sur `10000`

aws:autoscaling:launchconfiguration

Configurez les instances Amazon Elastic Compute Cloud (Amazon EC2) de votre environnement.

Les instances utilisées pour votre environnement sont créées à l'aide d'un modèle de lancement Amazon EC2 ou d'une ressource de configuration de lancement de groupe Auto Scaling. Les options suivantes fonctionnent avec ces deux types de ressources.

Pour de plus amples informations, veuillez consulter the section called “ EC2 Instances Amazon”. Vous pouvez également consulter plus d'informations sur Amazon Elastic Block Store (EBS) dans le chapitre Amazon EBS du guide de l'utilisateur Amazon EC2.

Espace de noms : **`aws:autoscaling:launchconfiguration`**

Nom	Description	Par défaut	Valeurs valides

DisableDefaultEC2SecurityGroup	Lorsqu'il est défini sur la valeur par défaut de`false`, Elastic Beanstalk crée un groupe de sécurité par défaut qui autorise le trafic provenant d'Internet ou d'un équilibreur de charge sur les ports standard pour le protocole HTTP (80). Il attache ce groupe de sécurité aux instances EC2 de l'environnement lorsqu'il crée l'environnement.
	
	Lorsqu'il est défini sur `true` Elastic Beanstalk, le groupe de sécurité par défaut n'est pas attribué aux instances EC2 pour un nouvel environnement. Pour un environnement existant, Elastic Beanstalk annulera l'attribution du groupe de sécurité EC2 par défaut aux instances EC2 de votre environnement. Par conséquent, vous devez également définir les configurations suivantes :
	
	• L'`SecurityGroups` option de cet espace de noms nécessitera au moins une valeur pour définir votre ou vos groupes de sécurité personnalisés.
	• Pour les environnements dotés d'un équilibreur de charge, vous devrez également définir les `SecurityGroups` options dans un autre espace de noms afin de configurer des groupes de sécurité personnalisés pour l'équilibreur de charge.
		`false`	`true`
			
			`false`

Nom	Description	Par défaut	Valeurs valides
	Pour les équilibreurs de charge des applications, définissez l'option dans l'espace de aws:elbv2:loadbalancer noms. Pour les équilibreurs de charge classiques, définissez l'option dans l'espace de aws:elb:loadbalancer noms. • Pour de plus amples informations, veuillez consulter Gestion des groupes de sécurité EC2. Si une valeur est spécifiée pour EC2KeyName un environnement `DisableDefaultEC2SecurityGroup` défini sur `true` un groupe de sécurité par défaut, elle ne sera pas associée aux instances EC2.		

Nom	Description	Par défaut	Valeurs valides

DisableIMDSv1	Définissez sur `true` pour désactiver le service de métadonnées d'instance version 1 (IMDSv1) et l'appliquer IMDSv2.
	
	Réglez sur `false` pour activer à la fois IMDSv1 et IMDSv2.
	
	Les instances de votre environnement se présentent par défaut comme suit, en fonction du système d'exploitation de la plateforme :
	
	• Windows Server AL2 et versions antérieures : activez les deux IMDSv1 et IMDSv2 (DisableIMDSv1par défaut, c'est) `false`
	• AL2023— activer uniquement IMDSv2 (DisableIMDSv1par défaut) `true`
	
	Pour plus d'informations, consultez Configuration du service des métadonnées d'instance.
	
	⚠ Important
	Ce paramètre d'option peut amener Elastic Beanstalk à migrer un environnement existant avec des configurations de lancement vers des modèles de lancement. Cela nécessite les autorisations nécessaires pour gérer les
		`false` – plateformes basées sur le serveur Windows, Amazon Linux 2 et antérieur
		
		`true` – plateformes basées sur Amazon Linux 2023
			`true`
			
			`false`

Nom	Description	Par défaut	Valeurs valides
	modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si vous utilisez des politique s personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre environnement. Pour plus d'informations et d'autres considérations, consultez Migration de votre environne ment Elastic Beanstalk pour lancer des modèles.		

Nom	Description	Par défaut	Valeurs valides
EC2KeyName	Vous pouvez utiliser une paire de clés pour vous connecter en toute sécurité à votre instance EC2. Si une valeur est spécifiée pour `EC2KeyName` un environnement `DisableDefaultEC2SecurityGroup` défini sur `true` un groupe de sécurité par défaut, elle ne sera pas associée aux instances EC2. **Note** Si vous utilisez la console Elastic Beanstalk pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console remplace cette option par une valeur recommandée.	Aucune	

Nom	Description	Par défaut	Valeurs valides
IamInstanceProfile	Un profil d'instance permet aux utilisateurs et aux AWS services Gestion des identités et des accès AWS (IAM) d'accéder à des informations d'identification de sécurité temporaires pour effectuer des appels d' AWS API. Spécifiez le nom du profil d'instance ou son ARN. Exemples : • `aws-elasticbeanstalk-ec2-role` • `arn:aws:iam::123456789012:instance-profile/aws-elasticbeanstalk-ec2-role` **Note** Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console et l'interface de ligne de commande (CLI) EB remplacent cette option par une valeur recommandée.	Aucune	Nom du profil d'instance ou ARN.

Nom	Description	Par défaut	Valeurs valides
ImageId	Vous pouvez remplacer la valeur Amazon Machine Image (AMI) par défaut en spécifiant votre propre ID d'AMI personnalisé. Exemple : `ami-1f316660`	Aucune	

Nom	Description	Par défaut	Valeurs valides
InstanceType	Type d'instance utilisé pour exécuter votre application dans un environnement Elastic Beanstalk. **Important** L'option `InstanceType` est obsolète. Elle est remplacée par la nouvelle option `InstanceTypes` plus puissante dans l'espace de noms `aws:ec2:instances`. Vous pouvez utiliser cette nouvelle option pour spécifier une liste d'un ou plusieurs types d'instance pour votre environnement. La première valeur de cette liste est équivalente à la valeur de l'option `InstanceType` incluse dans l'espace de noms `aws:autoscaling:launchconfiguration` décrit ici. Nous vous recommandons de spécifier les types d'instance en utilisant la nouvelle option. Si elle est spécifiée, la nouvelle option prévaut sur la précédente. Pour de plus amples informations, veuillez consulter the section called “Espace de noms aws:ec2:instances”.	Varie selon le compte et la région.	Un type d'instance EC2. Varie selon le compte, la région et la zone de disponibilité. Vous pouvez obtenir une liste des types d'instance Amazon EC2 filtrés par ces valeurs. Pour plus d'informations, consultez la section Types d'instances disponibles dans le guide de l'utilisateur Amazon EC2.

Nom	Description	Par défaut	Valeurs valides
	Les types d'instance disponibles dépendent des zones de disponibilité et de la région utilisées. Si vous choisissez un sous-réseau, la zone de disponibilité qui contient ce sous-réseau détermine les types d'instance disponibles. • Elastic Beanstalk ne prend pas en charge les types d'instance Amazon EC2 Mac. • Pour plus d'informations sur les familles et les types d'instances Amazon EC2, consultez la section Types d'instances dans le guide de l'utilisateur Amazon EC2. • Pour plus d'informations sur les types d'instances disponibles dans les différentes régions, consultez la section Types d'instances disponibles dans le guide de l'utilisateur Amazon EC2. **Note** Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console et l'interface de ligne de commande (CLI) EB		

Nom	Description	Par défaut	Valeurs valides
	remplacent cette option par une valeur recommandée.		

Nom	Description	Par défaut	Valeurs valides

LaunchTemplateTagPropagationEnabled	Définissez la valeur sur `true` pour activer la propagation des balises d'environnement vers les modèles de lancement pour des ressources spécifiques fournies à l'environnement.
	
	Elastic Beanstalk peut uniquement propager des balises pour lancer des modèles pour les ressources suivantes :
	
	• Volumes EBS
	• Instances EC2
	• Interface réseau EC2
	• CloudFormation modèles de lancement qui définissent une ressource
	
	Cette contrainte existe car les balises CloudFormation ne sont autorisées que pour des ressources spécifiques lors de la création de modèles. Pour plus d'informations, consultez TagSpecification le guide de AWS CloudFormation l'utilisateur.
	
	⚠ Important
	
	• La modification de la valeur de cette option de `false` à `true` pour un environnement existant peut constituer un changement critique pour les balises existantes.
		`false`	`true`
			
			`false`

Nom	Description	Par défaut	Valeurs valides
	• Lorsque cette fonctionnalité est activée, la propagation des balises nécessite le remplacement d'EC2, ce qui peut entraîner des temps d'arrêt. Vous pouvez activer les mises à jour dynamiques pour appliquer les modifications de configuration par lots et éviter les temps d'arrêt de service pendant le processus de mise à jour. Pour de plus amples informations, veuillez consulter Configuration changes. Pour en savoir plus sur l'utilisation des modèles de lancement, consultez ce qui suit : • Launch templates dans le Guide de l'utilisateur Amazon EC2 Auto Scaling • Utilisation de modèles dans le Guide de l'utilisateur AWS CloudFormation • Extraits de modèle Elastic Beanstalk dans le Guide de l'utilisateur AWS CloudFormation Pour plus d'informations sur cette option, consultez Propagation des balises vers les modèles de lancement.		

Nom	Description	Par défaut	Valeurs valides

MonitoringInterval	Intervalle (en minutes) auquel vous souhaitez que CloudWatch les métriques Amazon soient renvoyées.	`5 minute`	`1 minute`
			
			`5 minute`

Nom	Description	Par défaut	Valeurs valides
SecurityGroups	Répertorie les identifiants des groupes de sécurité Amazon EC2 à attribuer aux instances EC2 du groupe Auto Scaling afin de définir des règles de pare-feu pour les instances. Utilisez cette option `DisableDefaultEC2SecurityGroup` pour associer vos propres groupes de sécurité personnalisés qui définissent les règles de pare-feu pour les instances EC2. Pour de plus amples informations, veuillez consulter Environnements à charge équilibrée (multi-instances). **Important** Vous devrez peut-être effectuer une configuration supplémentaire pour empêcher le blocage du trafic entrant vers vos instances EC2. Cela s'applique uniquement aux environnements multi-instances dotés de groupes de sécurité EC2 personnalisés. Les groupes de sécurité EC2 doivent inclure une règle entrante qui accorde l'accès au trafic acheminé depuis l'équilibreur de charge. Pour de plus amples informations, veuillez consulter Gestion des groupes de sécurité EC2	`elasticbeanstalk-default`	

Nom	Description	Par défaut	Valeurs valides
	dans des environnements multi-instances. Vous pouvez fournir une chaîne unique de valeurs séparées par des virgules contenant les identifiants des groupes de sécurité Amazon EC2 existants ou des références AWS::EC2::Security Group à des ressources créées dans le modèle. Vous devez fournir au moins une valeur pour cette option si `DisableDefaultEC2SecurityGroup` cet espace de noms est défini sur. `true`		

Nom	Description	Par défaut	Valeurs valides
SSHSource Restriction	Utilisé pour verrouiller l'accès SSH à un environnement. Par exemple, vous pouvez verrouiller l'accès SSH aux instances EC2 afin que seul un hôte bastion puisse accéder aux instances du sous-réseau privé. Cette chaîne prend la forme suivante : *protocol*, *fromPort*, *toPort*, *source_restriction* *protocol* Le protocole pour la règle de trafic entrant. *fromPort* Le numéro de port de départ. *toPort* Le numéro de port fin. *source_restriction* La plage de routage interdomaines sans classe (CIDR) ou le groupe de sécurité par lequel le trafic doit être acheminé. Spécifiez le groupe de sécurité avec l'ID du groupe de sécurité. Pour spécifier un groupe de sécurité à partir d'un autre compte, incluez l'ID du AWS compte avant l'ID du groupe de sécurité, en le séparant par une barre oblique.	Aucune	

Nom	Description	Par défaut	Valeurs valides
	L'autre compte doit se trouver dans la même AWS région. Notez la syntaxe :*aws-account-id* /*security-group-id*. Par exemple : **123456789012 /sg-99999999** Exemples : • `tcp, 22, 22, 54.240.196.185/32` • `tcp, 22, 22, my-security-group-id` • `tcp, 22, 22, 123456789012/their-security-group-id`		

Nom	Description	Par défaut	Valeurs valides
BlockDeviceMappings	Attachez des volumes Amazon EBS supplémentaires ou des volumes de stockage d'instance sur toutes les instances dans le groupe Auto Scaling. **⚠ Important** Ce paramètre d'option peut amener Elastic Beanstalk à migrer un environnement existant avec des configurations de lancement vers des modèles de lancement. Cela nécessite les autorisations nécessaires pour gérer les modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si vous utilisez des politiques personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre environnement. Pour plus d'informations et d'autres considérations, consultez Migration de votre environnement Elastic Beanstalk pour lancer des modèles. Lorsque vous mappez des volumes de stockage d'instance, vous devez	Aucune	• taille : doit être comprise entre 500 et 16 384 Gio • débit : doit être compris entre 125 et 1 000 mébioctets par seconde (Mio/s)

Nom	Description	Par défaut	Valeurs valides
	mapper uniquement le nom du périphérique à un nom de volume. Toutefois, nous vous recommandons, lors du mappage de volumes Amazon EBS, de spécifier en outre certains ou tous les champs suivants (chaque champ doit être séparé par un signe deux points) : • ID d'instantané • taille, en Go • Supprimer à la suspension (`true` ou `false`) • type de stockage (uniquement pour `gp3`, `gp2`, `standard`, `st1`, `sc1`, ou `io1`) • IOPS (uniquement pour `gp3` ou `io1`) • débit (uniquement pour `gp3`) L'exemple suivant attache trois volumes Amazon EBS, un volume vide gp2 de 100 Go et un instantané, un volume io1 vide de 20 Go avec 2000 IOPS provisionnées, et un volume de stockage d'instance `ephemeral0` . Plusieurs volumes de stockage d'instance peuvent être attachées si le type d'instance assure la prise en charge. `/dev/sdj=:100:true:gp2,/dev/sdh=snap-51eef269,/dev/sdi=:20:true:io1:2000,/dev/sdb=ephemeral0`		

Nom	Description	Par défaut	Valeurs valides
RootVolumeType	Type de volume (magnétique, usage général SSD ou IOPS provisionnés (SSD)) à utiliser pour le volume Amazon EBS racine attaché aux instances EC2 de votre environnement. **Important** Ce paramètre d'option peut amener Elastic Beanstalk à migrer un environnement existant avec des configurations de lancement vers des modèles de lancement. Cela nécessite les autorisations nécessaires pour gérer les modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si vous utilisez des politiques personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre environnement. Pour plus d'informations et d'autres considérations, consultez Migration de votre environnement Elastic Beanstalk pour lancer des modèles.	Varie selon la plateforme.	`standard` pour stockage magnétique. `gp2` ou `gp3` pour polyvalent (SSD). `io1` pour IOPS provisionnés (SSD).

Nom	Description	Par défaut	Valeurs valides
RootVolumeSize	Capacité de stockage du volume Amazon EBS racine en Go. Nécessaire si vous définissez `RootVolumeType` sur IOPS provisionnées (SSD). Par exemple, `"64"`.	Varie selon la plateforme pour stockage magnétique et usage général (SSD). Aucune pour IOPS provisionnées (SSD).	`10` sur `16384` Go pour usage général et IOPS provisionnées (SSD). `8` sur `1024` Go pour magnétique.
RootVolumeIOPS	input/output Opérations par seconde (IOPS) souhaitées pour un volume racine SSD IOPS provisionné ou pour un volume racine `gp3` SSD à usage général. Le rapport maximal d'IOPS selon la taille du volume est de 500 à 1. Par exemple, un volume à 3 000 IOPS doit être d'au moins 6 Gio.	Aucune	`100` à `20000` pour les volumes racine de SSD IOPS provisionnés io1. `3000` à `16000` pour les volumes racine de SSD `gp3` polyvalents.
RootVolumeThroughput	Débit souhaité en megioctets par seconde (MiB/s)) pour approvisionner le volume racine Amazon EBS attaché à l'instance EC2 de votre environnement. **Note** Cette option s'applique uniquement aux types de stockages `gp3`.	Aucune	`125` sur `1000`

aws:autoscaling:scheduledaction

Configurez les actions planifiées pour le groupe Auto Scaling de votre environnement. Pour chaque action, indiquez un `resource_name` en plus du nom de l'option, de l'espace de noms et de la valeur de chaque paramètre. Pour obtenir des exemples, consultez Espace de noms aws:autoscaling:scheduledaction.

Espace de noms : **`aws:autoscaling:scheduledaction`**

Nom	Description	Par défaut	Valeurs valides
StartTime	Pour les actions exceptionnelles, choisissez la date et l'heure pour exécuter l'action. Pour des actions récurrentes, choisissez le moment auquel l'action doit être activée.	Aucune	Horodatage ISO-8601 unique à travers toutes les actions de dimensionnement planifiées.
EndTime	Date et heure futures (dans le UTC/GMT fuseau horaire) auxquelles vous souhaitez que l'action de dimension nement planifiée cesse de se répéter. Si vous ne spécifiez pas de EndTime, l'action se reproduit en fonction de l'`Recurrence` expression. Exemple : `2015-04-28T04:07:2Z` Lorsqu'une action planifiée se termine, Amazon EC2 Auto Scaling ne revient pas automatiquement à ses paramètres précédents. Configurez une seconde action planifiée pour revenir aux paramètres d'origine selon vos besoins.	Aucune	Horodatage ISO-8601 unique à travers toutes les actions de dimensionnement planifiées.
MaxSize	Nombre maximal d'instances à appliquer lorsque l'action s'exécute.	Aucune	`0` sur `10000`

Nom	Description	Par défaut	Valeurs valides
MinSize	Nombre minimal d'instances à appliquer lorsque l'action s'exécute.	Aucune	`0` sur `10000`
DesiredCapacity	Définissez la capacité initiale souhaitée pour le groupe Auto Scaling. Lorsque l'action planifiée est appliquée, des déclencheurs ajustent la capacité souhaitée en fonction de leurs paramètres.	Aucune	`0` sur `10000`
Recurrence	La fréquence à laquelle vous souhaitez que l'action planifiée s'exécute. Si vous ne spécifiez pas de récurrence, l'action de dimensionnement s'exécute une seule fois, comme indiqué par la valeur `StartTime` .	Aucune	Expression Cron.

Suspend	Définissez le paramètre sur `true` pour désactiver temporairement une action planifiée récurrente.	`false`	`true`
			
			`false`

aws:autoscaling:trigger

Configurez les déclencheurs de mise à l'échelle pour le groupe Auto Scaling de votre environnement.

> **Note**
>
> Trois options de cet espace de noms déterminent combien de temps la métrique d'un déclencheur peut rester au-delà de ses limites définies avant le lancement du déclencheur. Ces options sont liées comme suit :
> `BreachDuration = Period * EvaluationPeriods`
> Les valeurs par défaut de ces options (respectivement 5, 5 et 1) correspondent à cette équation. Si vous spécifiez des valeurs incohérentes, Elastic Beanstalk peut modifier l'une d'entre elles afin que l'équation soit toujours satisfaite.

Espace de noms : **`aws:autoscaling:trigger`**

Nom	Description	Par défaut	Valeurs valides
BreachDuration	Durée (en minutes) pendant laquelle une métrique peut excéder la limite définie (comme indiqué dans `UpperThreshold` et `LowerThreshold`) avant que le déclencheur ne soit appelé.	`5`	`1` sur `600`
LowerBreachScaleIncrement	Nombre d'instances Amazon EC2 à supprimer lorsque vous effectuez une activité de dimensionnement.	`-1`	
LowerThreshold	Si la mesure passe sous cette valeur pendant la durée de la faille, un déclencheur est appelé.	`2000000`	`0` sur `20000000`

MeasureName	Métrique utilisée pour votre déclencheur Auto Scaling.
	
	Note
	`HealthyHostCount`, `UnhealthyHostCount` et `TargetResponseTime` ne sont applicables qu'aux environnements dotés d'un équilibreur de charge dédié. Il ne s'agit pas de valeurs de métriques valides pour les environnements configurés avec un équilibreur de charge partagé. Pour plus d'informations sur les types d'équilibreurs de charge, consultez la section Équilibreur de charge
		`NetworkOut`	`CPUUtilization`
			
			`NetworkIn`
			
			`NetworkOut`
			
			`DiskWriteOps`
			
			`DiskReadBytes`
			
			`DiskReadOps`
			
			`DiskWriteBytes`
			
			`Latency`
			
			`RequestCount`

Nom	Description	Par défaut	Valeurs valides

	pour votre environnement Elastic Beanstalk.		`HealthyHostCount`
			`UnhealthyHostCount`
			`TargetResponseTime`

Period	Spécifie la fréquence à laquelle Amazon CloudWatch mesure les statistiques de votre déclencheur. La valeur est le nombre de minutes entre deux périodes consécutives.	`5`	1 sur `600`
EvaluationPeriods	Nombre de périodes d'évaluation consécutives utilisé pour déterminer si une utilisation hors limites est en cours.	`1`	1 sur `600`

Statistic	Statistique que le déclencheur utilise, telle que `Average`.	`Average`	`Minimum`
			`Maximum`
			`Sum`
			`Average`

Nom	Description	Par défaut	Valeurs valides

Unit	Unité de mesure du déclencheur, telle que `Bytes`.	`Bytes`	`Seconds`
			`Percent`
			`Bytes`
			`Bits`
			`Count`
			`Bytes/Second`
			`Bits/Second`
			`Count/Second`
			`None`

UpperBreachScaleIncrement	Nombre d'instances Amazon EC2 à ajouter lors d'une activité de dimensionnement.	1	
UpperThreshold	Si la mesure est supérieure à ce nombre pendant la durée de la faille, un déclencheur est appelé.	`6000000`	`0` sur `20000000`

aws:autoscaling:updatepolicy:rollingupdate

Configurer les mises à jour propagées du groupe Auto Scaling de votre environnement

Espace de noms : **`aws:autoscaling:updatepolicy:rollingupdate`**

Nom	Description	Par défaut	Valeurs valides
MaxBatchSize	Le nombre d'instances incluses dans chaque lot de la mise à jour propagée.	Un tiers de la taille minimale du groupe Auto Scaling, arrondie	1 sur `10000`

Nom	Description	Par défaut	Valeurs valides
		à l'entier suivant le plus élevé.	
MinInstancesInService	Le nombre minimum d'instances qui doivent être en service au sein du groupe Auto Scaling, tandis que d'autres instances sont résiliées.	La taille minimale du groupe Auto Scaling ou une taille inférieure à la taille maximale du groupe Auto Scaling, la valeur la plus basse prévalant.	`0` sur 9999

Nom	Description	Par défaut	Valeurs valides

RollingUpdateEnabled	Si `true`, permet le déploiement de mises à jour pour un environnement. Les mises à jour propagées sont utiles quand vous apportez de petites et fréquentes mises à jour à votre application logicielle Elastic Beanstalk et que vous voulez éviter l'interruption des applications.
	
	La configuration de cette valeur sur true active automatiquement les options `MaxBatchSize` , `MinInstancesInService` et `PauseTime` . La définition d'une de ces options définit également automatiquement l'option `RollingUpdateEnabled` sur `true`. La configuration de cette option sur `false` désactive les mises à jour propagées.
		`false`	`true`
			
			`false`

Nom	Description	Par défaut	Valeurs valides
	Note Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console et l'interface de ligne de commande EB CLI remplacent cette option par une valeur recommandée.		

Nom	Description	Par défaut	Valeurs valides

RollingUpdateType	Trois types différents : les mises à jour propagées basées sur le temps, les mises à jour propagées basées sur l'état et les mises à jour immuables.
	
	Les mises à jour progressives basées sur le temps s'appliquent PauseTime entre les lots. Les mises à jour propagées basées sur l'intégrité attendent que de nouvelles instances transmettent les vérifications de l'état avant de passer au lot suivant. Immutable updates (Les mises à jour immuables) lancent un ensemble complet d'instances dans un nouveau groupe Auto Scaling.
	
	Note
	Si vous utilisez la console Elastic
		`Time`	`Time`
			
			`Health`
			
			`Immutable`

Nom	Description	Par défaut	Valeurs valides
	Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console et l'interface de ligne de commande (CLI) EB remplacent cette option par une valeur recommandée.		
PauseTime	Délai d'attente (en secondes, minutes ou heures) du service Elastic Beanstalk entre la fin des mises à jour d'un lot d'instances et le début de celles du lot suivant.	Calculée automatiquement en fonction du type d'instance et du conteneur.	`PT0S`* (0 seconde) à `PT1H` (1 heure)

Nom	Description	Par défaut	Valeurs valides
Timeout	Durée maximale (en minutes ou heures) pour que toutes les instances d'un lot d'instances passent les vérifications de l'état avant que la mise à jour soit annulée.	`PT30M` (30 minutes)	`PT5M`* (5 minutes) à `PT1H` (1 heure) * format de ISO8601durée : `PT#H#M#S` où chaque # est le nombre d'heures, de minutes, de and/or secondes, respectivement.

aws:ec2:instances

Configurez les instances de votre environnement, y compris les options d'instances Spot. Cet espace de noms complète `aws:autoscaling:launchconfiguration` et `aws:autoscaling:asg`.

Pour de plus amples informations, veuillez consulter the section called “Groupe Auto Scaling”.

Espace de noms : **`aws:ec2:instances`**

Nom	Description	Par défaut	Valeurs valides

EnableSpot	Activez les demandes d'instances Spot pour votre environnement. Lorsqu'elles sont définies sur `false`, certaines options de cet espace de noms n'ont aucun effet.
	
	Important
	Ce paramètre d'option peut amener Elastic Beanstalk à migrer un environnement existant avec des configura
		`false`	`true`
			
			`false`

Nom	Description	Par défaut	Valeurs valides
	tions de lancement vers des modèles de lancement. Cela nécessite les autorisations nécessaires pour gérer les modèles de lancement. Ces autorisations sont incluses dans notre politique gérée. Si vous utilisez des politique s personnalisées au lieu de nos politiques gérées, la création ou les mises à jour de l'environnement risquent d'échouer lorsque vous mettez à jour la configuration de votre environnement. Pour plus d'informations et d'autres considérations, consultez Migration de votre environne ment Elastic Beanstalk pour lancer des modèles.		

Nom	Description	Par défaut	Valeurs valides
InstanceTypes	Liste séparée par des virgules de types d'instance que vous souhaitez que votre environnement utilise (par exemple, `t2.micro,t3.micro`). Lorsque cette option `EnableSpot` `SpotAllocationStrategy` est définie sur `true` et si elle est définie sur`capacity-optimized-prioritized` , la liste de valeurs spécifiée dans cette option détermine la priorité du type d'instance pour la stratégie d'allocation des instances Spot. Lorsque les instances Spot ne sont pas activées (champ `EnableSpot` défini sur `false`), seul le premier type d'instance de la liste est utilisé. Le premier type d'instance de la liste dans cette option est équivalent à la valeur de l'option `InstanceType` dans l'espace de noms `aws:autoscaling:launchconfiguration` . Nous ne recommandons pas d'utiliser cette dernière option, car elle est obsolète. Si vous spécifiez les deux, le premier type d'instance de la liste de l'option `InstanceTypes` est utilisé et `InstanceType` est ignoré. Les types d'instance disponibles dépendent des zones de disponibilité et de la région utilisées. Si vous	Liste de deux types d'instance. Varie selon le compte et la région.	Un à quarante types d'instances EC2. Nous en recommandons au moins deux. Varie selon le compte, la région et la zone de disponibilité. Vous pouvez obtenir une liste des types d'instance Amazon EC2 filtrés par ces valeurs. Pour plus d'informations, consultez la section Types d'instances disponibles dans le guide de l'utilisateur Amazon EC2. Les types d'instance doivent tous faire partie de la même architecture (`arm64`, `x86_64`, `i386`). `Supported Architectures` fait également partie de cet espace de noms. Si vous fournissez des valeurs pour `Supported Architectures` , la ou les valeurs que vous entrez pour `InstanceTypes`

Nom	Description	Par défaut	Valeurs valides
	choisissez un sous-réseau, la zone de disponibilité qui contient ce sous-réseau détermine les types d'instance disponibles. • Elastic Beanstalk ne prend pas en charge les types d'instance Amazon EC2 Mac. • Pour plus d'informations sur les familles et les types d'instances Amazon EC2, consultez la section Types d'instances dans le guide de l'utilisateur Amazon EC2. • Pour plus d'informations sur les types d'instances disponibles dans les différentes régions, consultez la section Types d'instances disponibles dans le guide de l'utilisateur Amazon EC2. **Note** Certains AWS comptes plus anciens peuvent fournir à Elastic Beanstalk des types d'instances par défaut qui ne prennent pas en charge les instances Spot (par exemple, t1.micro). Si vous activez les demandes d'instances Spot et que vous obtenez une erreur concernant un type d'instance qui ne prend pas en charge		doivent appartenir à une seule des architectures que vous fournissez pour `Supported Architectures` .

Nom	Description	Par défaut	Valeurs valides
	les instances Spot, assurez-vous de configurer des types d'instance qui prennent en charge les instances Spot. Pour choisir des types d'instance Spot, utilisez Spot Instance Advisor. Lorsque vous mettez à jour la configuration de votre environnement et supprimez un ou plusieurs types d'instance de l'option `InstanceTypes` , Elastic Beanstalk résilie toutes les instances Amazon EC2 exécutées sur les types d'instance supprimés . Le groupe Auto Scaling de votre environnement lance ensuite de nouvelles instances, si nécessaire pour compléter la capacité souhaitée , en utilisant les types d'instance spécifiés actuels.		

Nom	Description	Par défaut	Valeurs valides

SpotAllocationStrategy	Spécifie la stratégie d'allocation des instances ponctuelles qui détermine la manière dont les instances ponctuelles sont allouées à partir des pools de capacités ponctuelles disponibles.
	
	Si ce paramètre est défini sur `capacity-optimized-prioritized`, l'ordre des valeurs entrées `InstanceTypes` définit la priorité du type d'instance pour l'allocation.
	
	Cette option est pertinente uniquement lorsque `EnableSpot` est défini sur `true`.
		`capacity-optimized`	`capacity-optimized`
			
			`price-capacity-optimized`
			
			`capacity-optimized-prioritized`
			
			`lowest-price`

SpotFleetOnDemandBase	Nombre minimal d'instances à la demande que votre groupe Auto Scaling alloue avant de considérer les instances Spot à mesure que votre environnement augmente. Cette option est pertinente uniquement lorsque `EnableSpot` est défini sur `true`.	`0`	`0` à option `MaxSize` dans l'espace de noms `aws:autoscaling:asg`

Nom	Description	Par défaut	Valeurs valides
SpotFleetOnDemandAboveBasePercentage	Pourcentage d'instances à la demande dans le cadre de la capacité supplémentaire que votre groupe Auto Scaling alloue au-delà des instances `SpotOnDemandBase` . Cette option est pertinente uniquement lorsque `EnableSpot` est défini sur `true`.	`0` pour un environment d'instance unique `70` pour un environment à charge équilibrée	`0` sur `100`

SpotMaxPrice	Prix maximum par heure unitaire, en dollars américains, que vous êtes prêt à payer pour une instance Spot. Pour obtenir des recommandations concernant les options de prix maximum pour les instances Spot, consultez l'historique des tarifs des instances Spot dans le guide de l'utilisateur Amazon EC2.
	
	Cette option est pertinente uniquement lorsque `EnableSpot` est défini sur `true`.
		Prix à la demande par type d'instance. La valeur de l'option dans ce cas est `null`.	`0.001` sur `20.0`
			
			`null`

Nom	Description	Par défaut	Valeurs valides

SupportedArchitectures	Liste séparée par des virgules des types d'architecture d'instance EC2 que votre environnement doit utiliser.
	
	Elastic Beanstalk prend en charge les types d'instance basés sur les architectures de processeur suivantes :
	
	• AWS Architecture Arm 64 bits de Graviton (arm64)
	• Architecture 64 bits (x86_64)
	• Architecture 32 bits (i386)
	
	Pour en savoir plus sur l'architecture du processeur et les types d'instance Amazon EC2, consultez the section called “Types d' EC2 instances Amazon”.
		Aucun	`arm64`
			
			`x86_64`
			
			`i386`
			
			Note
			L'architecture 32 bits `i386` n'est pas prise en charge par la majorité des plateformes Elastic Beanstalk. Nous vous recommandons de choisir plutôt les types d'architecture `x86_64` ou `arm64`.

aws:ec2:vpc

Configurez votre environnement pour lancer des ressources dans un Amazon Virtual Private Cloud (Amazon VPC) personnalisé. Si vous ne configurez pas de paramètres dans cet espace de noms, Elastic Beanstalk lance les ressources dans le VPC par défaut.

Espace de noms : **`aws:ec2:vpc`**

Nom	Description	Par défaut	Valeurs valides
VPCId	L'ID de votre VPC Amazon.	Aucune	
Subnets	Le ou IDs les sous-réseaux du groupe Auto Scaling. Si vous avez plusieurs sous-réseaux, spécifiez la valeur sous la forme d'une seule chaîne de sous-réseau séparée par des virgules IDs (par exemple,). `"subnet-11111111,subnet-22222222"`	Aucune	
ELBSubnets	Le ou IDs les sous-réseaux de l'équilibreur de charge élastique. Si vous avez plusieurs sous-réseaux, spécifiez la valeur sous la forme d'une seule chaîne de sous-réseau séparée par des virgules IDs (par exemple,). `"subnet-11111111,subnet-22222222"`	Aucune	
ELBScheme	Spécifiez `internal` si vous souhaitez créer un équilibreur de charge interne dans votre Amazon VPC afin qu'aucun accès à votre application Elastic Beanstalk ne soit possible depuis l'extérieur de votre Amazon VPC. Si vous spécifiez une valeur autre que `public` ou `internal`, Elastic Beanstalk ignore la valeur.	`public`	`public` `internal`
DBSubnets	Contient les sous-réseaux IDs de base de données. Ce paramètre n'est utilisé que si vous souhaitez ajouter une instance DB Amazon RDS dans le cadre de votre application. Si vous avez plusieurs sous-réseaux, spécifiez la valeur sous la forme d'une seule chaîne de sous-réseau séparée par des virgules IDs (par exemple,). `"subnet-11111111,subnet-22222222"`	Aucune	

Nom	Description	Par défaut	Valeurs valides

Associate PublicIpAddress	Spécifie s'il faut lancer des instances avec des adresses IP publiques dans votre VPC Amazon. Les instances avec des adresses IP publiques ne requièrent pas de périphérique NAT pour communiquer avec Internet. Vous devez définir la valeur sur `true` si vous souhaitez inclure votre équilibreur de charge et des instances dans un seul sous-réseau public.
	
	Cette option n'a aucun effet sur un environnement d'instance unique, qui a toujours une seule instance Amazon EC2 avec une adresse IP Elastic. Cette option est pertinente pour les environnements évolutifs avec équilibrage de charge.
		Aucune	`true`
			
			`false`

aws:elasticbeanstalk:application

Configurez un chemin de vérification de l'état pour votre application. Pour de plus amples informations, veuillez consulter Création de rapports d'intégrité de base.

Espace de noms : **`aws:elasticbeanstalk:application`**

Nom	Description	Par défaut	Valeurs valides

URL de vérification d'intégrité de l'application	Chemin d'accès où les demandes de vérification de l'état sont envoyées. Si ce chemin n'est pas défini, l'équilibreur de charge tente d'établir une connexion TCP sur le port 80 pour vérifier le statut de l'état de votre application. Définissez sur un chemin d'accès commençant par / pour envoyer une requête HTTP GET à ce chemin d'accès. Vous pouvez également inclure un protocole (HTTP,	Aucun	Les valeurs valides sont les suivantes :
			
			/ (HTTP GET pour le chemin d'accès racine)
			
			`/health`
			
			`HTTPS:443/`
			
			`HTTPS:443/ health`

Nom	Description	Par défaut	Valeurs valides
	HTTPS, TCP, ou SSL) et un port avant le chemin d'accès pour vérifier la connectivité HTTPS ou utiliser un port autre que le port par défaut. **Note** Si vous utilisez la console Elastic Beanstalk pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console remplace cette option par une valeur recommandée.		

L'interface de ligne de commande (CLI) EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Consultez Valeurs recommandées pour plus de détails.

aws:elasticbeanstalk:application:environment

Configurez les propriétés d'environnement pour votre application.

Espace de noms : **`aws:elasticbeanstalk:application:environment`**

Nom	Description	Par défaut	Valeurs valides
Tout nom de variable d'environnement.	Passez en paires clé-valeur.	Aucune	Toute valeur de variable d'environnement.

Pour plus d'informations, consultez Variables d'environnement et autres paramètres du logiciel.

patte : tige de haricot élastique : application : secrets environnementaux

Configurez les variables d'environnement pour qu'elles servent de secrets d'environnement à votre application. Les secrets d'environnement stockent AWS Secrets Manager les secrets ou les AWS Systems Manager paramètres du magasin de paramètres.

Espace de noms : **`aws:elasticbeanstalk:application:environmentsecrets`**

Nom	Description	Par défaut	Valeurs valides
OptionName	Spécifie le nom de la variable d'environnement qui doit contenir la valeur de la banque secrète ou de la banque de paramètres.	Aucun	Tout nom de variable d'environnement.
Value	Spécifie l'ARN de la valeur stockée dans AWS Secrets Manager ou AWS Systems Manager Parameter Store. Pendant le démarrage de l'instance, Elastic Beanstalk initie la variable d'environnement à la valeur stockée dans cette ressource ARN. **Note** Assurez-vous que les autorisations nécessaires sont en place pour que le rôle de profil d'instance EC2 de votre environnement puisse accéder au secret et au paramètre ARNs. Pour de plus amples informations, veuillez consulter Autorisations IAM requises.	Aucun	Valeur ARN valide pour un AWS Secrets Manager secret ou une valeur de AWS Systems Manager paramètre Parameter Store.

Pour de plus amples informations, veuillez consulter Configuration des secrets en tant que variables d'environnement.

aws:elasticbeanstalk:cloudwatch:logs

Configurez le streaming des journaux d'instance pour votre application.

Espace de noms : **`aws:elasticbeanstalk:cloudwatch:logs`**

Nom	Description	Par défaut	Valeurs valides
StreamLogs	Spécifie s'il convient de créer des groupes dans les CloudWatch journaux pour les journaux de proxy et de déploiement, et de diffuser les journaux de chaque instance de votre environnement.	`false`	`true` `false`
DeleteOnTerminate	Indique s'il faut supprimer les groupes de journaux lorsque l'environnement est suspendu. Si la valeur est `false`, les journaux sont conservés pendant `RetentionInDays` jours.	`false`	`true` `false`
RetentionInDays	Le nombre de jours de conservation des événements de journal avant leur expiration.	7	1, 3, 5, 7, 14, 30, 60, 90, 120, 150, 180, 365, 400, 545, 731, 1827, 3653

aws:elasticbeanstalk:cloudwatch:logs:health

Configurez le streaming des journaux d'intégrité de l'environnement pour votre application.

Espace de noms : **`aws:elasticbeanstalk:cloudwatch:logs:health`**

Nom	Description	Par défaut	Valeurs valides
HealthStreamingEnabled	Pour les environnements dans lesquels les rapports de santé améliorés sont activés, indique s'il convient de créer un groupe dans CloudWatch Logs pour l'état de l'environnement et d'archiver les données relatives à l'état de l'environnement d'Elastic Beanstalk. Pour plus d'informations sur l'activation de l'état amélioré, consultez `aws:elasticbeanstalk:health reporting:system`.	`false`	`true` `false`
DeleteOnTerminate	Indique s'il faut supprimer le groupe de journaux lorsque l'environnement est suspendu. Si la valeur est `false`, les données d'intégrité sont conservées pendant `RetentionInDays` jours.	`false`	`true` `false`
RetentionInDays	Nombre de jours pendant lequel il faut conserver les données d'intégrité archivées avant leur expiration.	7	1, 3, 5, 7, 14, 30, 60, 90, 120, 150, 180, 365, 400, 545, 731, 1827, 3653

aws:elasticbeanstalk:command

Configurez la politique de déploiement pour votre code d'application. Pour de plus amples informations, veuillez consulter the section called “Options de déploiement”.

Espace de noms : **`aws:elasticbeanstalk:command`**

Nom	Description	Par défaut	Valeurs valides

DeploymentPolicy	Sélectionnez une politique de déploiement pour les déploiements de version d'application.
	
	Note
	Si vous utilisez la console Elastic Beanstalk pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console remplace cette option par une valeur recommandée.
		`AllAtOnce`	`AllAtOnce`
			
			`Rolling`
			
			`RollingWithAdditionalBatch`
			
			`Immutable`
			
			`TrafficSplitting`

Timeout	Délai d'attente, en secondes, pour qu'une instance termine l'exécution des commandes. Elastic Beanstalk ajoute 240 secondes (quatre minutes) à la valeur `Timeout`. Par exemple, le délai d'attente par défaut est de 840 secondes (600+240), ou 14 minutes.	`600`	1 sur `3600`

BatchSizeType	Type de numéro spécifié dans BatchSize.
	
	Note
	Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console et l'interface de ligne de commande EB CLI
		`Percentage`	`Percentage`
			
			`Fixed`

Nom	Description	Par défaut	Valeurs valides
	remplacent cette option par une valeur recommandée.		
BatchSize	Pourcentage ou nombre fixe d'instances Amazon EC2 dans le groupe Auto Scaling sur lesquelles réaliser des déploiements simultanément. Les valeurs valides varient en fonction du BatchSizeTypeparamètre utilisé. **Note** Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console et l'interface de ligne de commande EB CLI remplacent cette option par une valeur recommandée.	`100`	1 à `100` (`Percentage`). 1à aws:autoscaling:asg : : () MaxSize `Fixed`

IgnoreHealthCheck	N'annulez pas un déploiement en raison d'un échec de vérifications de l'état.	`false`	`true`
			
			`false`

aws:elasticbeanstalk:environment

Configurez l'architecture et le rôle de service de votre environnement.

Espace de noms : **`aws:elasticbeanstalk:environment`**

Nom	Description	Par défaut	Valeurs valides

EnvironmentType	Affectez la valeur `SingleInstance` pour lancer une instance EC2 sans équilibreur de charge.	`LoadBalanced`	`SingleInstance`
			
			`LoadBalanced`

ServiceRole	Le nom d'un rôle IAM qu'Elastic Beanstalk utilise pour gérer des ressources pour l'environnement. Spécifiez un nom de rôle (éventuellement précédé d'un chemin personnalisé) ou son ARN. Exemples : • `aws-elasticbeanstalk-service-role` • *`custom-path`* `/`*`custom-role`* • `arn:aws:iam::123456789012:role/aws-elasticbeanstalk-service-role` **Note** Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console et l'interface de ligne de commande (CLI) EB remplacent cette option par une valeur recommandée.	Aucune	Nom de rôle IAM, chemin/nom ou ARN

Nom	Description	Par défaut	Valeurs valides

LoadBalancerType	Type de l'équilibreur de charge pour votre environnement. Pour de plus amples informations, veuillez consulter the section called “Équilibreur de charge”.	`classic`	`classic`
			`application`
			`network`

LoadBalancerIsShared	Indique si l'équilibreur de charge de l'environnement est dédié ou partagé. Cette option ne peut être définie que pour un Application Load Balancer. Elle ne peut pas être modifiée après la création de l'environnement. Avec `false`, l'environnement dispose de son propre équilibreur de charge dédié, créé et géré par Elastic Beanstalk. Avec `true`, l'environnement utilise un équilibreur de charge partagé, créé par vous et spécifié dans l'option `SharedLoadBalancer` de l'espace de noms aws:elbv2:loadbalancer.	`false`	`true` `false`

aws:elasticbeanstalk:environment:process:default

Configurez le processus par défaut de votre environnement.

Espace de noms : **`aws:elasticbeanstalk:environment:process:default`**

Nom	Description	Par défaut	Valeurs valides
DeregistrationDelay	Délai d'attente, en secondes, d'achèvement des demandes actives avant l'annulat	`20`	`0` sur `3600`

Nom	Description	Par défaut	Valeurs valides
	ion de l'enregis trement.		
HealthCheckInterval	Intervalle, en secondes, pendant lequel Elastic Load Balancing vérifie l'état des instances Amazon EC2 de votre application.	Avec un équilibreur de charge classique ou d'application : `15` Avec un dispositi f d'équilibrage de charge de réseau : `30`	Avec un équilibreur de charge classique ou d'application : `5` à `300` Avec un dispositi f d'équilibrage de charge de réseau : `10`, `30`
HealthCheckPath	Chemin d'accès vers lequel les demandes HTTP pour les vérifications de l'état sont envoyées.	`/`	Un chemin d'accès routable.
HealthCheckTimeout	Délai d'attente, en secondes, d'une réponse pendant une vérification de l'état. Cette option est applicable uniquemen t aux environnements avec un équilibreur de charge d'application.	`5`	`1` sur `60`
HealthyThresholdCo unt	Nombre de demandes consécutives réussies avant qu'Elastic Load Balancing ne modifie le statut de l'état de l'instance.	Avec un équilibreur de charge classique ou d'application : `3` Avec un dispositi f d'équilibrage de charge de réseau : `5`	`2` sur `10`

Nom	Description	Par défaut	Valeurs valides
MatcherHTTPCode	Liste séparée par des virgules de codes HTTP qui indiquent qu'une instance est en bonne santé. Cette option est applicable uniquement aux environnements avec un dispositif d'équilibrage de charge de réseau ou de charge d'application.	`200`	Avec un équilibreur de charge d'application : `200` à `499` Avec un dispositif d'équilibrage de charge de réseau : `200` à `399`
Port	Port d'écoute du processus.	`80`	1 sur `65535`

Protocol	Protocole utilisé par le processus.
	
	Avec un équilibreur de charge d'application, vous pouvez uniquement définir cette option sur `HTTP` ou `HTTPS`.
	
	Avec un dispositif d'équilibrage de charge de réseau, vous pouvez uniquement définir cette option sur `TCP`.
		Avec un équilibreur de charge classique ou d'application : `HTTP`
		
		Avec un dispositif d'équilibrage de charge de réseau : `TCP`
			`TCP`
			
			`HTTP`
			
			`HTTPS`

Nom	Description	Par défaut	Valeurs valides

StickinessEnabled	Définissez la valeur sur true pour activer les séances permanentes.
	
	Cette option est applicable uniquement aux environnements avec un équilibreur de charge d'application.
		`'false'`	`'false'`
			
			`'true'`

StickinessLBCookieDuration	Durée de vie, en secondes, du cookie de séance permanente. Cette option est applicable uniquement aux environnements avec un équilibreur de charge d'application.	`86400` (une journée)	1 sur `604800`
StickinessType	Définissez la valeur sur `lb_cookie` pour utiliser des cookies pour les séances permanentes. Cette option est applicable uniquement aux environnements avec un équilibreur de charge d'application.	`lb_cookie`	`lb_cookie`

Nom	Description	Par défaut	Valeurs valides
UnhealthyThreshold Count	Nombre de demandes consécutives infructueuses avant qu'Elastic Load Balancing ne modifie le statut de l'état de l'instance.	5	2 sur 10

aws:elasticbeanstalk:environment:process:process_name

Configurez des processus supplémentaires pour votre environnement.

Espace de noms : **aws:elasticbeanstalk:environment:process:*process_name***

Nom	Description	Par défaut	Valeurs valides
DeregistrationDelay	Délai d'attente, en secondes, d'achèvement des demandes actives avant l'annulation de l'enregistrement.	20	0 sur 3600
HealthCheckInterval	Intervalle, en secondes, pendant lequel Elastic Load Balancing vérifie l'état des instances Amazon EC2 pour votre application.	Avec un équilibreur de charge classique ou d'application : 15 Avec un dispositif d'équilibrage de charge de réseau : 30	Avec un équilibreur de charge classique ou d'application : 5 à 300 Avec un dispositif d'équilibrage de charge de réseau : 10, 30
HealthCheckPath	Chemin d'accès vers lequel les demandes HTTP pour les vérifications de l'état sont envoyées.	/	Un chemin d'accès routable.

Nom	Description	Par défaut	Valeurs valides
HealthCheckTimeout	Délai d'attente, en secondes, d'une réponse pendant une vérification de l'état. Cette option est applicable uniquement aux environnements avec un équilibreur de charge d'application.	5	1 sur 60
HealthyThresholdCount	Nombre de demandes consécutives réussies avant qu'Elastic Load Balancing ne modifie le statut de l'état de l'instance.	Avec un équilibreur de charge classique ou d'application : 3 Avec un dispositif d'équilibrage de charge de réseau : 5	2 sur 10
MatcherHTTPCode	Liste séparée par des virgules de codes HTTP indiquant qu'une instance est saine. Cette option est applicable uniquement aux environnements avec un dispositif d'équilibrage de charge de réseau ou de charge d'application.	200	Avec un équilibreur de charge d'application : 200 à 499 Avec un dispositif d'équilibrage de charge de réseau : 200 à 399
Port	Port d'écoute du processus.	80	1 sur 65535

Nom	Description	Par défaut	Valeurs valides

Protocol	Protocole utilisé par le processus.
	
	Avec un équilibreur de charge d'application, vous pouvez uniquement définir cette option sur `HTTP` ou `HTTPS`.
	
	Avec un dispositif d'équilibrage de charge de réseau, vous pouvez uniquement définir cette option sur `TCP`.
		Avec un équilibreur de charge classique ou d'application : `HTTP`
		
		Avec un dispositif d'équilibrage de charge de réseau : `TCP`
			`TCP`
			
			`HTTP`
			
			`HTTPS`

StickinessEnabled	Définissez la valeur sur true pour activer les séances permanentes.
	
	Cette option est applicable uniquement aux environnements avec un équilibreur de charge d'application.
		`'false'`	`'false'`
			
			`'true'`

Nom	Description	Par défaut	Valeurs valides
StickinessLBCookie Duration	Durée de vie, en secondes, du cookie de séance permanente. Cette option est applicable uniquement aux environnements avec un équilibreur de charge d'application.	`86400` (une journée)	1 sur `604800`
StickinessType	Définissez la valeur sur `lb_cookie` pour utiliser des cookies pour les séances permanentes. Cette option est applicable uniquement aux environnements avec un équilibreur de charge d'application.	`lb_cookie`	`lb_cookie`
UnhealthyThreshold Count	Nombre de demandes consécutives infructueuses avant qu'Elastic Load Balancing ne modifie le statut de l'état de l'instance.	`5`	2 sur `10`

aws:elasticbeanstalk:environment:proxy:staticfiles

Vous pouvez utiliser l'espace de noms suivant pour configurer le serveur proxy afin de servir des fichiers statiques. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin

spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application. Cela réduit le nombre de demandes que votre application doit traiter.

Mappez un chemin servi par le serveur proxy à un dossier dans le code source qui contient les ressources statiques. Chaque option que vous définissez dans cet espace de noms mappe un chemin d'accès différent.

> **Note**
>
> Cet espace de noms s'applique aux branches de plateforme basées sur Amazon Linux 2 et versions ultérieures. Si votre environnement utilise une version de plateforme basée sur une AMI Amazon Linux (antérieure à Amazon Linux 2), reportez-vous à la section the section called “Options spécifiques à une plateforme” pour les espaces de noms de fichiers statiques spécifiques à la plateforme.

Espace de noms : **`aws:elasticbeanstalk:environment:proxy:staticfiles`**

Nom	Valeur
Chemin d'accès où le serveur proxy sert les fichiers. Démarrez la valeur par `/`. Par exemple, spécifiez `/images` pour traiter les fichiers au niveau de *`subdomain`* `.elasticbeanstalk.com/images` .	Nom du dossier contenant les fichiers. Par exemple, spécifiez `staticimages` pour traiter les fichiers depuis un dossier nommé `staticimages` au niveau supérieur de votre solution groupée de fichiers source.

aws:elasticbeanstalk:healthreporting:system

Configurez des rapports améliorés sur l'état pour votre environnement.

Espace de noms : **`aws:elasticbeanstalk:healthreporting:system`**

Nom	Description	Par défaut	Valeurs valides

SystemType	Système de rapport d'état (de base ou amélioré). Les rapports améliorés sur l'état de santé nécessitent un rôle de service, une version 2	`basic`	`basic`
			
			`enhanced`

Nom	Description	Par défaut	Valeurs valides
	de la plateforme ou une version de plateforme ultérieure. **Note** Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console et l'interface de ligne de commande (CLI) EB remplacent cette option par une valeur recommandée.		
ConfigDocument	Document JSON qui décrit l'environnement et les métriques d'instance sur lesquels publier CloudWatch.	Aucune	

EnhancedHealthAuthEnabled	Autorise l'API interne qu'Elastic Beanstalk utilise à communiquer des informations d'état améliorées depuis les instances de votre environnement vers le service Elastic Beanstalk.
	
	Pour plus d'informations, consultez the section called “Rôles d'intégrité améliorée”.
	
	Note
	Cette option ne s'applique qu'aux rapports améliorés sur l'état (c'est-à-dire quand `SystemType` est défini sur `enhanced`).
		`true`	`true`
			
			`false`

Nom	Description	Par défaut	Valeurs valides

HealthCheckSuccessThreshold	Abaisse le seuil de réussite des vérifications de l'état des instances.
	
	Note
	Si vous utilisez la console Elastic Beanstalk pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console remplace cette option par une valeur recommandée.
		`Ok`	`Ok`
			
			`Warning`
			
			`Degraded`
			
			`Severe`

aws:elasticbeanstalk:hostmanager

Configurez les instances EC2 dans votre environnement pour télécharger des journaux ayant subi une rotation dans Amazon S3.

Espace de noms : **`aws:elasticbeanstalk:hostmanager`**

Nom	Description	Par défaut	Valeurs valides

LogPublicationControl	Copie les fichiers journaux des instances Amazon EC2 de votre application dans le compartiment Amazon S3 associé à votre application.	`false`	`true`
			
			`false`

aws:elasticbeanstalk:managedactions

Configurez les mises à jour gérées de la plateforme pour votre environnement.

Espace de noms : **`aws:elasticbeanstalk:managedactions`**

Nom	Description	Par défaut	Valeurs valides

ManagedActionsEnabled	Activez Mises à jour de la plateforme gérée.
	
	Lorsque vous définissez cela sur `true`, vous devez également spécifier une `PreferredStartTime` et `UpdateLevel`.
		`false`	`true`
			
			`false`

PreferredStartTime	Configurez une fenêtre de maintenance pour les actions gérées au format UTC. Par exemple, `"Tue:09:00"`.	Aucune	Jour et heure au format *day*:*hour*:*minute* .
ServiceRoleForManagedUpdates	Nom d'un rôle IAM utilisé par Elastic Beanstalk pour effectuer des mises à jour de plateforme gérées pour votre environnement. Vous pouvez utiliser soit le même rôle que celui que vous avez spécifié pour l'option `ServiceRole` de l'espace de noms `aws:elasticbeanstalk:environment`, soit le rôle lié au service de mises à jour gérées de votre compte. Dans ce dernier cas, si le compte n'a pas encore de rôle lié au service de mises à jour	Aucune	Identique à `ServiceRole` ou `AWSServiceRoleForElasticBeanstalkManagedUpdates`

Nom	Description	Par défaut	Valeurs valides
	gérées, Elastic Beanstalk le crée.		

aws:elasticbeanstalk:managedactions:platformupdate

Configurez les mises à jour gérées de la plateforme pour votre environnement.

Espace de noms : **`aws:elasticbeanstalk:managedactions:platformupdate`**

Nom	Description	Par défaut	Valeurs valides
UpdateLevel	Le plus haut niveau de mise à jour à appliquer aux mises à jour de plateforme gérées. Les plateformes sont versionnées. *`major`* *`minor`*. *`patch`*. Par exemple, 2.0.8 dispose d'une version majeure de 2, d'une version mineure de 0 et d'une version corrective de 8.	Aucune	`patch` pour les mises à jour de version corrective uniquement. `minor` pour les mises à jour des deux versions : mineure et corrective.

InstanceRefreshEnabled	Activez le remplacement d'instance hebdomadaire.
	
	Nécessite la définition de `ManagedActionsEnabled` sur `true`.
		`false`	`true`
			
			`false`

aws:elasticbeanstalk:monitoring

Configurez votre environnement pour résilier les instances EC2 dont les vérifications d'état ont échoué.

Espace de noms : **`aws:elasticbeanstalk:monitoring`**

Nom	Description	Par défaut	Valeurs valides

Automatically Terminate Unhealthy Instances	Mettez fin à une instance en cas d'échec de vérifications d'intégrité.
	
	Note
	Cette option est uniquement prise en charge sur les environnements hérités. Il détermine l'état d'une instance en fonction de son accessibilité et d'autres métriques basées sur l'instance. Elastic Beanstalk ne permet pas d'arrêter automatiquement des instances en fonction de l'état de l'application.
		`true`	`true`
			
			`false`

aws:elasticbeanstalk:sns:topics

Configurez les notifications pour votre environnement.

Espace de noms : **`aws:elasticbeanstalk:sns:topics`**

Nom	Description	Par défaut	Valeurs valides
Notification Endpoint	Point de terminaison dans lequel vous souhaitez être informé des événements importants qui affectent votre application. **Note** Si vous utilisez la console Elastic Beanstalk pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console remplace cette option par une valeur recommandée.	Aucune	

Notification Protocol	Protocole utilisé pour envoyer des notifications à votre point de terminaison.	`email`	`http`
			
			`https`
			
			`email`
			
			`email-json`
			
			`sqs`

Notification Topic ARN	Amazon Resource Name (ARN) pour la rubrique à laquelle vous avez souscrit.	Aucune	
Notification Topic Name	Nom de la rubrique à laquelle vous avez souscrit.	Aucune	

aws:elasticbeanstalk:sqsd

Configurez la file d'attente Amazon SQS pour un environnement de travail.

Espace de noms : **`aws:elasticbeanstalk:sqsd`**

Nom	Description	Par défaut	Valeurs valides
WorkerQueueURL	URL de la file d'attente à partir de laquelle le démon de la couche d'environnement de travail lit les messages. **Note** Lorsque vous ne spécifiez pas de valeur, la file d'attente créée automatiquement par Elastic Beanstalk est une file d'attente Amazon SQS standard. Lorsque vous indiquez une valeur, vous pouvez fournir l'URL d'une file d'attente standard ou celle d'une file d'attente FIFO Amazon SQS. Sachez que si vous spécifiez une file d'attente FIFO, les tâches périodiques ne sont pas prises en charge.	génération automatique	Si vous ne spécifiez pas de valeur, Elastic Beanstalk crée automatiquement une file d'attente.
HttpPath	Chemin d'accès relatif à l'application vers laquelle les messages HTTP POST sont envoyés.	/	

Nom	Description	Par défaut	Valeurs valides

MimeType	Type MIME du message envoyé dans la demande HTTP POST.	`application/json`	`application/json`
			
			`application/x-www-form-urlencoded`
			
			`application/xml`
			
			`text/plain`
			
			Type MIME personnalisé.

HttpConnections	Nombre maximal de connexions simultanées à toute application au sein d'une instance Amazon EC2. **Note** Si vous utilisez la console Elastic Beanstalk pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console remplace cette option par une valeur recommandée.	`50`	1 sur `100`
ConnectTimeout	Délai d'attente, en secondes, de connexion réussie à une application.	`5`	1 sur `60`

Nom	Description	Par défaut	Valeurs valides
InactivityTimeout	Délai d'attente, en secondes, d'une réponse sur une connexion existante à une application. Le message est traité à nouveau jusqu'à ce que le démon reçoive une réponse 200 (OK) à partir de l'application dans la couche d'environnement de travail ou lorsque la `RetentionPeriod` expire.	`299`	`1` sur `36000`
VisibilityTimeout	La quantité de temps, en secondes, durant laquelle un message entrant à partir de la file d'attente Amazon SQS est verrouillé afin d'être traité. Une fois que la durée configurée est écoulée, alors le message est à nouveau rendu visible dans la file d'attente permettant à tout autre démon de le lire.	300	`0` sur `43200`
ErrorVisibilityTimeout	La quantité de temps, en secondes, qui s'écoule avant qu'Elastic Beanstalk renvoie un message à la file d'attente Amazon SQS après l'échec d'une tentative de traitement avec une erreur explicite.	2 secondes	`0` à `43200` secondes
RetentionPeriod	Délai d'attente, en secondes, pendant lequel un message est valide et activement traité.	`345600`	`60` sur `1209600`

Nom	Description	Par défaut	Valeurs valides
MaxRetries	Nombre maximum de tentatives d'envoi du message par Elastic Beanstalk à l'application web qui le traite avant de passer le message à la file d'attente de lettres mortes.	`10`	1 sur `100`

aws:elasticbeanstalk:trafficsplitting

Configurez les déploiements avec répartition du trafic pour votre environnement.

Cet espace de noms s'applique lorsque vous définissez l'option `DeploymentPolicy` de l'espace de noms aws:elasticbeanstalk:command sur `TrafficSplitting`. Pour plus d'informations sur les politiques de déploiement, consultez the section called “Options de déploiement”.

Espace de noms : **`aws:elasticbeanstalk:trafficsplitting`**

Nom	Description	Par défaut	Valeurs valides
NewVersionPercent	Pourcentage initial du trafic client entrant qu'Elastic Beanstalk déplace vers les instances d'environnement exécutant la nouvelle version de l'application que vous déployez.	`10`	1 sur `100`
EvaluationTime	Période, en minutes, qu'Elastic Beanstalk attend après un déploiement sain initial avant de déplacer tout le trafic client entrant vers la nouvelle version de l'application que vous déployez.	`5`	3 sur `600`

aws:elasticbeanstalk:xray

Exécutez le AWS X-Ray daemon pour relayer les informations de suivi depuis votre application X-Ray intégrée.

Espace de noms : **`aws:elasticbeanstalk:xray`**

Nom	Description	Par défaut	Valeurs valides

`XRayEnabled`	Définissez sur `true` pour exécuter le démon X-Ray sur les instances dans votre environnement.	`false`	`true`
			
			`false`

aws:elb:healthcheck

Configurez les vérifications de l'état pour un Classic Load Balancer.

Espace de noms : **`aws:elb:healthcheck`**

Nom	Description	Par défaut	Valeurs valides
HealthyThreshold	Nombre de demandes consécutives réussies avant qu'Elastic Load Balancing ne modifie le statut de l'état de l'instance.	`3`	2 sur `10`
Interval	Intervalle pendant lequel Elastic Load Balancing vérifie l'état des instances Amazon EC2 de votre application.	`10`	5 sur `300`
Timeout	Délai d'attente, en secondes, pendant lequel Elastic Load Balancing attend une réponse avant de considérer l'instance non réactive.	`5`	2 sur `60`
Unhealthy Threshold	Nombre de demandes consécutives infructueuses avant qu'Elastic Load Balancing ne modifie le statut de l'état de l'instance.	`5`	2 sur `10`
(obsolète) Target	Destination d'une instance backend vers laquelle les vérifications de l'état sont envoyées. Utilisez	`TCP:80`	Cible au format

Nom	Description	Par défaut	Valeurs valides
	`Application Healthcheck URL` dans l'espace de noms `aws:elasticbeanstalk:application` à la place.		*PROTOCOL* : *PORT* / *PATH*

aws:elb:loadbalancer

Configurez le Classic Load Balancer de votre environnement.

Plusieurs options de cet espace de noms ne sont plus prises en charge et sont remplacées par des options spécifiques à l'écouteur dans l'espace de noms aws:elb:listener. Avec ces options qui ne sont plus prises en charge, vous ne pouvez configurer que deux écouteurs (un sécurisé et un non sécurisé) sur les ports standard.

Espace de noms : **`aws:elb:loadbalancer`**

Nom	Description	Par défaut	Valeurs valides

CrossZone	Configurez l'équilibreur de charge pour acheminer le trafic uniformément sur toutes les instances dans toutes les zones de disponibilité et non uniquement dans chaque zone.
	
	Note
	Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console et l'interface de ligne de commande
		`false`	`true`
			
			`false`

Nom	Description	Par défaut	Valeurs valides
	EB CLI remplacent cette option par une valeur recommandée.		
SecurityGroups	Attribuez un ou plusieurs groupes de sécurité que vous avez créés à l'équilibreur de charge. Obligatoire si `DisableDefaultEC2SecurityGroup` (aws:autoscaling:launchconfiguration) est défini sur `true`. Les environnements à charge équilibrée qui ont choisi de ne pas participer au groupe de sécurité Elastic Beanstalk EC2 par défaut doivent proposer cette option à un ou plusieurs groupes de sécurité. Pour plus d'informations, consultez Gestion des groupes de sécurité EC2.	Aucune	Un ou plusieurs groupes de sécurité IDs.
ManagedSecurityGroup	Attribuez un groupe de sécurité existant à l'équilibreur de charge de votre environnement au lieu d'en créer un. Pour utiliser ce paramètre, mettez à jour le paramètre `SecurityGroups` de cet espace de noms afin d'inclure l'ID de votre groupe de sécurité et supprimez l'ID du groupe de sécurité créé automatiquement, le cas échéant. Pour autoriser le trafic à partir de l'équilibreur de charge vers les instances EC2 de votre environnement, Elastic Beanstalk ajoute une règle au groupe de sécurité des instances, qui autorise le trafic entrant depuis le groupe de sécurité géré.	Aucune	Un ID de groupe de sécurité.

Nom	Description	Par défaut	Valeurs valides
(obsolète) LoadBalancerHTTPPort	Port d'écoute pour l'écouteur non sécurisé.	`80`	`OFF` `80`
(obsolète) LoadBalancerPortProtocol	Protocole à utiliser sur l'écouteur non sécurisé.	`HTTP`	`HTTP` `TCP`

(obsolète) LoadBalancerHTTPSPort	Port d'écoute pour l'écouteur sécurisé.	`OFF`	`OFF`
			`443`
			`8443`

(obsolète) LoadBalancerSSLPortProtocol	Protocole à utiliser sur l'écouteur sécurisé.	`HTTPS`	`HTTPS` `SSL`
(obsolète) SSLCertificateId	Amazon Resource Name (ARN) d'un certificat SSL à lier à l'écouteur sécurisé.	Aucune	

aws:elb:listener

Configurez l'écouteur par défaut (port 80) sur un Classic Load Balancer.

Espace de noms : **`aws:elb:listener`**

Nom	Description	Par défaut	Valeurs valides
ListenerProtocol	Le protocole utilisé par l'écouteur.	`HTTP`	`HTTP TCP`
InstancePort	Le port que cet écouteur utilise pour communiquer avec les instances EC2.	80	1 sur `65535`
InstanceProtocol	Le protocole que cet écouteur utilise pour communiquer avec les instances EC2.	`HTTP` lorsque `ListenerP`	`HTTP` ou `HTTPS` lorsque

Nom	Description	Par défaut	Valeurs valides

	Il doit se trouver sur la même couche de protocole Internet que le `ListenerProtocol`. Il doit avoir également le même niveau de sécurité que tout autre écouteur utilisant le même `InstancePort` que cet écouteur.
	
	Par exemple, si le `ListenerProtocol` est `HTTPS` (couche d'application, via une connexion sécurisée), vous pouvez définir le `InstanceProtocol` sur `HTTP` (également au niveau de la couche d'application, via une connexion non sécurisée). En outre, si vous définissez le `InstancePort` sur `80`, vous devez définir le `InstanceProtocol` sur `HTTP` dans tous les autres écouteurs avec le `InstancePort` défini sur `80`.
		`rotocol` est `HTTP`
		
		`TCP` lorsque `ListenerProtocol` est `TCP`
			`ListenerProtocol` est `HTTP` ou `HTTPS`
			
			`TCP` ou `SSL` lorsque `ListenerProtocol` est `TCP` ou `SSL`

PolicyNames	Une liste de noms de politique séparés par des virgules à appliquer au port pour cet écouteur. Nous vous recommandons d'utiliser plutôt l' LoadBalancerPorts option de l'espace de aws:elb:policies noms.	Aucune	

ListenerEnabled	Indique si cet écouteur est activé. Si vous spécifiez `false`, l'écouteur n'est pas inclus dans l'équilibreur de charge.	`true`	`true`
			
			`false`

aws:elb:listener:listener_port

Configurez des écouteurs supplémentaires sur un Classic Load Balancer.

Espace de noms : **`aws:elb:listener:`*`listener_port`***

Nom	Description	Par défaut	Valeurs valides
ListenerProtocol	Le protocole utilisé par l'écouteur.	`HTTP`	`HTTP` `HTTPS` `TCP` `SSL`
InstancePort	Le port que cet écouteur utilise pour communiquer avec les instances EC2.	Le même que *`listener_port`* .	1 sur `65535`

InstanceProtocol	Le protocole que cet écouteur utilise pour communiquer avec les instances EC2.
	
	Il doit se trouver sur la même couche de protocole Internet que le `ListenerProtocol` . Il doit avoir également le même niveau de sécurité que tout autre écouteur utilisant le même `InstancePort` que cet écouteur.
	
	Par exemple, si le `ListenerProtocol` est `HTTPS` (couche d'application, via une connexion sécurisée), vous pouvez définir le `InstanceProtocol` sur `HTTP` (également au niveau de la couche d'application, via une connexion non sécurisée). En outre, si vous définissez le `InstancePort` sur `80`, vous devez définir le `InstanceProtocol` sur `HTTP` dans tous les autres écouteurs avec le `InstancePort` défini sur `80`.
		`HTTP` lorsque `ListenerProtocol` est `HTTP` ou `HTTPS`
		
		`TCP` lorsque `ListenerProtocol` est `TCP` ou `SSL`
			`HTTP` ou `HTTPS` lorsque `ListenerProtocol` est `HTTP` ou `HTTPS`
			
			`TCP` ou `SSL` lorsque `ListenerProtocol` est `TCP` ou `SSL`

PolicyNames	Une liste de noms de politique séparés par des virgules à appliquer au port pour cet écouteur. Nous vous suggérons d'utilise	Aucune	

Nom	Description	Par défaut	Valeurs valides
	r plutôt l' LoadBalancerPorts option de l'espace de aws:elb:policies noms.		
SSLCertificateId	Amazon Resource Name (ARN) d'un certificat SSL à lier au processus d'écoute.	Aucune	
ListenerEnabled	Indique si cet écouteur est activé. Si vous spécifiez `false`, l'écouteur n'est pas inclus dans l'équilibreur de charge.	`true` si une autre option est définie. `false` dans le cas contraire.	`true` `false`

aws:elb:policies

Modifiez l'adhérence par défaut et les politiques globales d'équilibreur de charge pour un Classic Load Balancer.

Espace de noms : **`aws:elb:policies`**

Nom	Description	Par défaut	Valeurs valides

ConnectionDraining Enabled	Spécifie si l'équilibreur de charge gère les connexions existantes vers les instances qui sont devenues défectueuses ou désinscrites pour terminer les requêtes en cours.
	
	Note
	Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour
		`false`	`true`
			
			`false`

Nom	Description	Par défaut	Valeurs valides
	créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console et l'interface de ligne de commande EB CLI remplacent cette option par une valeur recommandée.		
ConnectionDrainingTimeout	Le nombre maximum de secondes pendant lesquels l'équilibreur de charge maintient des connexions existantes sur une instance au cours du drainage de la connexion avant de forcer la fermeture des connexions. **Note** Si vous utilisez la console Elastic Beanstalk pour créer un environnement, vous ne pouvez pas définir cette option dans un fichier de configuration. La console remplace cette option par une valeur recommandée.	`20`	1 sur `3600`
ConnectionSettingIdleTimeout	Durée, en secondes, pendant laquelle l'équilibreur de charge attend que les données soit envoyées ou reçues via la connexion. Si aucune donnée n'a été envoyée ou reçue une fois cette période écoulée, l'équilibreur de charge ferme la connexion.	`60`	1 sur `3600`

Nom	Description	Par défaut	Valeurs valides
LoadBalancerPorts	Une liste séparée par des virgules des ports d'écoute auxquels la politique par défaut (`AWSEB-ELB-StickinessPolicy`) s'applique.	Aucune	Vous pouvez utiliser `:all` pour indiquer tous les ports d'écoute
Stickiness Cookie Expiration	La quantité de temps, en secondes, durant laquelle chaque cookie est valide. Utilise la politique par défaut (`AWSEB-ELB-StickinessPolicy`).	`0`	`0` sur `1000000`
Stickiness Policy	Lie la séance d'un utilisateur à une instance de serveur spécifique afin que toutes les demandes provenant de l'utilisateur pendant la séance soient envoyées à la même instance de serveur. Utilise la politique par défaut (`AWSEB-ELB-StickinessPolicy`).	`false`	`true false`

aws:elb:policies:policy_name

Créez des politiques globales d'équilibreur de charge supplémentaires pour un Classic Load Balancer.

Espace de noms : **`aws:elb:policies:`*`policy_name`***

Nom	Description	Par défaut	Valeurs valides
CookieName	Le nom du cookie généré par l'application qui contrôle les durées de vie de la séance d'une politique `AppCookieStickinessPolicyType`. Cette politique ne	Aucune	

Nom	Description	Par défaut	Valeurs valides
	peut être associée qu'aux HTTP/HTTPS auditeurs.		
InstancePorts	Une liste des ports d'instance séparés par des virgules auxquels cette politique s'applique.	Aucune	Une liste de ports ou `:all`
LoadBalancerPorts	Une liste des ports d'écoute séparés par des virgules auxquels cette politique s'applique.	Aucune	Une liste de ports ou `:all`
ProxyProtocol	Pour une politique `ProxyProtocolPolicyType`, spécifie s'il convient d'inclure l'adresse IP et le port de la demande d'origine pour les messages TCP. Cette politique ne peut être associée qu'aux TCP/SSL auditeurs.	Aucune	`true false`
PublicKey	Le contenu d'une clé publique pour une politique `PublicKeyPolicyType` à utiliser lors de l'authentification du ou des serveurs backend. Cette politique ne peut pas être appliquée directement aux serveurs backend ou aux écouteurs. Elle doit faire partie d'une politique `BackendServerAuthenticationPolicyType`.	Aucune	

Nom	Description	Par défaut	Valeurs valides
PublicKeyPolicyNames	Liste séparée par des virgules de noms de stratégie (à partir des politiques `PublicKeyPolicyType`) pour une politique `BackendServerAuthenticationPolicyType` qui contrôle l'authentification sur un ou plusieurs serveurs backend. Cette politique peut être associée uniquement à des serveurs backend qui utilisent HTTPS/SSL.	Aucune	
SSLProtocols	Une liste séparée par des virgules de protocoles SSL devant être activés pour une politique `SSLNegotiationPolicyType` qui définit les chiffrements et les protocoles acceptés par l'équilibreur de charge. Cette politique ne peut être associée qu'aux HTTPS/SSL auditeurs.	Aucune	
SSLReferencePolicy	Nom d'une politique de sécurité prédéfinie qui respecte les meilleures pratiques de AWS sécurité et que vous souhaitez activer pour une `SSLNegotiationPolicyType` politique qui définit les chiffrements et les protocoles acceptés par l'équilibreur de charge. Cette politique ne peut être associée qu'aux HTTPS/SSL auditeurs.	Aucune	
Stickiness Cookie Expiration	La quantité de temps, en secondes, durant laquelle chaque cookie est valide.	`0`	`0` sur `1000000`

Nom	Description	Par défaut	Valeurs valides
Stickiness Policy	Lie la séance d'un utilisateur à une instance de serveur spécifique afin que toutes les demandes provenant de l'utilisateur pendant la séance soient envoyées à la même instance de serveur.	`false`	`true false`

aws:elbv2:listener:default

Configurez l'écouteur par défaut (port 80) sur un Application Load Balancer ou un Network Load Balancer.

Cet espace de noms ne s'applique pas à un environnement qui utilise un équilibreur de charge partagé. Les équilibreurs de charge partagés n'ont pas d'écouteur par défaut.

Espace de noms : **`aws:elbv2:listener:default`**

Nom	Description	Par défaut	Valeurs valides
DefaultProcess	Nom du processus vers lequel transférer le trafic lorsqu'aucune règle ne correspond à la demande.	`default`	Un nom de processus.
ListenerEnabled	Définissez la valeur sur `false` pour désactiver l'écouteur. Vous pouvez utiliser cette option pour désactiver l'écouteur par défaut sur le port 80.	`true`	`true` `false`

Nom	Description	Par défaut	Valeurs valides
Protocol	Protocole du trafic à traiter.	Avec un équilibreur de charge d'application : `HTTP` Avec un dispositif d'équilibrage de charge de réseau : `TCP`	Avec un équilibreur de charge d'application : `HTTP`, `HTTPS` Avec un dispositif d'équilibrage de charge de réseau : `TCP`
Rules	Liste des règles à appliquer à l'écouteur Cette option est applicable uniquement aux environnements avec un équilibreur de charge Application Load Balancer.	Aucune	Liste séparée par des virgules de noms de règles.
SSLCertificateArns	Amazon Resource Name (ARN) du certificat SSL à lier à l'écouteur. Cette option est applicable uniquement aux environnements avec un équilibreur de charge Application Load Balancer.	Aucune	L'ARN d'un certificat stocké dans IAM ou ACM.

Nom	Description	Par défaut	Valeurs valides
SSLPolicy	Spécifiez une politique de sécurité à appliquer à l'écouteur. Cette option est applicable uniquement aux environnements avec un équilibreur de charge Application Load Balancer.	Aucune (valeur par défaut ELB)	Le nom d'une politique de sécurité d'équilibreur de charge.

aws:elbv2:listener:listener_port

Configurez des écouteurs supplémentaires sur un Application Load Balancer ou un Network Load Balancer.

> **Note**
>
> Pour un Application Load Balancer partagé, vous ne pouvez spécifier que l'option `Rule`. Les autres options ne s'appliquent pas aux équilibreurs de charge partagés.

Espace de noms : **`aws:elbv2:listener:`*`listener_port`***

Nom	Description	Par défaut	Valeurs valides
DefaultProcess	Nom du processus vers lequel transférer le trafic lorsqu'aucune règle ne correspond à la demande.	`default`	Un nom de processus.

ListenerEnabled	Définissez la valeur sur `false` pour désactiver l'écouteur. Vous pouvez utiliser	`true`	`true`
			
			`false`

Nom	Description	Par défaut	Valeurs valides
	cette option pour désactiver l'écouteur par défaut sur le port 80.		
Protocol	Protocole du trafic à traiter.	Avec un équilibreur de charge d'application : `HTTP` Avec un dispositif d'équilibrage de charge de réseau : `TCP`	Avec un équilibreur de charge d'application : `HTTP`, `HTTPS` Avec un dispositif d'équilibrage de charge de réseau : `TCP`
Rules	Liste des règles à appliquer à l'écouteur Cette option est applicable uniquement aux environnements avec un Application Load Balancer. Si votre environnement utilise un Application Load Balancer partagé et que vous ne spécifiez pas cette option pour un processus d'écoute, Elastic Beanstalk associe automatiquement la règle `default` à un écouteur de port 80.	Aucune	Liste séparée par des virgules de noms de règles.

Nom	Description	Par défaut	Valeurs valides
SSLCertificateArns	Amazon Resource Name (ARN) du certificat SSL à lier à l'écouteur. Cette option est applicable uniquement aux environnements avec un équilibreur de charge Application Load Balancer.	Aucune	L'ARN d'un certificat stocké dans IAM ou ACM.
SSLPolicy	Spécifiez une politique de sécurité à appliquer à l'écouteur. Cette option est applicable uniquement aux environnements avec un équilibreur de charge Application Load Balancer.	Aucune (valeur par défaut ELB)	Le nom d'une politique de sécurité d'équilibreur de charge.

aws:elbv2:listenerrule:rule_name

Définissez les règles d'écouteur pour un Application Load Balancer. Si une requête correspond aux noms d'hôte ou aux chemins dans une règle, l'équilibreur de charge les transfère vers le processus en question. Pour utiliser une règle, ajoutez-la à un écouteur avec l'option `Rules` dans le namespace `aws:elbv2:listener:listener_port`.

> **Note**
>
> Cet espace de noms n'est pas applicable aux environnements avec un dispositif d'équilibrage de charge de réseau.

Espace de noms : **`aws:elbv2:listenerrule:`*`rule_name`***

Nom	Description	Par défaut	Valeurs valides
HostHeaders	Liste de noms d'hôte à faire correspondre. Par exemple, `my.example.com` .	Équilibreur de charge dédié : Aucun Équilibreur de charge partagé : CNAME de l'environnement	Chaque nom peut contenir jusqu'à 128 caractères. Un modèle peut inclure à la fois des lettres majuscules et minuscules, des chiffres, des traits d'union (–) et jusqu'à trois caractères génériques (* correspond à zéro ou plus ; ? correspond exactement à un caractère). Vous pouvez répertorier plusieurs noms, tous séparés par une virgule. L'Application Load Balancer prend en charge jusqu'à cinq règles `HostHeader` et `PathPattern` combinées. Pour plus d'informations, consultez Conditions de l'hôte dans le Guide de l'utilisateur des Application Load Balancers.

Nom	Description	Par défaut	Valeurs valides
PathPatterns	Modèles de chemin d'accès à faire correspondre (par exemple, `/img/*`). Cette option est applicable uniquement aux environnements avec un équilibreur de charge d'application.	Aucune	Chaque modèle peut contenir jusqu'à 128 caractères. Un modèle peut inclure des lettres majuscules et minuscules, des chiffres, des tirets (–) et jusqu'à trois caractères génériques (* correspond à zéro ou plus ; ? correspond exactement à un caractère). Vous pouvez ajouter plusieurs modèles de chemin d'accès séparés par des virgules. L'Application Load Balancer prend en charge jusqu'à cinq règles `HostHeader` et `PathPattern` combinées. Pour plus d'informations, consultez Conditions de chemin d'accès dans le Guide de l'utilisateur des Application Load Balancers.

Nom	Description	Par défaut	Valeurs valides
Priority	Priorité de cette règle lorsque plusieurs règles correspondent. Le nombre inférieur est prioritaire. Deux règles ne peuvent pas avoir la même priorité. Avec un équilibreur de charge partagé, Elastic Beanstalk traite les priorités des règles comme relatives dans les environnements de partage et les mappe aux priorités absolues lors de la création.	1	1 sur `1000`
Process	Nom du processus vers lequel transférer le trafic lorsque cette règle correspond à la demande.	`default`	Un nom de processus.

aws:elbv2:loadbalancer

Configurez un Application Load Balancer.

Pour un équilibreur de charge partagé, seules les options `SharedLoadBalancer` et `SecurityGroups` sont valides.

> **Note**
>
> Cet espace de noms n'est pas applicable aux environnements avec un Network Load Balancer.

Espace de noms : **`aws:elbv2:loadbalancer`**

Nom	Description	Par défaut	Valeurs valides
AccessLogsS3Bucket	Compartiment Amazon S3 où les journaux d'accès sont stockés. Le compartiment doit se trouver dans la	Aucune	Un nom de compartiment.

Nom	Description	Par défaut	Valeurs valides
	même région que l'environnement et autoriser l'accès en écriture à l'équilibreur de charge.		
AccessLogsS3Enabled	Activez le stockage des journaux d'accès.	`false`	`true` `false`
AccessLogsS3Prefix	Préfixe à ajouter aux noms des journaux d'accès. Par défaut, l'équilibreur de charge télécharge les journaux dans un répertoire nommé AWSLogs dans le compartiment que vous spécifiez. Spécifiez un préfixe pour placer le AWSLogs répertoire dans un autre répertoire.	Aucune	
IdleTimeout	Délai d'attente, en secondes, de la fin d'une demande avant de fermer les connexions au client et à l'instance.	Aucune	1 sur `3600`

Nom	Description	Par défaut	Valeurs valides
IpAddressType	Spécifie la configuration du format d'adresse IP pour l'équilibreur de charge de l'environnement. Utilisez l'option dualstack pour activer le IPv6 protocole, ainsi que le protocole. IPv4 Seuls les équilibreurs de charge d'application et les équilibreurs de charge réseau prennent en charge l'option dualstack. Les environne ments à instance unique et les environnements qui utilisent des équilibreurs de charge classiques ne le prennent pas en charge. Vous devez associer le VPC de l'environnement et tous les sous-rése aux VPC à des blocs IPv6 CIDR pour prendre en charge l'option dualstack. Pour de plus amples informations, veuillez consulter the section called “Configuration de la double pile”.	ipv4	ipv4, double pile

Nom	Description	Par défaut	Valeurs valides
ManagedSecurityGroup	Attribuez un groupe de sécurité existant à l'équilibreur de charge de votre environnement au lieu d'en créer un. Pour utiliser ce paramètre, mettez à jour le paramètre `SecurityGroups` dans cet espace de noms afin d'inclure l'ID de votre groupe de sécurité et de supprimer l'ID du groupe de sécurité créé automatiquement, le cas échéant. Pour autoriser le trafic de l'équilibreur de charge vers les instances EC2 de votre environnement, Elastic Beanstalk ajoute une règle au groupe de sécurité des instances, qui autorise le trafic entrant depuis le groupe de sécurité géré.	Le groupe de sécurité qu'Elastic Beanstalk crée pour votre équilibreur de charge.	Un ID de groupe de sécurité.

Nom	Description	Par défaut	Valeurs valides
SecurityGroups	Liste des groupes de sécurité à lier à l'équilibreur de charge. Obligatoire si `DisableDefaultEC2SecurityGroup` (aws:autoscaling:launchconfiguration) est défini sur `true`. Les environnements à charge équilibrée qui ont choisi de ne pas participer au groupe de sécurité Elastic Beanstalk EC2 par défaut doivent proposer cette option à un ou plusieurs groupes de sécurité. Pour plus d'informations, consultez Gestion des groupes de sécurité EC2. Pour un équilibreur de charge partagé, si vous ne spécifiez pas cette valeur, Elastic Beanstalk vérifie si un groupe de sécurité existant qu'il gère est déjà attaché à l'équilibreur de charge. Si aucun groupe n'est lié à l'équilibreur de charge, Elastic Beanstalk crée un groupe de sécurité et le lie à l'équilibreur de charge. Elastic Beanstalk supprime ce groupe de sécurité lorsque le dernier environnement partageant l'équilibreur de charge s'arrête. Les groupes de sécurité de l'équilibreur de charge sont utilisés pour configurer la règle d'entrée du groupe	Le groupe de sécurité qu'Elastic Beanstalk crée pour votre équilibreur de charge.	Liste des groupes de sécurité séparés par des virgules. IDs

Nom	Description	Par défaut	Valeurs valides
	de sécurité de l'instance Amazon EC2.		

Nom	Description	Par défaut	Valeurs valides
SharedLoadBalancer	Amazon Resource Name (ARN) de l'équilibreur de charge partagé. Cette option s'applique uniquement à un Application Load Balancer. Elle est requise lorsque l'option `LoadBalancerIsShared` de l'espace de noms aws:elasticbeanstalk:environment est définie sur `true`. Vous ne pouvez pas modifier l'ARN de l'équilibreur de charge partagé après la création de l'environnement. Critères d'une valeur valide : • Il doit s'agir d'un équilibreur de charge valide et actif dans la AWS région où se trouve l'environnement. • Il doit se trouver dans le même Amazon Virtual Private Cloud (Amazon VPC) que l'environnement. • Il ne peut pas s'agir d'un équilibreur de charge créé par Elastic Beanstalk en tant qu'équilibreur de charge dédié pour un autre environnement. Vous pouvez identifier ces équilibreurs de charge dédiés par le préfixe `awseb-`. Exemple :	Aucune	ARN d'un équilibreur de charge valide répondant aux critères décrits ici.

Nom	Description	Par défaut	Valeurs valides
	`arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/app/FrontEndLB/0dbf78d8ad96abbc`		

aws:rds:dbinstance

Configurez une instance de base de données Amazon RDS attachée.

Espace de noms : **`aws:rds:dbinstance`**

Nom	Description	Par défaut	Valeurs valides
DBAllocatedStorage	La taille de stockage de base de données allouée, spécifiée en giga-octets.	MySQL: `5` Oracle: `10` sqlserver-se: `200` sqlserver-ex: `30` sqlserver-web: `30`	MySQL: `5-1024` Oracle: `10-1024` sqlserver: ne peut pas être modifié

DBDeletionPolicy	Spécifie s'il faut conserver, supprimer ou créer un instantané de l'instance de base de données lors de la résiliation d'un environnement.
	Cette option fonctionne conjointement avec `HasCoupledDatabase` , qui est également une option de cet espace de noms.
		`Delete`	`Delete`
			`Retain`
			`Snapshot`

Nom	Description	Par défaut	Valeurs valides
	Warning La suppression d'une instance DB entraîne une perte permanente de données.		

DBEngine	Le nom du moteur de base de données à utiliser pour cette instance.	`mysql`	`mysql`
			`oracle-se1`
			`sqlserver-ex`
			`sqlserver-web`
			`sqlserver-se`
			`postgres`

DBEngineVersion	Le numéro de version du moteur de base de données.	`5.5`	
DBInstanceClass	Le type de l'instance de base de données	`db.t2.micro` (`db.m1.large` pour un environnement qui ne s'exécute pas dans un VPC Amazon)	Pour plus d'informations, consultez la section Classes d'instances de bases de données dans le Guide de l'utilisateur Amazon Relational Database Service.

Nom	Description	Par défaut	Valeurs valides
DBPassword	Le nom du mot de passe utilisateur principal pour l'instance de base de données.	Aucune	
DBSnapshotIdentifier	L'identificateur du Snapshot DB d'origine à partir duquel effectuer la restauration.	Aucune	
DBUser	Le nom de l'utilisateur principal pour l'instance DB.	ebroot	

Nom	Description	Par défaut	Valeurs valides

HasCoupledDatabase	Spécifie si une instance de base de données est couplée à votre environnement. Si elle bascule sur `true`, Elastic Beanstalk crée une nouvelle instance de base de données couplée à votre environnement. Si elle bascule sur `false`, Elastic Beanstalk initie le découplage de l'instance de base de données de votre environnement.
	
	Cette option fonctionne conjointement avec `DBDeletionPolicy` , qui est également une option de cet espace de noms.
	
	Note
	Remarque : si vous basculez cette valeur sur `true` après avoir découplé la base de données précédente, Elastic Beanstalk crée une nouvelle base de données avec les paramètres d'option de base de données précédents. Toutefois, pour maintenir la sécurité de votre environnement, il ne conserve pas les paramètres `DBUser` et `DBPassword` existants. Vous devez à nouveau spécifier `DBUser` et `DBPassword` .
		`false`	`true`
			
			`false`

Nom	Description	Par défaut	Valeurs valides

MultiAZDatabase	Spécifie si un déploiement multi-AZ d'une instance de base de données doit être créé. Pour plus d'informations sur les déploiements multi-AZ avec Amazon Relational Database Service (RDS), consultez Régions et zones de disponibilité dans le Guide de l'utilisateur Amazon Relational Database Service.	`false`	`true`
			
			`false`

Options spécifiques à une plateforme

Certaines plateformes Elastic Beanstalk définissent des espaces de noms d'options spécifiques à la plateforme. Ces espaces de noms et leurs options sont répertoriés ci-dessous pour chaque plateforme.

> **Note**
>
> Auparavant, dans les versions de plateforme basées sur l'AMI Amazon Linux (antérieure à Amazon Linux 2), les deux fonctionnalités suivantes et leurs espaces de noms respectifs étaient considérés comme des fonctionnalités spécifiques à la plateforme et étaient répertoriés ici par plateforme :
>
> - Configuration du proxy pour les fichiers statiques – `aws:elasticbeanstalk:environment:proxy:staticfiles`
> - AWS X-Ray soutien — `aws:elasticbeanstalk:xray`
>
> Dans les versions de plateforme Amazon Linux 2, Elastic Beanstalk implémente ces fonctionnalités de manière cohérente sur toutes les plateformes de prise en charge. L'espace de noms associé est désormais répertorié dans la page the section called “Options générales”. Nous n'en avons gardé mention sur cette page que pour les plates-formes qui avaient des espaces de noms différents.

Plates-formes

- Options de la plateforme Docker
- Options de la plateforme Go
- Options de la plateforme Java SE
- Java avec options de plateforme Tomcat
- Historique de la plateforme .NET Core sous Linux
- Options de la plateforme .NET
- Options de plateforme Node.js
- Options de la plateforme PHP
- Options de la plateforme Python
- Options de la plateforme Ruby

Options de la plateforme Docker

Les options de configuration suivantes spécifiques à Docker s'appliquent aux plateformes Docker et Docker préconfigurées.

> **Note**
>
> Ces options de configuration ne s'appliquent pas à
>
> - Plateforme Docker (Amazon Linux 2) avec Docker Compose
> - Plateforme Docker multiconteneur AL1 (AMI Amazon Linux) : cette plateforme a été retirée

Espace de noms : **`aws:elasticbeanstalk:environment:proxy`**

Nom	Description	Par défaut	Valeurs valides
ProxyServer	Spécifie le serveur web à utiliser comme proxy.	`nginx`	`nginx` `none`— Amazon Linux AM et Docker uniquement w/DC

Options de la plateforme Go

Options de plateforme AMI Amazon Linux (pré-Amazon Linux 2)

Espace de nom : **`aws:elasticbeanstalk:container:golang:staticfiles`**

Vous pouvez utiliser l'espace de noms suivant pour configurer le serveur proxy afin de servir des fichiers statiques. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application. Cela réduit le nombre de demandes que votre application doit traiter.

Mappez un chemin servi par le serveur proxy à un dossier dans le code source qui contient les ressources statiques. Chaque option que vous définissez dans cet espace de noms mappe un chemin d'accès différent.

Nom	Valeur
Chemin d'accès où le serveur proxy va servir les fichiers. Exemple : `/images` pour traiter les fichiers à l'emplacement *`subdomain`* `.elasticbeanstalk.com/images` .	Nom du dossier contenant les fichiers. Exemple : `staticimages` pour traiter les fichiers depuis un dossier nommé `staticimages` au niveau supérieur de votre bundle de fichiers source.

Options de la plateforme Java SE

Options de plateforme AMI Amazon Linux (pré-Amazon Linux 2)

Espace de nom : **`aws:elasticbeanstalk:container:java:staticfiles`**

Vous pouvez utiliser l'espace de noms suivant pour configurer le serveur proxy afin de servir des fichiers statiques. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application. Cela réduit le nombre de demandes que votre application doit traiter.

Mappez un chemin servi par le serveur proxy à un dossier dans le code source qui contient les ressources statiques. Chaque option que vous définissez dans cet espace de noms mappe un chemin d'accès différent.

Nom	Valeur
Chemin d'accès où le serveur proxy va servir les fichiers. Exemple : `/images` pour traiter les fichiers à l'emplacement *`subdomain`* `.elasticbeanstalk.com/images` .	Nom du dossier contenant les fichiers. Exemple : `staticimages` pour traiter les fichiers depuis un dossier nommé `staticimages` au niveau supérieur de votre bundle de fichiers source.

Java avec options de plateforme Tomcat

Espace de noms : **`aws:elasticbeanstalk:application:environment`**

Nom	Description	Par défaut	Valeurs valides
JDBC_CONNECTION_STRING	La chaîne de connexion à une base de données externe.	Non applicable	Non applicable

Pour plus d'informations, consultez Variables d'environnement et autres paramètres du logiciel.

Espace de noms : **`aws:elasticbeanstalk:container:tomcat:jvmoptions`**

Nom	Description	Par défaut	Valeurs valides
JVM Options	Passez des options de ligne de commande à la JVM au démarrage.	Non applicable	Non applicable
Xmx	Tailles maximum des segments de mémoire de la JVM.	`256m`	Non applicable
XX:MaxPermSize	Section du segment de mémoire de la JVM qui est utilisée pour stocker des définitions de classe et des métadonnées associées.	`64m`	N/A

Nom	Description	Par défaut	Valeurs valides
	Note Cette option s'applique uniquement aux versions Java antérieures à Java 8 et n'est pas prise en charge sur les plateformes Elastic Beanstalk Tomcat basées sur Amazon Linux 2 et les versions ultérieures.		
Xms	Tailles initiales du segment de mémoire de la JVM.	`256m`	N/A
`optionName`	Spécifiez les options JVM arbitraires en plus de celles définies par la plate-forme Tomcat.	Non applicable	Non applicable

Espace de noms : **`aws:elasticbeanstalk:environment:proxy`**

Nom	Description	Par défaut	Valeurs valides

GzipCompression	Définissez la valeur sur `false` pour désactiver la compression des réponses.
	
	Valable uniquement sur les versions de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2).
		`true`	`true`
			
			`false`

ProxyServer	Définissez le proxy à utiliser sur les instances de votre environnement. Si vous définissez cette option sur `apache`, Elastic Beanstalk utilise Apache 2.4.
	
	Définissez cette option sur `apache/2.2` si votre application n'est pas prête à migrer à partir d'Apache 2.2 en raison des paramètre
		`nginx` (Amazon Linux 2)
		
		`apache` (AMI Amazon Linux)
			`apache`
			
			`apache/2.2` – AMI Amazon Linux uniquement
			
			`nginx`

Nom	Description	Par défaut	Valeurs valides
	s de configuration de proxy incompatibles. Cette valeur n'est valide que sur les versions de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2). Définissez cette option sur `nginx` pour utiliser nginx. Il s'agit de la version par défaut de la plateforme Amazon Linux 2. Pour de plus amples informations, veuillez consulter Configuration du serveur proxy.		

Historique de la plateforme .NET Core sous Linux

Espace de noms : **`aws:elasticbeanstalk:environment:proxy`**

Nom	Description	Par défaut	Valeurs valides

ProxyServer	Spécifie le serveur web à utiliser comme proxy.	`nginx`	`nginx`
			
			`none`

Options de la plateforme .NET

Espace de noms : **`aws:elasticbeanstalk:container:dotnet:apppool`**

Nom	Description	Par défaut	Valeurs valides

Target Runtime	Choisissez la version .NET Framework pour votre application.	`4.0`	`2.0`
			
			`4.0`

Enable 32-bit Applications	Réglez le paramètre sur `True` pour exécuter des applications 32 bits.	`False`	`True`

Nom	Description	Par défaut	Valeurs valides
			`False`

Options de plateforme Node.js

Espace de noms : **`aws:elasticbeanstalk:environment:proxy`**

Nom	Description	Par défaut	Valeurs valides

ProxyServer	Définissez le proxy à utiliser sur les instances de votre environnement.	`nginx`	`apache`
			
			`nginx`

Options de plateforme AMI Amazon Linux (pré-Amazon Linux 2)

Espace de noms : **`aws:elasticbeanstalk:container:nodejs`**

Nom	Description	Par défaut	Valeurs valides
NodeCommand	Commande utilisée pour démarrer l'application Node.js. Si une chaîne vide est spécifiée, `app.js` est utilisé, puis `server.js` , et enfin `npm start` dans cet ordre.	""	Non applicable
NodeVersion	Version de Node.js. Par exemple, `4.4.6` Les versions Node.js prises en charge varient selon les versions de plateforme Node.js. Pour obtenir la liste des versions prises en charge, accédez à Node.js dans le document Plateformes AWS Elastic Beanstalk .	varie	varie

Nom	Description	Par défaut	Valeurs valides
	Note Lorsque la prise en charge de la version de Node.js que vous utilisez est supprimée de la plateforme, vous devez modifier ou supprimer le paramètre de version avant de procéder à une mise à jour de la plateforme. Cela peut se produire lorsqu'une faille de sécurité est identifiée pour une ou plusieurs versions de Node.js Dans ce cas, la tentative de mise à jour vers une nouvelle version de la plate-forme qui ne prend pas en charge la configuration NodeVersionéchoue. Pour éviter d'avoir à créer un nouvel environnement, remplacez l'option de NodeVersionconfiguration par une version de Node.js prise en charge à la fois par l'ancienne version de plate-forme et par la nouvelle, ou supprimez le paramètre d'option, puis effectuez la mise à jour de la plate-forme.		
GzipCompression	Spécifie si la compression gzip est activée. Si cette ProxyServer valeur est définie surnone, la compression gzip est désactivée.	`false`	`true` `false`

Nom	Description	Par défaut	Valeurs valides

ProxyServer	Spécifie le serveur web qui doit être utilisé pour les connexions proxy à Node.js.
	Si cette ProxyServer valeur est définie surnone, les mappages de fichiers statiques ne prennent pas effet et la compression gzip est désactivée.
		`nginx`	`apache`
			`nginx`
			`none`

Espace de nom : **`aws:elasticbeanstalk:container:nodejs:staticfiles`**

Vous pouvez utiliser l'espace de noms suivant pour configurer le serveur proxy afin de servir des fichiers statiques. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application. Cela réduit le nombre de demandes que votre application doit traiter.

Mappez un chemin servi par le serveur proxy à un dossier dans le code source qui contient les ressources statiques. Chaque option que vous définissez dans cet espace de noms mappe un chemin d'accès différent.

> **Note**
>
> Les paramètres de fichiers statiques ne s'appliquent pas si `aws:elasticbeanstalk:container:nodejs::ProxyFiles` est défini sur `none`.

Nom	Valeur
Chemin d'accès où le serveur proxy va servir les fichiers. Exemple : `/images` pour traiter les fichiers à l'emplacement *`subdomain`* `.elasticbeanstalk.com/images` .	Nom du dossier contenant les fichiers. Exemple : `staticimages` pour traiter les fichiers depuis un dossier nommé `staticimages` au niveau supérieur de votre bundle de fichiers source.

Options de la plateforme PHP

Espace de noms : **`aws:elasticbeanstalk:container:php:phpini`**

Nom	Description	Par défaut	Valeurs valides
document_root	Spécifiez le répertoire enfant de votre projet qui est traité comme la racine web destinée au public.	`/`	Une chaîne vide est traitée comme `/`, ou spécifiez une chaîne commençant par `/`
memory_limit	Volume de mémoire alloué à l'environnement PHP.	`256M`	Non applicable

zlib.output_compression	Spécifie si PHP doit utiliser ou non la compression pour la sortie.	`Off`	`On`
			`Off`
			`true`
			`false`

allow_url_fopen	Spécifie si les fonctions du fichier PHP sont autorisées à extraire des données des sites distants, comme les sites web ou les serveurs FTP.	`On`	`On`
			`Off`
			`true`
			`false`

display_errors	Spécifie si les messages d'erreur doivent faire partie de la sortie.	`Off`	`On` `Off`
max_execution_time	Définit la durée maximale, en secondes, durant laquelle un script est autorisé à s'exécuter avant qu'il soit résilié par l'environnement.	`60`	`0` à `9223372036854775807` (PHP_INT_MAX)
composer_options	Définit les options personnalisées à utiliser lors de l'installation de dépendanc	N/A	Non applicable

Nom	Description	Par défaut	Valeurs valides
	es à l'aide de Composer via la composer.phar install commande. Pour plus d'informations, consultez la section installation sur le site Web getcomposer.org.		

Espace de noms : **`aws:elasticbeanstalk:environment:proxy`**

Nom	Description	Par défaut	Valeurs valides
ProxyServer	Définissez le proxy à utiliser sur les instances de votre environnement.	`nginx`	`apache` `nginx`

> **Note**
>
> Pour plus d'informations sur la plateforme PHP, consultez Utilisation de la plateforme PHP Elastic Beanstalk.

Options de la plateforme Python

Espace de noms : **`aws:elasticbeanstalk:application:environment`**

Nom	Description	Par défaut	Valeurs valides
DJANGO_SETTINGS_MODULE	Spécifie le fichier de paramètres à utiliser.	Non applicable	Non applicable

Pour plus d'informations, consultez Variables d'environnement et autres paramètres du logiciel.

Espace de noms : **`aws:elasticbeanstalk:container:python`**

Nom	Description	Par défaut	Valeurs valides
WSGIPath	Le fichier qui contient l'application WSGI. Ce fichier doit avoir une `application` joignable.	Sur les versions de plateforme Python Amazon Linux 2 : `application` Sur les versions de la plateforme AMI Python Amazon Linux : `application.py`	Non applicable
NumProcesses	Le nombre de processus de démon qui doit être démarré pour le groupe de processus lorsque vous exécutez des applications WSGI.	1	Non applicable
NumThreads	Le nombre de threads à créer pour gérer des demandes dans chaque processus de démon au sein du groupe de processus lorsque vous exécutez des applications WSGI.	15	Non applicable

Espace de noms : **`aws:elasticbeanstalk:environment:proxy`**

Nom	Description	Par défaut	Valeurs valides

ProxyServer	Définissez le proxy à utiliser sur les instances de votre environnement.	`nginx`	`apache`
			
			`nginx`

Options de plateforme AMI Amazon Linux (pré-Amazon Linux 2)

Espace de nom : **`aws:elasticbeanstalk:container:python:staticfiles`**

Vous pouvez utiliser l'espace de noms suivant pour configurer le serveur proxy afin de servir des fichiers statiques. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application. Cela réduit le nombre de demandes que votre application doit traiter.

Mappez un chemin servi par le serveur proxy à un dossier dans le code source qui contient les ressources statiques. Chaque option que vous définissez dans cet espace de noms mappe un chemin d'accès différent.

Par défaut, le serveur proxy d'un environnement Python sert tous les fichiers dans un dossier nommé `static` sur le chemin d'accès `/static`.

Espace de noms : **`aws:elasticbeanstalk:container:python:staticfiles`**

Nom	Valeur
Chemin d'accès où le serveur proxy va servir les fichiers. Exemple : `/images` pour traiter les fichiers à l'emplacement *`subdomain`* `.elasticbeanstalk.com/images` .	Nom du dossier contenant les fichiers. Exemple : `staticimages` pour traiter les fichiers depuis un dossier nommé `staticimages` au niveau supérieur de votre bundle de fichiers source.

Options de la plateforme Ruby

Espace de noms : **`aws:elasticbeanstalk:application:environment`**

Nom	Description	Par défaut	Valeurs valides
RAILS_SKIP_MIGRATIONS	Spécifie s'il convient d'exécuter `` `rake db:migrate` `` au nom d'applications utilisateurs ; ou si cette valeur doit être ignorée. Ceci n'est applicable qu'aux applications Rails 3.	`false`	`true` `false`
RAILS_SKIP_ASSET_COMPILATION	Spécifie si le conteneur doit exécuter `` `rake assets:precompile` `` au nom des applications utilisateurs ; ou si cette valeur doit être ignorée. Cela est également applicable uniquement aux applications Rails 3.	`false`	`true` `false`
BUNDLE_WITHOUT	Une liste séparée par deux points (`:`) de groupes à ignorer lors de l'installation de dépendances à partir d'un Gemfile.	`test:development`	Non applicable
RACK_ENV	Spécifie dans quelle étape de l'environnement une application peut être exécutée. Des exemples d'environnements courants incluent le développement, la production, le test.	`production`	Non applicable

Pour plus d'informations, consultez Variables d'environnement et autres paramètres du logiciel.

Options personnalisées

Utilisez l'espace de noms `aws:elasticbeanstalk:customoption` pour définir des options et des valeurs qui peuvent être lues dans les blocs `Resources` des autres fichiers de configuration. Utilisez des options personnalisées pour collecter les paramètres spécifiés par les utilisateurs dans un fichier de configuration unique.

Par exemple, vous disposez peut-être d'un fichier de configuration complexe qui définit une ressource pouvant être configurée par l'utilisateur qui lance l'environnement. Si vous utilisez `Fn::GetOptionSetting` pour récupérer la valeur d'une option personnalisée, vous pouvez placer la définition de cette option dans un autre fichier de configuration, où l'utilisateur pourra plus facilement la trouver et la modifier.

En outre, comme il s'agit d'options de configuration, les options personnalisées peuvent être définies au niveau de l'API pour remplacer les valeurs définies dans un fichier de configuration. Pour plus d'informations, consultez la section Priorité.

Les options personnalisées sont définies comme toute autre option :

```
option_settings:
  aws:elasticbeanstalk:customoption:
    option name: option value
```

Par exemple, le fichier de configuration suivant crée une option nommée `ELBAlarmEmail` et lui attribue la valeur `someone@example.com` :

```
option_settings:
  aws:elasticbeanstalk:customoption:
    ELBAlarmEmail: someone@example.com
```

Autre exemple : un fichier de configuration peut définir une rubrique SNS qui lit l'option avec `Fn::GetOptionSetting` pour renseigner la valeur de l'attribut `Endpoint` :

```
Resources:
  MySNSTopic:
    Type: AWS::SNS::Topic
    Properties:
      Subscription:
        - Endpoint:
            Fn::GetOptionSetting:
              OptionName: ELBAlarmEmail
              DefaultValue: nobody@example.com
          Protocol: email
```

Vous trouverez d'autres exemples d'extraits utilisant la fonction `Fn::GetOptionSetting` sur la page Ajout et personnalisation des ressources de l'environnement Elastic Beanstalk.

Personnalisation d'environnement avancée avec fichiers de configuration (`.ebextensions`)

Vous pouvez ajouter des fichiers de AWS Elastic Beanstalk configuration (`.ebextensions`) au code source de votre application Web pour configurer votre environnement et personnaliser les AWS ressources qu'il contient. Les fichiers de configuration sont des documents au format YAML ou JSON dotés d'une extension de `.config` fichier que vous placez dans un dossier nommé `.ebextensions` et que vous déployez dans le bundle de sources de votre application.

Example extensions .eb/ .config network-load-balancer

Dans cet exemple on effectue une modification de configuration simple. Une option de configuration est modifiée de sorte à définir le type de l'équilibreur de charge de votre environnement sur Network Load Balancer.

```
option_settings:
  aws:elasticbeanstalk:environment:
    LoadBalancerType: network
```

Nous vous recommandons d'utiliser un format YAML pour vos fichiers de configuration, car il est plus lisible que JSON. YAML prend en charge des commentaires, des commandes sur plusieurs lignes, plusieurs alternatives pour l'utilisation de guillemets, et bien plus encore. Toutefois, vous pouvez effectuer une modification de configuration dans les fichiers de configuration Elastic Beanstalk comme avec YAML ou JSON.

> **Conseil**
>
> Lorsque vous développez ou testez de nouveaux fichiers de configuration, lancez un environnement propre qui exécute l'application par défaut et déployez les fichiers dans cet environnement. Des fichiers de configuration au format erroné empêcheront irrémédiablement le démarrage d'un nouvel environnement.

La section `option_settings` d'un fichier de configuration définit des valeurs pour des options de configuration. Les options de configuration vous permettent de configurer votre environnement Elastic Beanstalk AWS , les ressources qu'il contient et le logiciel qui exécute votre application. Les fichiers de configuration sont seulement une parmi de multiples façons de définir les options de configuration.

Cette `Resources`section vous permet de personnaliser davantage les ressources de l'environnement de votre application et de définir des AWS ressources supplémentaires au-delà des fonctionnalités fournies par les options de configuration. Vous pouvez ajouter et configurer toutes les ressources prises en charge par AWS CloudFormation Elastic Beanstalk pour créer des environnements.

Les autres sections d'un fichier de configuration (`packages`,`sources`,`files`,`users`,`groups`,`commands`,`container_commands`, et`services`) vous permettent de configurer les EC2 instances lancées dans votre environnement. Chaque fois qu'un serveur est démarré dans votre environnement, Elastic Beanstalk exécute les opérations définies dans ces sections pour préparer le système d'exploitation et le système de stockage pour votre application.

Pour obtenir des exemples de fichiers .ebextensions couramment utilisés, consultez la rubrique Référentiel de fichiers de configuration Elastic Beanstalk.

Exigences

- Emplacement — Elastic Beanstalk `.ebextensions` traitera tous les dossiers présents dans votre déploiement. Toutefois, nous vous recommandons de placer tous vos fichiers de configuration dans un seul dossier, nommé`.ebextensions`, à la racine de votre bundle source. Comme les dossiers commençant par un point peuvent être masquées par des navigateurs de fichiers, assurez-vous que le dossier est ajouté lorsque vous créez votre bundle de fichiers source. Pour de plus amples informations, veuillez consulter Création d'un bundle de sources d'applications Elastic Beanstalk.
- Attribution d'un nom – Les fichiers de configuration doivent porter l'extension de fichier `.config`.
- Mise en forme – Les fichiers de configuration doivent être conformes aux spécifications YAML ou JSON.

  Lorsque vous utilisez YAML, utilisez des espaces pour mettre retrait les clés à différents niveaux d'imbrication. Pour plus d'informations sur YAML, consultez YAML Ain't Markup Language (YAML™) Version 1.1.
- Unicité – Utilisez chaque clé une seule fois dans chaque fichier de configuration.

> **Avertissement**
>
> Si vous utilisez une clé (par exemple, `option_settings`) deux fois dans le même fichier de configuration, l'une des sections sera supprimée. Combinez les sections dupliquées en une seule section, ou placez-les dans des fichiers de configuration distincts.

Le processus de déploiement varie légèrement selon le client que vous utilisez pour gérer vos environnements. Consultez les sections suivantes pour obtenir des détails :

- Console Elastic Beanstalk
- INTERFACE DE LIGNE DE COMMANDE EB
- AWS CLI

Rubriques

- Paramètres d'option
- Personnalisation du logiciel sur des serveurs Linux
- Personnalisation du logiciel sur des serveurs Windows
- Ajout et personnalisation des ressources de l'environnement Elastic Beanstalk

Paramètres d'option

Vous pouvez utiliser la clé `option_settings` pour modifier la configuration d'Elastic Beanstalk et définir des variables qui peuvent être récupérées depuis votre application à l'aide des variables d'environnement. Certains espaces de noms vous permettent d'étendre le nombre de paramètres et spécifient les noms de paramètres. Pour une liste des options de configuration et des espaces de noms, consultez Options de configuration.

Les paramètres d'option peuvent également être appliqués directement dans un environnement au cours de la création de l'environnement ou d'une mise à jour de l'environnement. Les paramètres appliqués directement à l'environnement remplacent ceux pour les mêmes options dans les fichiers de configuration. Si vous supprimez des paramètres de la configuration d'un environnement, les paramètres dans les fichiers de configuration entrent en vigueur. Consultez Priorité pour plus de détails.

Syntaxe

La syntaxe standard pour les paramètres d'option est un ensemble d'objets, chacun ayant un `namespace`, `option_name` et une clé `value`.

```
option_settings:
  - namespace:  namespace
    option_name:  option name
    value:  option value
  - namespace:  namespace
    option_name:  option name
    value:  option value
```

La clé `namespace` est facultative. Si vous ne spécifiez pas un espace de noms, la valeur par défaut utilisée est `aws:elasticbeanstalk:application:environment` :

```
option_settings:
  - option_name:  option name
    value:  option value
  - option_name:  option name
    value:  option value
```

Elastic Beanstalk prend également en charge une syntaxe raccourcie pour les paramètres d'option qui vous permet de spécifier des options sous forme de paires clé-valeur sous l'espace de noms :

```
option_settings:
  namespace:
    option name: option value
    option name: option value
```

Exemples

Les exemples suivants définissent une option spécifique à la plateforme Tomcat dans l'espace de noms `aws:elasticbeanstalk:container:tomcat:jvmoptions` et une propriété d'environnement nommée `MYPARAMETER`.

Au format YAML standard :

Example.ebextensions/options.config

```
option_settings:
```

```
  - namespace:  aws:elasticbeanstalk:container:tomcat:jvmoptions
    option_name:  Xmx
    value:  256m
  - option_name: MYPARAMETER
    value: parametervalue
```

Au format raccourci :

Example.ebextensions/options.config

```
option_settings:
  aws:elasticbeanstalk:container:tomcat:jvmoptions:
    Xmx: 256m
  aws:elasticbeanstalk:application:environment:
    MYPARAMETER: parametervalue
```

Dans JSON :

Example.ebextensions/options.config

```
{
  "option_settings": [
    {
      "namespace": "aws:elasticbeanstalk:container:tomcat:jvmoptions",
      "option_name": "Xmx",
      "value": "256m"
    },
    {
      "option_name": "MYPARAMETER",
      "value": "parametervalue"
    }
  ]
}
```

Personnalisation du logiciel sur des serveurs Linux

Cette section décrit le type d'informations que vous pouvez inclure dans un fichier de configuration pour personnaliser le logiciel sur vos EC2 instances exécutant Linux. Pour obtenir des informations générales sur la personnalisation et la configuration de vos environnements Elastic Beanstalk, veuillez consulter Configuration d'environnements Elastic Beanstalk. Pour plus d'informations sur la personnalisation du logiciel sur vos EC2 instances exécutant Windows, consultezPersonnalisation du logiciel sur des serveurs Windows.

Vous pouvez souhaiter personnaliser et configurer le logiciel dont dépend votre application. Vous pouvez ajouter des commandes à exécuter pendant le provisionnement de l'instance, définir des utilisateurs et des groupes Linux et télécharger ou créer directement des fichiers sur vos instances d'environnement. Ces fichiers peuvent être des dépendances requises par l'application (par exemple, des packages supplémentaires provenant du référentiel yum) ou des fichiers de configuration en remplacement d'un fichier de configuration de proxy par exemple, pour remplacer des paramètres spécifiques qui sont définis par défaut par Elastic Beanstalk.

Remarques

- Sur les plateformes Amazon Linux 2, au lieu de fournir des fichiers et des commandes dans des fichiers de configuration .ebextensions, nous vous recommandons vivement d'utiliser Buildfile, Profile et les hooks de plateforme dès que possible, pour configurer et exécuter du code personnalisé sur vos instances d'environnement pendant le provisionnement d'instance. Pour plus d'informations sur ces mécanismes, consultez the section called “Extension des plateformes Linux”.
- YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Les fichiers de configuration prennent en charge les clés suivantes qui affectent le serveur Linux sur lequel votre application s'exécute.

Clés

- Forfaits
- Groupes
- Utilisateurs
- Sources
- Fichiers
- Commandes
- Services
- Commandes de conteneur
- Exemple : utilisation de CloudWatch métriques Amazon personnalisées

Les clés sont traitées dans l'ordre dans lequel elles sont répertoriées ici.

Observez les événements de votre environnement pendant le développement et les tests des fichiers de configuration. Elastic Beanstalk ignore un fichier de configuration qui contient des erreurs de validation, comme une clé non valide, et ne traite aucune autre clé dans le même fichier. Lorsque cela se produit, Elastic Beanstalk ajoute un événement d'avertissement dans le journal des événements.

Forfaits

Vous pouvez utiliser la clé `packages` pour télécharger et installer des applications et des composants prépackagés.

Syntaxe

```
packages:
  name of package manager:
    package name: version
    ...
  name of package manager:
    package name: version
    ...
  ...
```

Vous pouvez spécifier plusieurs packages sous chaque clé de gestionnaire de package.

Formats de packages pris en charge

Actuellement, Elastic Beanstalk prend en charge les gestionnaires de package suivants : yum, rubygems, python et rpm. Les packages sont traités dans l'ordre suivant : rpm, yum, puis rubygems et python. Il n'y a pas de classement entre rubygems et python. Au sein de chaque gestionnaire de packages, l'ordre des packages d'installation n'est pas garanti. Utilisez un gestionnaire de package pris en charge par votre système d'exploitation.

> **Note**
>
> Elastic Beanstalk prend en charge deux gestionnaires de package sous-jacents pour Python : pip et easy_install. Toutefois, dans la syntaxe du fichier de configuration, vous devez spécifier `python` comme nom du gestionnaire de package. Lorsque vous utilisez un fichier de configuration pour spécifier un gestionnaire de package Python, Elastic Beanstalk utilise Python 2.7. Si votre application repose sur une autre version de Python, vous pouvez

indiquer que les packages doivent être installés dans un fichier `requirements.txt`. Pour de plus amples informations, veuillez consulter Spécification des dépendances à l'aide d'un fichier d'exigences sur Elastic Beanstalk.

Spécification des versions

Au sein de chaque gestionnaire de package, chaque package est spécifié sous la forme d'un nom de package et d'une liste de versions. La version peut être une chaîne, une liste de versions, ou une chaîne ou une liste vide. Une chaîne ou une liste vide indique que vous souhaitez la dernière version. Pour le gestionnaire rpm, la version est spécifiée sous la forme d'un chemin d'accès à un fichier sur disque ou d'une URL. Les chemins relatifs ne sont pas pris en charge.

Si vous spécifiez une version d'un package, Elastic Beanstalk tente d'installer cette version, même si une version plus récente du package est déjà installée sur l'instance. Si une version plus récente est déjà installée, le déploiement échoue. Certains gestionnaires de package prennent en charge plusieurs versions, mais pas tous. Veuillez vérifier la documentation de votre gestionnaire de package pour plus d'informations. Si vous ne spécifiez aucune version et qu'une version du package est déjà installée, Elastic Beanstalk n'installe pas une nouvelle version, car il part du principe que vous souhaitez conserver et utiliser la version existante.

Exemple d'extrait

L'extrait suivant spécifie une URL de la version pour rpm, demande la dernière version à yum et la version 0.10.2 de chef à rubygems.

```
packages:
  yum:
    libmemcached: []
    ruby-devel: []
    gcc: []
  rpm:
    epel: http://download.fedoraproject.org/pub/epel/5/i386/epel-release-5-4.noarch.rpm
  rubygems:
    chef: '0.10.2'
```

Groupes

Vous pouvez utiliser la `groups` touche pour créer des Linux/UNIX groupes et pour attribuer des groupes IDs. Pour créer un groupe, ajoutez une nouvelle paire clé-valeur qui mappe un nouveau nom

de groupe à un ID de groupe facultatif. La clé des groupes peut contenir un ou plusieurs noms de groupe. Le tableau suivant répertorie les clés disponibles.

Syntaxe

```
groups:
  name of group: {}
  name of group:
    gid: "group id"
```

Options

`gid`

Numéro d'identification d'un groupe.

Si un ID de groupe est spécifié et que le nom du groupe existe déjà, la création du groupe échoue. Si un autre groupe est associé à l'ID de groupe spécifié, le système d'exploitation peut refuser la création du groupe.

Exemple d'extrait

L'extrait suivant spécifie un groupe nommé groupOne sans attribuer d'ID de groupe, et un groupe nommé groupTwo ayant spécifié 45 comme valeur d'ID de groupe.

```
groups:
  groupOne: {}
  groupTwo:
    gid: "45"
```

Utilisateurs

Vous pouvez utiliser la `users` clé pour créer des Linux/UNIX utilisateurs sur l' EC2 instance.

Syntaxe

```
users:
  name of user:
    groups:
```

```
      - name of group
    uid: "id of the user"
    homeDir: "user's home directory"
```

Options

`uid`

ID d'un utilisateur. Le processus de création échoue si le nom d'utilisateur existe avec un autre ID d'utilisateur. Si l'ID d'utilisateur est déjà affecté à un utilisateur existant, le système d'exploitation peut refuser la demande de création.

`groups`

Liste de noms de groupes. L'utilisateur est ajouté à chaque groupe de la liste.

`homeDir`

Répertoire de base de l'utilisateur.

Les utilisateurs sont créés en tant qu'utilisateurs du système non interactif avec le shell `/sbin/nologin`. Ce paramètre est intégré à la conception et ne peut pas être modifié.

Exemple d'extrait

```
users:
  myuser:
    groups:
      - group1
      - group2
    uid: "50"
    homeDir: "/tmp"
```

Sources

Vous pouvez utiliser la `sources` clé pour télécharger un fichier d'archive à partir d'une URL publique et le décompresser dans un répertoire cible de l' EC2 instance.

Syntaxe

```
sources:
```

```
  target directory: location of archive file
```

Formats pris en charge

Les formats pris en charge sont les suivants : tar, tar+gzip, tar+bz2 et zip. Vous pouvez faire référence à des emplacements externes tels qu'Amazon Simple Storage Service (Amazon S3) (par exemple, `https://amzn-s3-demo-bucket.s3.amazonaws.com/myobject`) tant que l'URL est publiquement accessible.

Exemple d'extrait

L'exemple suivant télécharge un fichier .zip public à partir d'un compartiment Amazon S3 et le décompresse dans `/etc/myapp`:

```
sources:
  /etc/myapp: https://amzn-s3-demo-bucket.s3.amazonaws.com/myobject
```

> **Note**
>
> Plusieurs extractions ne doivent pas réutiliser le même chemin cible. L'extraction d'une autre source vers le même chemin cible remplacera le contenu au lieu de l'ajouter.

Fichiers

Vous pouvez utiliser la `files` clé pour créer des fichiers sur l' EC2 instance. Le contenu peut être soit en ligne dans le fichier de configuration, ou bien le contenu peut être extrait d'une URL. Les fichiers sont écrits sur disque par ordre lexicographique.

Vous pouvez utiliser la clé `files` pour télécharger des fichiers privés depuis Amazon S3 en fournissant un profil d'instance pour l'autorisation.

Si le chemin d'accès au fichier que vous spécifiez existe déjà sur l'instance, le fichier existant est conservé avec l'extension `.bak` ajoutée à son nom.

Syntaxe

```
files:
```

```
  "target file location on disk":
     mode: "six-digit octal value"
     owner: name of owning user for file
     group: name of owning group for file
     source: URL
     authentication: authentication name:

  "target file location on disk":
     mode: "six-digit octal value"
     owner: name of owning user for file
     group: name of owning group for file
     content: |
      # this is my
      # file content
     encoding: encoding format
     authentication: authentication name:
```

Options

`content`

Contenu de la chaîne à ajouter dans le fichier. Spécifiez `content` ou `source`, mais pas les deux.

`source`

URL du fichier à télécharger. Spécifiez `content` ou `source`, mais pas les deux.

`encoding`

Format d'encodage de la chaîne spécifiée via l'option `content`.

Valeurs valides : `plain` | `base64`

`group`

Groupe Linux auquel appartient le fichier.

`owner`

Utilisateur Linux auquel appartient le fichier.

`mode`

Valeur octale à six chiffres qui représente le mode de ce fichier. Non pris en charge pour les systèmes Windows. Utilisez les trois premiers chiffres pour les liens symboliques et les trois

derniers chiffres pour définir les autorisations. Pour créer un lien symbolique, spécifiez `120xxx`, où `xxx` définit les autorisations du fichier cible. Pour définir des autorisations pour un fichier, utilisez les trois derniers chiffres, tels que `000644`.

`authentication`

Le nom d'une méthode d'authentification CloudFormation à utiliser. Vous pouvez ajouter des méthodes d'authentification aux métadonnées du groupe Auto Scaling via la clé Resources. Vous trouverez un exemple ci-dessous.

Exemple d'extrait

```
files:
  "/home/ec2-user/myfile" :
    mode: "000755"
    owner: root
    group: root
    source: http://foo.bar/myfile

  "/home/ec2-user/myfile2" :
    mode: "000755"
    owner: root
    group: root
    content: |
      this is my
      file content
```

Exemple d'utilisation d'un lien symbolique. Cela crée un lien `/tmp/myfile2.txt` qui renvoie vers le fichier existant `/tmp/myfile1.txt`.

```
files:
  "/tmp/myfile2.txt" :
    mode: "120400"
    content: "/tmp/myfile1.txt"
```

L'exemple suivant utilise la clé Resources pour ajouter une méthode d'authentification nommée S3Auth et l'utilise pour télécharger un fichier privé à partir d'un compartiment Amazon S3 :

```
Resources:
  AWSEBAutoScalingGroup:
```

```
    Metadata:
      AWS::CloudFormation::Authentication:
        S3Auth:
          type: "s3"
          buckets: ["amzn-s3-demo-bucket2"]
          roleName:
            "Fn::GetOptionSetting":
              Namespace: "aws:autoscaling:launchconfiguration"
              OptionName: "IamInstanceProfile"
              DefaultValue: "aws-elasticbeanstalk-ec2-role"

files:
  "/tmp/data.json" :
    mode: "000755"
    owner: root
    group: root
    authentication: "S3Auth"
    source: https://elasticbeanstalk-us-west-2-123456789012.s3-us-west-2.amazonaws.com/
data.json
```

Commandes

Vous pouvez utiliser la `commands` touche pour exécuter des commandes sur l' EC2 instance. Les commandes exécutées avant l'application et le serveur web sont définies et le fichier de version de l'application est extrait.

Les commandes spécifiées s'exécutent en tant qu'utilisateur racine et sont traitées dans l'ordre alphabétique par nom. Commandes exécutées dans le répertoire racine (par défaut). Pour exécuter des commandes à partir d'un autre répertoire, utilisez l'option `cwd`.

Pour résoudre les problèmes associés à vos commandes, vous pouvez trouver leur sortie dans les journaux d'instance.

Syntaxe

```
commands:
  command name:
    command: command to run
    cwd: working directory
    env:
      variable name: variable value
    test: conditions for command
```

```
    ignoreErrors: true
```

Options

`command`

Tableau (collection de séquence de blocs en syntaxe YAML) ou chaîne spécifiant la commande à exécuter. Remarques importantes :

- Si vous utilisez une chaîne, vous n'avez pas besoin de placer la totalité de la chaîne entre guillemets. Si vous utilisez des guillemets, utilisez des caractères d'échappement pour les occurrences littérales du même type de guillemets.
- Si vous utilisez un tableau, vous n'avez pas besoin de caractères d'espacement ou d'inclure des paramètres de commande entre guillemets. Chaque élément de tableau est un argument de commande unique. N'utilisez pas un tableau pour spécifier plusieurs commandes.

Les exemples suivants sont tous équivalents :

```
commands:
  command1:
    command: git commit -m "This is a comment."
  command2:
    command: "git commit -m \"This is a comment.\""
  command3:
    command: 'git commit -m "This is a comment."'
  command4:
    command:
      - git
      - commit
      - -m
      - This is a comment.
```

Pour spécifier plusieurs commandes, utilisez un bloc littéral scalaire, comme illustré dans l'exemple suivant.

```
commands:
  command block:
    command: |
      git commit -m "This is a comment."
      git push
```

`env`

(Facultatif) Définit des variables d'environnement pour la commande. Cette propriété remplace, plutôt que d'ajouter, l'environnement existant.

`cwd`

(Facultatif) Le répertoire de travail. Si cette option n'est pas spécifiée, les commandes sont exécutées à partir du répertoire racine (/).

`test`

(Facultatif) Commande qui doit renvoyer la valeur `true` (code de sortie 0) afin qu'Elastic Beanstalk traite la commande (par exemple, un script shell) contenue dans la clé `command`.

`ignoreErrors`

(Facultatif) Valeur booléenne qui détermine si les autres commandes doivent s'exécuter si la commande contenue dans la clé `command` échoue (renvoie une valeur non nulle). Définissez cette valeur sur `true` si vous voulez continuer à exécuter des commandes même si la commande échoue. Définissez-la sur `false` si vous souhaitez arrêter l'exécution des commandes si la commande échoue. La valeur par défaut est `false`.

Exemple d'extrait

L'exemple d'extrait suivant exécute un script Python.

```
commands:
  python_install:
    command: myscript.py
    cwd: /home/ec2-user
    env:
      myvarname: myvarvalue
    test: "[ -x /usr/bin/python ]"
```

Services

Vous pouvez utiliser la clé `services` pour définir les services qui doivent être démarrés ou arrêtés lors du lancement de l'instance. La clé `services` vous permet également de spécifier des dépendances sur des sources, des packages et des fichiers de sorte que si un redémarrage est nécessaire en raison d'une installation de fichiers, Elastic Beanstalk prenne en charge le redémarrage du service.

Syntaxe

```
services:
  sysvinit:
    name of service:
      enabled: "true"
      ensureRunning: "true"
      files:
        - "file name"
      sources:
        - "directory"
      packages:
        name of package manager:
          "package name[: version]"
      commands:
        - "name of command"
```

Options

`ensureRunning`

Définissez cette option sur `true` afin de garantir que le service s'exécute après qu'Elastic Beanstalk a terminé.

Définissez cette option sur `false` afin de garantir que le service ne s'exécute pas après qu'Elastic Beanstalk a terminé.

Ignorez cette clé pour n'apporter aucune modification à l'état du service.

`enabled`

Définissez cette option sur `true` afin de garantir que le service soit démarré automatiquement lors du démarrage.

Définissez cette option sur `false` afin de garantir que le service ne soit pas démarré automatiquement lors du démarrage.

Ignorez cette clé pour n'apporter aucune modification à cette propriété.

`files`

Liste de fichiers. Si Elastic Beanstalk en change un directement via le bloc de fichiers, le service est redémarré.

`sources`

Liste de répertoires. Si Elastic Beanstalk développe une archive dans l'un de ces répertoires, le service est redémarré.

`packages`

Une carte du gestionnaire de package sur une liste de noms de package. Si Elastic Beanstalk installe ou met à jour l'un de ces packages, le service est redémarré.

`commands`

Liste de noms de commandes. Si Elastic Beanstalk exécute la commande spécifiée, le service est redémarré.

Exemple d'extrait

Voici un exemple d'extrait :

```
services:
  sysvinit:
    myservice:
      enabled: true
      ensureRunning: true
```

Commandes de conteneur

Vous pouvez utiliser la clé `container_commands` pour exécuter des commandes qui affectent le code source de votre application. Les commandes de conteneur s'exécutent après que l'application et le serveur web ont été configurés et que l'archive de version d'application a été extraite, mais avant que la version d'application soit déployée. Les commandes non-conteneur et autres opérations de personnalisation sont effectuées avant le code source d'application en cours d'extraction.

Les commandes spécifiées s'exécutent en tant qu'utilisateur racine et sont traitées dans l'ordre alphabétique par nom. Les commandes de conteneur sont exécutées depuis le répertoire intermédiaire, où votre code source est extrait avant d'être déployé sur le serveur d'applications. Toutes les modifications apportées à votre code source dans le répertoire intermédiaire avec une commande conteneur seront incluses lorsque la source est déployée sur son emplacement final.

> **Note**
>
> La sortie de vos commandes de conteneur est enregistrée dans le journal d'instance `cfn-init-cmd.log`. Pour plus d'informations sur la récupération et l'affichage des journaux d'instance, consultez Afficher les journaux des EC2 instances Amazon.

Vous pouvez utiliser `leader_only` pour exécuter uniquement la commande sur une seule instance, ou configurer un `test` pour exécuter uniquement la commande lorsqu'une commande test évalue sur `true`. Les commandes de conteneur principales uniquement sont exécutées uniquement au cours de la création et du déploiement de l'environnement, tandis que d'autres commandes et opérations de personnalisation de serveur sont effectuées chaque fois qu'une instance est mise en service ou mise à jour. Les commandes de conteneur principales uniquement ne sont pas exécutées pour lancer des modifications de configuration, par exemple, une modification d'ID d'AMI ou de type d'instance.

Syntaxe

```
container_commands:
  name of container_command:
    command: "command to run"
    leader_only: true
  name of container_command:
    command: "command to run"
```

Options

`command`

Une chaîne ou un tableau de chaînes à exécuter.

`env`

(Facultatif) Définissez les variables d'environnement avant d'exécuter la commande, en ignorant toute valeur existante.

`cwd`

(Facultatif) Le répertoire de travail. Par défaut, il s'agit du répertoire intermédiaire de l'application décompressée.

`leader_only`

(Facultatif) Exécutez uniquement la commande sur une seule instance choisie par Elastic Beanstalk. Les commandes de conteneur principales uniquement sont exécutées avant d'autres commandes de conteneur. Une commande peut être principale uniquement ou disposer d'un `test`, mais pas les deux (`leader_only` a la priorité).

`test`

(Facultatif) Exécutez une commande test qui doit renvoyer la valeur `true` afin d'exécuter la commande de conteneur. Une commande peut être principale uniquement ou disposer d'un `test`, mais pas les deux (`leader_only` a la priorité).

`ignoreErrors`

(Facultatif) Ne manquez pas les déploiements si la commande de conteneur renvoie une valeur différente de 0 (réussite). Définissez sur `true` pour activer.

Exemple d'extrait

Voici un exemple d'extrait :

```
container_commands:
  collectstatic:
    command: "django-admin.py collectstatic --noinput"
  01syncdb:
    command: "django-admin.py syncdb --noinput"
    leader_only: true
  02migrate:
    command: "django-admin.py migrate"
    leader_only: true
  99customize:
    command: "scripts/customize.sh"
```

Exemple : utilisation de CloudWatch métriques Amazon personnalisées

Cette rubrique fournit un exemple de configuration qui intègre les métriques d'Elastic Beanstalk à l'agent Amazon pour CloudWatch les plateformes basées sur Amazon Linux 2 et versions ultérieures. L'exemple de configuration utilise des fichiers et des commandes dans un fichier `.ebextensions` de configuration.

Amazon CloudWatch est un service Web qui vous permet de surveiller, de gérer et de publier diverses métriques, ainsi que de configurer des actions d'alarme en fonction des données issues des métriques. Vous pouvez définir des métriques personnalisées pour votre propre usage, et Elastic Beanstalk transmettra ces métriques à Amazon. CloudWatch Une fois qu'Amazon CloudWatch contient vos statistiques personnalisées, vous pouvez les consulter dans la CloudWatch console Amazon.

L' CloudWatch agent Amazon

L' CloudWatch agent Amazon permet la collecte de CloudWatch métriques et de journaux à la fois depuis les EC2 instances Amazon et les serveurs sur site sur tous les systèmes d'exploitation. L'agent prend en charge les métriques collectées au niveau du système. Il prend également en charge la collecte personnalisée de journaux et de métriques à partir de vos applications ou services. Pour plus d'informations sur l' CloudWatch agent Amazon, consultez la section Collecte de métriques et de journaux avec l' CloudWatch agent dans le guide de CloudWatch l'utilisateur Amazon.

> **Note**
>
> Elastic Beanstalk Enhanced Health Reporting prend en charge de manière native la publication d'un large éventail de métriques relatives aux instances et à l'environnement sur. CloudWatch Consultez Publication de métriques CloudWatch personnalisées Amazon pour un environnement pour plus de détails.

Rubriques

- Fichier de configuration .Ebextensions
- Permissions
- Afficher les métriques dans la CloudWatch console

Fichier de configuration .Ebextensions

Cet exemple utilise les fichiers et les commandes d'un fichier de configuration .ebextensions pour configurer et exécuter l' CloudWatch agent Amazon sur la plateforme Amazon Linux 2. L'agent est préemballé avec Amazon Linux 2. Si vous utilisez un autre système d'exploitation, des étapes supplémentaires pour installer l'agent peuvent être nécessaires. Pour plus d'informations, consultez la section Installation de l' CloudWatch agent dans le guide de CloudWatch l'utilisateur Amazon.

Pour utiliser cet exemple, enregistrez-le dans un fichier nommé `cloudwatch.config`, dans un répertoire nommé `.ebextensions` au niveau supérieur de votre répertoire de projet. Déployez ensuite votre application via la console Elastic Beanstalk (en incluant le répertoire .ebextensions dans votre bundle de fichiers source) ou via l'interface de ligne de commande EB.

Pour plus d'informations sur les fichiers de configuration, consultez Personnalisation d'environnement avancée avec fichiers de configuration (`.ebextensions`).

Ce fichier contient deux sections :

- `files` — Cette section ajoute le fichier de configuration d'agent. Il indique les métriques et les journaux que l'agent doit envoyer à Amazon CloudWatch. Dans cet exemple, nous envoyons uniquement la métrique mem_used_percent. Pour une liste complète des métriques au niveau du système prises en charge par l' CloudWatch agent Amazon, consultez la section Mesures collectées par l' CloudWatch agent dans le guide de CloudWatch l'utilisateur Amazon.
- `container_commands` — Cette section contient la commande qui démarre l'agent, en transmettant le fichier de configuration en tant que paramètre. Pour en savoir plus sur `container_commands`, consultez Commandes de conteneur.

.ebextensions/cloudwatch.config

```
files:
  "/opt/aws/amazon-cloudwatch-agent/bin/config.json":
    mode: "000600"
    owner: root
    group: root
    content: |
      {
        "agent": {
          "metrics_collection_interval": 60,
          "run_as_user": "root"
        },
        "metrics": {
          "namespace": "System/Linux",
          "append_dimensions": {
            "AutoScalingGroupName": "${aws:AutoScalingGroupName}"
          },
          "metrics_collected": {
            "mem": {
              "measurement": [
                "mem_used_percent"
```

```
              ]
            }
          }
        }
      }
container_commands:
  start_cloudwatch_agent:
    command: /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a
 append-config -m ec2 -s -c file:/opt/aws/amazon-cloudwatch-agent/bin/config.json
```

Permissions

Les instances de votre environnement ont besoin des autorisations IAM appropriées pour publier des CloudWatch métriques Amazon personnalisées à l'aide de l' CloudWatch agent Amazon. Vous accordez des autorisations aux instances de votre environnement en les ajoutant au profil d'instance de l'environnement. Vous pouvez ajouter des autorisations au profil d'instance avant ou après le déploiement de votre application.

Pour accorder l'autorisation de publier CloudWatch des métriques

1. Ouvrez la console IAM à l’adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Rôles.
3. Sélectionnez le rôle de profil d'instance de votre environnement. Par défaut, lorsque vous créez un environnement via la console Elastic Beanstalk ou l'interface de ligne de commande EB, il s'agit de `aws-elasticbeanstalk-ec2-role`.
4. Sélectionnez l’onglet Autorisations.
5. Sous Politiques d'autorisations, dans la section Autorisations, choisissez Attacher des politiques.
6. Sous Autorisations d'attachement, choisissez la politique AWS gérée CloudWatchAgentServerPolicy. Cliquez ensuite sur Attacher une politique.

Pour plus d'informations sur la gestion des politiques, consultez Utilisation des politiques dans le Guide de l'utilisateur IAM.

Afficher les métriques dans la CloudWatch console

Après avoir déployé le fichier de CloudWatch configuration dans votre environnement, consultez la CloudWatchconsole Amazon pour consulter vos statistiques. Les métriques personnalisées seront situées dans l'espace de CWAgentnoms.

Pour plus d'informations, consultez la section Affichage des statistiques disponibles dans le guide de CloudWatch l'utilisateur Amazon.

Personnalisation du logiciel sur des serveurs Windows

Vous pouvez souhaiter personnaliser et configurer le logiciel dont dépend votre application. Ces fichiers pourraient être soit des dépendances requises par l'application, par exemple, des services ou des packages supplémentaires qui doivent être exécutés. Pour obtenir des informations générales sur la personnalisation et la configuration de vos environnements Elastic Beanstalk, consultez Configuration d'environnements Elastic Beanstalk.

> **Note**
>
> YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Les fichiers de configuration prennent en charge les clés suivantes qui affectent le serveur Windows sur lequel votre application s'exécute.

Clés

- Forfaits
- Sources
- Fichiers
- Commandes
- Services
- Commandes de conteneur

Les clés sont traitées dans l'ordre dans lequel elles sont répertoriées ici.

> **Note**
>
> Des versions antérieures (sans version) des versions de plateforme .NET ne traitent pas les fichiers de configuration dans l'ordre correct. Découvrez-en plus sur Migration entre les principales versions de la plateforme Windows Server pour Elastic Beanstalk.

Observez les événements de votre environnement pendant le développement et les tests des fichiers de configuration. Elastic Beanstalk ignore un fichier de configuration qui contient des erreurs de validation, comme une clé non valide, et ne traite aucune autre clé dans le même fichier. Lorsque cela se produit, Elastic Beanstalk ajoute un événement d'avertissement dans le journal des événements.

Forfaits

Utilisez la clé `packages` pour télécharger, et installer des applications et des composants prépackagés.

Dans les environnements Windows, Elastic Beanstalk prend en charge le téléchargement et l'installation des packages MSI. (Les environnements Linux prennent en charge d'autres gestionnaires de packages. Pour plus d'informations, consultez Forfaitssur la page Personnalisation des logiciels sur les serveurs Linux.)

Vous pouvez faire référence à des emplacements externes, comme un objet Amazon Simple Storage Service (Amazon S3), tant que l'URL est publiquement accessible.

Si vous spécifiez plusieurs packages `msi:`, leur ordre d'installation n'est pas garanti.

Syntaxe

Spécifiez le nom de votre choix comme nom du package et une URL vers un emplacement de fichier MSI comme valeur. Vous pouvez spécifier plusieurs packages sous la clé `msi:`.

```
packages:
  msi:
    package name: package url
    ...
```

Exemples

L'exemple suivant indique une URL permettant de télécharger mysql à partir `https://dev.mysql.com/`.

```
packages:
  msi:
    mysql: https://dev.mysql.com/get/Downloads/Connector-Net/mysql-connector-net-8.0.11.msi
```

L'exemple suivant indique un objet Amazon S3 comme emplacement de fichier MSI.

```
packages:
  msi:
    mymsi: https://amzn-s3-demo-bucket.s3.amazonaws.com/myobject.msi
```

Sources

Utilisez la clé `sources` pour télécharger un fichier d'archives à partir d'une URL publique et le décompresser dans un répertoire cible de l'instance EC2.

Syntaxe

```
sources:
  target directory: location of archive file
```

Formats pris en charge

Dans les environnements Windows, Elastic Beanstalk prend en charge le format .zip. (Les environnements Linux prennent en charge d'autres formats. Pour plus d'informations, consultez Sources sur la page Personnalisation des logiciels sur les serveurs Linux.)

Vous pouvez faire référence à des emplacements externes, comme un objet Amazon Simple Storage Service (Amazon S3), tant que l'URL est publiquement accessible.

Exemple

L'exemple suivant télécharge un fichier .zip public depuis un Amazon S3 et le décompresse dans. `c:/myproject/myapp`

```
sources:
```

```
  "c:/myproject/myapp": https://amzn-s3-demo-bucket.s3.amazonaws.com/myobject.zip
```

Fichiers

Utilisez la clé `files` pour créer des fichiers sur l'instance EC2. Le contenu peut être en ligne dans le fichier de configuration, ou issu d'une URL. Les fichiers sont écrits sur disque par ordre lexicographique. Pour télécharger des fichiers privés depuis Amazon S3, fournissez un profil d'instance pour l'autorisation.

Syntaxe

```
files:
  "target file location on disk":
    source: URL
    authentication: authentication name:

  "target file location on disk":
    content: |
      this is my content
    encoding: encoding format
```

Options

`content`

(Facultatif) Une chaîne.

`source`

(Facultatif) L'URL à partir de laquelle le fichier est chargé. Cette option ne peut pas être spécifiée avec la clé de contenu.

`encoding`

(Facultatif) Le format d'encodage. Cette option est utilisée uniquement pour une valeur de clé de contenu fournie. La valeur par défaut est `plain`.

Valeurs valides : `plain` | `base64`

`authentication`

(Facultatif) Le nom d'une méthode d'authentification CloudFormation à utiliser. Vous pouvez ajouter des méthodes d'authentification aux métadonnées du groupe Auto Scaling via la clé Resources.

Exemples

L'exemple suivant illustre les deux manières de fournir un contenu d'un fichier : depuis une URL, ou en ligne dans le fichier de configuration.

```
files:
  "c:\\targetdirectory\\targetfile.txt":
    source: http://foo.bar/myfile

  "c:/targetdirectory/targetfile.txt":
    content: |
      # this is my file
      # with content
```

> **Note**
>
> Si vous utilisez une barre oblique inverse (\) dans votre chemin d'accès, vous devez la faire précéder d'une autre barre oblique inverse (caractère d'échappement), comme illustré dans l'exemple précédent.

L'exemple suivant utilise la clé Resources pour ajouter une méthode d'authentification nommée S3Auth et l'utilise pour télécharger un fichier privé depuis un Amazon S3 :

```
files:
  "c:\\targetdirectory\\targetfile.zip":
    source: https://elasticbeanstalk-us-east-2-123456789012.s3.amazonaws.com/prefix/
myfile.zip
    authentication: S3Auth

Resources:
  AWSEBAutoScalingGroup:
    Metadata:
      AWS::CloudFormation::Authentication:
        S3Auth:
          type: "s3"
          s: ["amzn-s3-demo-bucket"]
          roleName:
            "Fn::GetOptionSetting":
              Namespace: "aws:autoscaling:launchconfiguration"
              OptionName: "IamInstanceProfile"
```

```
                DefaultValue: "aws-elasticbeanstalk-ec2-role"
```

Commandes

Utilisez la clé `commands` pour exécuter des commandes sur l'instance EC2. Les commandes sont traitées dans l'ordre alphabétique par nom et elles s'exécutent avant la configuration de l'application et du serveur web, avant que le fichier de version d'application soit extrait.

Les commandes spécifiées s'exécutent en tant qu'utilisateur Administrateur.

Pour résoudre les problèmes associés à vos commandes, vous pouvez trouver leur sortie dans les journaux d'instance.

Syntaxe

```
commands:
  command name:
    command: command to run
```

Options

`command`

Soit un tableau, soit une chaîne spécifiant la commande à exécuter. Si vous utilisez un tableau, vous n'avez pas besoin de caractères d'espacement ou d'inclure des paramètres de commande entre guillemets.

`cwd`

(Facultatif) Le répertoire de travail. Par défaut, Elastic Beanstalk tente de trouver l'emplacement du répertoire de votre projet. S'il ne le trouve pas, il utilise `c:\Windows\System32` comme valeur par défaut.

`env`

(Facultatif) Définit des variables d'environnement pour la commande. Cette propriété remplace, plutôt que d'ajouter, l'environnement existant.

`ignoreErrors`

(Facultatif) Valeur booléenne qui détermine si les autres commandes doivent s'exécuter si la commande contenue dans la clé `command` échoue (renvoie une valeur non nulle). Définissez

cette valeur sur `true` si vous voulez continuer à exécuter des commandes même si la commande échoue. Définissez-la sur `false` si vous souhaitez arrêter l'exécution des commandes si la commande échoue. La valeur par défaut est `false`.

`test`

(Facultatif) Commande qui doit renvoyer la valeur `true` (code de sortie 0) afin qu'Elastic Beanstalk traite la commande contenue dans la clé `command`.

`waitAfterCompletion`

(Facultatif) Secondes d'attente une fois que la commande est terminée avant d'exécuter la commande suivante. La valeur par défaut est de **`60`** secondes. Vous pouvez également spécifier**`forever`**.

> **Important**
>
> Les redémarrages du système pendant le déploiement ne sont pas pris en charge. Si le système redémarre à la suite d'une commande, l'initialisation de l'instance échouera, entraînant l'échec du déploiement.
> Pour contourner le problème, vous pouvez l'utiliser `.ebextensions` `configuration`pour planifier des redémarrages une fois le déploiement terminé.

Exemple

L'exemple suivant enregistre la sortie de la commande `set` dans le fichier spécifié. S'il existe une commande suivante, Elastic Beanstalk exécute cette commande immédiatement à la fin de cette commande. Si cette commande exige un redémarrage, Elastic Beanstalk redémarre l'instance immédiatement à la fin de la commande.

```
commands:
  test:
    command: set > c:\\myapp\\set.txt
    waitAfterCompletion: 0
```

Services

Utilisez la clé `services` pour définir les services qui doivent être démarrés ou arrêtés lors du lancement de l'instance. La clé `services` vous permet également de spécifier des dépendances sur

des sources, des packages et des fichiers de sorte que si un redémarrage est nécessaire en raison d'une installation de fichiers, Elastic Beanstalk prenne en charge le redémarrage du service.

Syntaxe

```
services:
  windows:
    name of service:
      files:
        - "file name"
      sources:
        - "directory"
      packages:
        name of package manager:
            "package name[: version]"
      commands:
        - "name of command"
```

Options

`ensureRunning`

(Facultatif) Définissez cette option sur `true` afin de garantir que le service s'exécute après qu'Elastic Beanstalk a terminé.

Définissez cette option sur `false` afin de garantir que le service ne s'exécute pas après qu'Elastic Beanstalk a terminé.

Ignorez cette clé pour n'apporter aucune modification à l'état du service.

`enabled`

(Facultatif) Définissez cette option sur `true` afin de garantir que le service soit démarré automatiquement lors du démarrage.

Définissez cette option sur `false` afin de garantir que le service ne soit pas démarré automatiquement lors du démarrage.

Ignorez cette clé pour n'apporter aucune modification à cette propriété.

`files`

Liste de fichiers. Si Elastic Beanstalk en change un directement via le bloc de fichiers, le service est redémarré.

`sources`

Liste de répertoires. Si Elastic Beanstalk développe une archive dans l'un de ces répertoires, le service est redémarré.

`packages`

Une carte du gestionnaire de package sur une liste de noms de package. Si Elastic Beanstalk installe ou met à jour l'un de ces packages, le service est redémarré.

`commands`

Liste de noms de commandes. Si Elastic Beanstalk exécute la commande spécifiée, le service est redémarré.

Exemple

```
services:
  windows:
    myservice:
      enabled: true
      ensureRunning: true
```

Commandes de conteneur

Utilisez la clé `container_commands` pour exécuter des commandes qui affectent le code source de votre application. Les commandes de conteneur s'exécutent après que l'application et le serveur web ont été configurés et que l'archive de version d'application a été extraite, mais avant que la version d'application soit déployée. Les commandes non-conteneur et autres opérations de personnalisation sont effectuées avant le code source d'application en cours d'extraction.

Les commandes de conteneur sont exécutées depuis le répertoire intermédiaire, où votre code source est extrait avant d'être déployé sur le serveur d'applications. Toutes les modifications apportées à votre code source dans le répertoire intermédiaire avec une commande conteneur seront incluses lorsque la source est déployée sur son emplacement final.

Pour résoudre les problèmes associés à vos commandes de conteneur, vous pouvez trouver leur sortie dans les journaux d'instance.

Utilisez l'option `leader_only` pour exécuter uniquement la commande sur une seule instance, ou configurer un `test` pour exécuter uniquement la commande lorsqu'une commande test a la valeur `true`. Les commandes de conteneur principales uniquement sont exécutées uniquement au cours de

la création et du déploiement de l'environnement, tandis que d'autres commandes et opérations de personnalisation de serveur sont effectuées chaque fois qu'une instance est mise en service ou mise à jour. Les commandes de conteneur principales uniquement ne sont pas exécutées pour lancer des modifications de configuration, par exemple, une modification d'ID d'AMI ou de type d'instance.

Syntaxe

```
container_commands:
  name of container_command:
    command: command to run
```

Options

`command`

Une chaîne ou un tableau de chaînes à exécuter.

`env`

(Facultatif) Définissez les variables d'environnement avant d'exécuter la commande, en ignorant toute valeur existante.

`cwd`

(Facultatif) Le répertoire de travail. Par défaut, il s'agit du répertoire intermédiaire de l'application décompressée.

`leader_only`

(Facultatif) Exécutez uniquement la commande sur une seule instance choisie par Elastic Beanstalk. Les commandes de conteneur principales uniquement sont exécutées avant d'autres commandes de conteneur. Une commande peut être principale uniquement ou disposer d'un `test`, mais pas les deux (`leader_only` a la priorité).

`test`

(Facultatif) Exécutez une commande test qui doit renvoyer la valeur `true` afin d'exécuter la commande de conteneur. Une commande peut être principale uniquement ou disposer d'un `test`, mais pas les deux (`leader_only` a la priorité).

`ignoreErrors`

(Facultatif) Ne manquez pas les déploiements si la commande de conteneur renvoie une valeur différente de 0 (réussite). Définissez sur `true` pour activer.

`waitAfterCompletion`

(Facultatif) Secondes d'attente une fois que la commande est terminée avant d'exécuter la commande suivante. La valeur par défaut est de **`60`** secondes. Vous pouvez également spécifier**`forever`**.

> **Important**
>
> Les redémarrages du système pendant le déploiement ne sont pas pris en charge. Si le système redémarre à la suite d'une commande, l'initialisation de l'instance échouera, entraînant l'échec du déploiement.
> Pour contourner le problème, vous pouvez l'utiliser `.ebextensions` configurationpour planifier des redémarrages une fois le déploiement terminé.

Exemple

L'exemple suivant enregistre la sortie de la commande `set` dans le fichier spécifié. Elastic Beanstalk exécute la commande sur une instance et redémarre l'instance immédiatement à la fin de la commande.

```
container_commands:
  foo:
    command: set > c:\\myapp\\set.txt
    leader_only: true
    waitAfterCompletion: 0
```

Ajout et personnalisation des ressources de l'environnement Elastic Beanstalk

Vous pouvez personnaliser les ressources de votre environnement qui font partie de votre environnement Elastic Beanstalk. Par exemple, vous pouvez ajouter une file d'attente Amazon SQS et une alarme indiquant la profondeur de la file d'attente, ou vous pouvez ajouter un cluster Amazon ElastiCache . Vous pouvez facilement personnaliser votre environnement en même temps que vous déployez votre version d'application, en incluant un fichier de configuration dans votre bundle source.

Vous pouvez utiliser la `Resources` clé d'un fichier de configuration pour créer et personnaliser AWS des ressources dans votre environnement. Les ressources définies dans les fichiers de configuration

sont ajoutées au CloudFormation modèle utilisé pour lancer votre environnement. Tous les types de CloudFormation ressources sont pris en charge.

> **Note**
>
> Chaque fois que vous ajoutez une ressource qui n'est pas gérée par Elastic Beanstalk, veillez à ajouter une politique utilisateur avec les autorisations appropriées Gestion des identités et des accès AWS pour vos utilisateurs (IAM). Les politiques d'utilisateur gérées fournies par Elastic Beanstalk ne couvrent que les autorisations pour les ressources gérées par Elastic Beanstalk.

Par exemple, le fichier de configuration suivant ajoute un hook de cycle de vie Auto Scaling au groupe Auto Scaling par défaut créé par Elastic Beanstalk :

`~/my-app/.ebextensions/as-hook.config`

```
Resources:
  hookrole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument: {
               "Version": "2012-10-17",
               "Statement": [ {
                  "Effect": "Allow",
                  "Principal": {
                     "Service": [ "autoscaling.amazonaws.com" ]
                  },
                  "Action": [ "sts:AssumeRole" ]
               } ]
            }
      Policies: [ {
               "PolicyName": "SNS",
               "PolicyDocument": {
                      "Version": "2012-10-17",
                      "Statement": [{
                          "Effect": "Allow",
                          "Resource": "*",
                          "Action": [
                              "sqs:SendMessage",
                              "sqs:GetQueueUrl",
                              "sns:Publish"
```

```
                        ]
                      }
                    ]
                }
             } ]
hooktopic:
  Type: AWS::SNS::Topic
  Properties:
    Subscription:
      - Endpoint: "my-email@example.com"
        Protocol: email
lifecyclehook:
  Type: AWS::AutoScaling::LifecycleHook
  Properties:
    AutoScalingGroupName: { "Ref" : "AWSEBAutoScalingGroup" }
    LifecycleTransition: autoscaling:EC2_INSTANCE_TERMINATING
    NotificationTargetARN: { "Ref" : "hooktopic" }
    RoleARN: { "Fn::GetAtt" : [ "hookrole", "Arn"] }
```

Cet exemple définit trois ressources : `hookrole`, `hooktopic` et `lifecyclehook`. Les deux premières ressources sont un rôle IAM, qui autorise Amazon EC2 Auto Scaling à publier des messages sur Amazon SNS, et une rubrique SNS, qui transmet les messages du groupe Auto Scaling vers une adresse e-mail. Elastic Beanstalk crée ces ressources avec les propriétés et les types spécifiés.

La ressource finale, `lifecyclehook`, correspond au hook du cycle de vie lui-même :

```
lifecyclehook:
  Type: AWS::AutoScaling::LifecycleHook
  Properties:
    AutoScalingGroupName: { "Ref" : "AWSEBAutoScalingGroup" }
    LifecycleTransition: autoscaling:EC2_INSTANCE_TERMINATING
    NotificationTargetARN: { "Ref" : "hooktopic" }
    RoleARN: { "Fn::GetAtt" : [ "hookrole", "Arn"] }
```

La définition du hook de cycle de vie utilise deux fonctions pour renseigner les valeurs relatives aux propriétés du hook. `{ "Ref" : "AWSEBAutoScalingGroup" }` récupère le nom du groupe Auto Scaling créé par Elastic Beanstalk pour l'environnement. `AWSEBAutoScalingGroup` est l'un des noms de ressource standard fournis par Elastic Beanstalk.

Pour `AWS::IAM::Role`, `Ref` renvoie uniquement le nom du rôle, pas l'ARN. Pour obtenir l'ARN du paramètre `RoleARN`, vous devez utiliser une autre fonction intrinsèque, `Fn::GetAtt`, qui permet

d'obtenir un attribut à partir d'une ressource. `RoleARN: { "Fn::GetAtt" : [ "hookrole", "Arn"] }` obtient l'attribut `Arn` depuis la ressource `hookrole`.

`{ "Ref" : "hooktopic" }` permet d'obtenir l'ARN de la rubrique Amazon SNS créée précédemment dans le fichier de configuration. La valeur renvoyée par `Ref` varie en fonction du type de ressource et se trouve dans la rubrique du Guide de CloudFormation l'utilisateur relative au type de AWS::SNS::Topic ressource.

Modification des ressources créées par Elastic Beanstalk pour votre environnement

Les ressources créées par Elastic Beanstalk pour votre environnement ont des noms. Vous pouvez utiliser ces noms pour obtenir des informations sur les ressources avec une fonction, ou modifier les propriétés sur les ressources pour personnaliser leur comportement. Cette rubrique décrit les AWS ressources utilisées par Elastic Beanstalk dans les différents types d'environnements.

> **Note**
>
> La rubrique précédente Ressources personnalisées fournit quelques cas d'utilisation et des exemples de personnalisation des ressources d'environnement. Vous pouvez également trouver d'autres exemples de fichiers de configuration dans la rubrique suivante Exemples de ressources personnalisées.

Les environnements de serveur Web disposent des ressources suivantes.

Environnements de serveur web

- `AWSEBAutoScalingGroup`(AWS::AutoScaling::AutoScalingGroupe) — Le groupe Auto Scaling rattaché à votre environnement.
- Une des deux ressources suivantes.
  - `AWSEBAutoScalingLaunchConfiguration`(AWS::AutoScaling::LaunchConfiguration) — La configuration de lancement attachée au groupe Auto Scaling de votre environnement.
  - `AWSEBEC2LaunchTemplate`(AWS::EC2::LaunchTemplate) — Le modèle de lancement Amazon EC2 utilisé par le groupe Auto Scaling de votre environnement.

> **Note**
>
> Si votre environnement utilise une fonctionnalité qui exige des modèles de lancement Amazon EC2 et que votre stratégie utilisateur ne dispose pas des autorisations requises, la

création ou la mise à jour de l'environnement peut échouer. Utilisez la AdministratorAccess politique des utilisateurs gérés par AWSElasticBeanstalk ou ajoutez les autorisations requises à votre politique personnalisée.

- `AWSEBEnvironmentName`(AWS::ElasticBeanstalk::Environment) — Votre environnement.
- `AWSEBSecurityGroup`(AWS::EC2::SecurityGroup) — Le groupe de sécurité attaché à votre groupe Auto Scaling.
- `AWSEBRDSDatabase`(AWS : :RDS : : DBInstance) — L'instance de base de données Amazon RDS attachée à votre environnement (le cas échéant).

Dans un environnement à charge équilibrée, vous pouvez accéder à des ressources supplémentaires liées à l'équilibreur de charge. Les équilibreurs de charge classiques disposent d'une ressource pour le programme d'équilibrage de charge et d'une autre pour le groupe de sécurité attaché à celui-ci. Les équilibreurs de charge d'application et de réseau ont des ressources supplémentaires pour l'écouteur par défaut, la règle de l'écouteur et le groupe cible de l'équilibreur de charge.

Environnements à charge équilibrée

- `AWSEBLoadBalancer`(AWS::ElasticLoadBalancing::LoadBalancer) — L'équilibreur de charge classique de votre environnement.
- `AWSEBV2LoadBalancer`(AWS::ElasticLoadBalancingV2::LoadBalancer) — Équilibreur de charge d'application ou de réseau de votre environnement.
- `AWSEBLoadBalancerSecurityGroup`(AWS::EC2::SecurityGroup) — Dans un Amazon Virtual Private Cloud (Amazon VPC) personnalisé uniquement, le nom du groupe de sécurité créé par Elastic Beanstalk pour l'équilibreur de charge. Dans un VPC par défaut ou dans EC2-Classic, Elastic Load Balancing attribue un groupe de sécurité par défaut à l'équilibreur de charge.
- `AWSEBV2LoadBalancerListener`(AWS::ElasticLoadBalancingV2::Listener) — Un écouteur qui permet à l'équilibreur de charge de vérifier les demandes de connexion et de les transmettre à un ou plusieurs groupes cibles.
- `AWSEBV2LoadBalancerListenerRule`(AWS::ElasticLoadBalancingV2::ListenerRule) — Définit les requêtes sur lesquelles un écouteur Elastic Load Balancing agit et les actions qu'il entreprend.
- `AWSEBV2LoadBalancerTargetGroup`(AWS::ElasticLoadBalancingV2::TargetGroup) — Un groupe cible Elastic Load Balancing qui achemine les demandes vers une ou plusieurs cibles enregistrées, telles que les instances Amazon EC2.

Les environnements de travail disposent de ressources pour la file d'attente SQS qui place en mémoire tampon les demandes entrantes, et d'une table Amazon DynamoDB utilisée par les instances pour effectuer le choix principal.

Environnements Worker

- `AWSEBWorkerQueue`(AWS::SQS::Queue) — La file d'attente Amazon SQS à partir de laquelle le démon extrait les demandes qui doivent être traitées.
- `AWSEBWorkerDeadLetterQueue`(AWS::SQS::Queue) — La file d'attente Amazon SQS qui stocke les messages qui ne peuvent pas être livrés ou qui n'ont pas été traités correctement par le daemon.
- `AWSEBWorkerCronLeaderRegistry`(AWS::DynamoDB::Table) — La table Amazon DynamoDB qui est le registre interne utilisé par le démon pour les tâches périodiques.

Autres clés CloudFormation de modèle

Nous avons déjà introduit des clés de fichier de configuration CloudFormation telles que `Resources`files, et`packages`. Elastic Beanstalk ajoute le contenu des fichiers CloudFormation de configuration au modèle compatible avec votre environnement. Vous pouvez ainsi CloudFormation utiliser d'autres sections pour effectuer des tâches avancées dans vos fichiers de configuration.

Clés

- Parameters
- Sorties
- Mappages

Parameters

Les paramètres constituent une alternative aux options personnalisées d'Elastic Beanstalk que vous pouvez utiliser pour définir des valeurs utilisées dans d'autres endroits de vos fichiers de configuration. À l'instar des options personnalisées, vous pouvez utiliser des paramètres pour rassembler des valeurs configurables par l'utilisateur dans un seul endroit. Contrairement aux options personnalisées, vous ne pouvez pas utiliser l'API d'Elastic Beanstalk pour définir des valeurs de paramètres, et le nombre de paramètres que vous pouvez définir dans un modèle est limité par. CloudFormation

L'une des raisons pour lesquelles vous souhaiterez peut-être utiliser des paramètres est de faire en sorte que vos fichiers de configuration servent également CloudFormation de modèles. Si vous utilisez des paramètres plutôt que des options personnalisées, vous pouvez utiliser le fichier de configuration pour créer la même ressource dans CloudFormation sa propre pile. Par exemple, vous pouvez disposer d'un fichier de configuration qui ajoute un système de fichiers Amazon EFS à votre environnement à des fins de test, puis utiliser le même fichier pour créer un système de fichiers indépendant non rattaché au cycle de vie de votre environnement pour une utilisation en production.

L'exemple suivant illustre l'utilisation de paramètres pour rassembler des valeurs configurables par l'utilisateur au début d'un fichier de configuration.

Example L oadbalancer-accesslogs-existingbucket .config — Paramètres

```
Parameters:
  bucket:
    Type: String
    Description: "Name of the Amazon S3 bucket in which to store load balancer logs"
    Default: "amzn-s3-demo-bucket"
  bucketprefix:
    Type: String
    Description: "Optional prefix. Can't start or end with a /, or contain the word
 AWSLogs"
    Default: ""
```

Sorties

Vous pouvez utiliser un bloc `Outputs` pour exporter des informations sur les ressources créées vers CloudFormation. Vous pouvez ensuite utiliser la `Fn::ImportValue` fonction pour extraire la valeur dans un CloudFormation modèle extérieur à Elastic Beanstalk.

L'exemple suivant crée une rubrique Amazon SNS et exporte son ARN vers celui-ci CloudFormation avec le nom. `NotificationTopicArn`

Example sns-topic.config

```
Resources:
  NotificationTopic:
    Type: AWS::SNS::Topic

Outputs:
  NotificationTopicArn:
    Description: Notification topic ARN
```

```
    Value: { "Ref" : "NotificationTopic" }
    Export:
      Name: NotificationTopicArn
```

Dans un fichier de configuration pour un autre environnement, ou dans un CloudFormation modèle externe à Elastic Beanstalk, vous pouvez `Fn::ImportValue` utiliser la fonction pour obtenir l'ARN exporté. Cet exemple affecte la valeur exportée à une propriété d'environnement nommée `TOPIC_ARN`.

Example env.config

```
option_settings:
  aws:elasticbeanstalk:application:environment:
    TOPIC_ARN: '`{ "Fn::ImportValue" : "NotificationTopicArn" }`'
```

Mappages

Vous pouvez utiliser un mappage pour stocker des paires clé/valeur organisées par espace de noms. Un mappage peut vous aider à organiser les valeurs que vous utilisez dans l'ensemble de vos configurations, ou à modifier une valeur de paramètre en fonction d'une autre valeur. Par exemple, la configuration suivante définit la valeur d'un paramètre d'ID de compte en fonction de la région en cours.

Example [oadbalancer-accesslogs-newbucketL.config — Mappages]

```
Mappings:
  Region2ELBAccountId:
    us-east-1:
      AccountId: "111122223333"
    us-west-2:
      AccountId: "444455556666"
    us-west-1:
      AccountId: "123456789012"
    eu-west-1:
      AccountId: "777788889999"
...
            Principal:
              AWS:
                ? "Fn::FindInMap"
                :
                  - Region2ELBAccountId
                  -
```

```
                  Ref: "AWS::Region"
                - AccountId
```

Fonctions

Vous pouvez utiliser des fonctions dans vos fichiers de configuration pour renseigner les valeurs des propriétés de ressource à l'aide des informations issues d'autres ressources ou de paramètres d'option de configuration Elastic Beanstalk. Elastic CloudFormation Beanstalk prend en charge les `Ref` fonctions `Fn::GetAtt` (`Fn::Join`,,) et une fonction spécifique à Elastic Beanstalk,. `Fn::GetOptionSetting`

Fonctions

- Réf
- Ventilateur : GetAtt
- Fn::Join
- Ventilateur : GetOptionSetting

Réf

Permet `Ref` de récupérer la représentation sous forme de chaîne par défaut d'une AWS ressource. La valeur renvoyée par `Ref` dépend du type de ressource et parfois d'autres facteurs également. Par exemple, un groupe de sécurité (AWS::EC2::SecurityGroup) renvoie le nom ou l'ID du groupe de sécurité, selon que le groupe de sécurité se trouve dans un Amazon Virtual Private Cloud (Amazon VPC) par défaut, un VPC EC2 classique ou un VPC personnalisé.

```
{ "Ref" : "resource name" }
```

> **Note**
>
> Pour plus d'informations sur chaque type de ressource, y compris la ou les valeurs de renvoi de `Ref`, consultez Référence des types de ressources AWS dans le Guide de l'utilisateur CloudFormation .

À partir de l'exemple de hook de cycle de vie Auto Scaling :

```
Resources:
```

```
  lifecyclehook:
    Type: AWS::AutoScaling::LifecycleHook
    Properties:
      AutoScalingGroupName: { "Ref" : "AWSEBAutoScalingGroup" }
```

Vous pouvez également l'utiliser `Ref` pour récupérer la valeur d'un CloudFormation paramètre défini ailleurs dans le même fichier ou dans un autre fichier de configuration.

Ventilateur : GetAtt

Permet `Fn::GetAtt` de récupérer la valeur d'un attribut d'une AWS ressource.

```
{ "Fn::GetAtt" : [ "resource name", "attribute name"] }
```

À partir de l'exemple de hook de cycle de vie Auto Scaling :

```
Resources:
  lifecyclehook:
    Type: AWS::AutoScaling::LifecycleHook
    Properties:
      RoleARN: { "Fn::GetAtt" : [ "hookrole", "Arn"] }
```

Voir Fn : : GetAtt pour plus d'informations.

Fn::Join

Utilisez `Fn::Join` pour associer des chaînes avec un délimiteur. Les chaînes peuvent être codées en dur ou utiliser le résultat de `Fn::GetAtt` ou `Ref`.

```
{ "Fn::Join" : [ "delimiter", [ "string1", "string2" ] ] }
```

Pour plus d'informations, consultez Fn::Join.

Ventilateur : GetOptionSetting

Utilisez `Fn::GetOptionSetting` pour récupérer la valeur d'un paramètre d'option de configuration appliqué à l'environnement.

```
"Fn::GetOptionSetting":
  Namespace: "namespace"
  OptionName: "option name"
  DefaultValue: "default value"
```

À partir de l'exemple stockage de clés privées :

```
Resources:
  AWSEBAutoScalingGroup:
    Metadata:
      AWS::CloudFormation::Authentication:
        S3Auth:
          type: "s3"
          buckets: ["elasticbeanstalk-us-west-2-123456789012"]
          roleName:
            "Fn::GetOptionSetting":
              Namespace: "aws:autoscaling:launchconfiguration"
              OptionName: "IamInstanceProfile"
              DefaultValue: "aws-elasticbeanstalk-ec2-role"
```

Exemples de ressources personnalisées

Voici une liste d'exemples de fichiers de configuration que vous pouvez utiliser pour personnaliser vos environnements Elastic Beanstalk :

- DynamoDB et SNS CloudWatch
- Elastic Load Balancing et CloudWatch
- ElastiCache
- RDS et CloudWatch
- SQS, SNS et CloudWatch

Les sous-rubriques de cette page fournissent des exemples étendus pour l'ajout et la configuration de ressources personnalisées dans un environnement Elastic Beanstalk.

Exemples

- Exemple : ElastiCache
- Exemple : SQS et CloudWatch SNS
- Exemple : DynamoDB, et SNS CloudWatch

Exemple : ElastiCache

Les exemples suivants ajoutent un ElastiCache cluster Amazon aux plateformes EC2-Classic et EC2-VPC (Amazon Virtual Private Cloud (Amazon VPC) par défaut et personnalisées). Pour plus

d'informations sur ces plateformes et sur la manière de déterminer celles prises en charge par EC2 pour votre région et votre AWS compte, consultezhttps://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-supported-platforms.html. Consultez ensuite la section de cette rubrique qui s'applique à votre plateforme.

- Plateformes EC2-classic
- EC2-VPC (par défaut)
- EC2-VPC (personnalisé)

Plateformes EC2-classic

Cet exemple ajoute un ElastiCache cluster Amazon à un environnement avec des instances lancées sur la plateforme EC2-Classic. Toutes les propriétés répertoriées dans cet exemple correspondent aux propriétés requises minimales qui doivent être définies pour chaque type de ressource. Vous pouvez télécharger l'exemple ElastiCacheci-dessous.

> **Note**
>
> Cet exemple crée AWS des ressources qui peuvent vous être facturées. Pour plus d'informations sur la AWS tarification, consultezhttps://aws.amazon.com/pricing/. Certains services font partie du niveau d'utilisation AWS gratuit. Si vous êtes un nouveau client, vous pouvez essayer ces services gratuitement. Pour plus d'informations, consultez https://aws.amazon.com/free/.

Pour utiliser cet exemple, procédez comme suit :

1. Créez un répertoire `.ebextensions` dans le répertoire de niveau supérieur de votre bundle de fichiers source.
2. Créez deux fichiers de configuration avec l'extension `.config`, puis placez-les dans votre répertoire `.ebextensions`. Un fichier de configuration définit les ressources et l'autre fichier de configuration définit les options.
3. Déployez votre application sur Elastic Beanstalk.

   YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Créez un fichier de configuration (par exemple, `elasticache.config`) qui définit les ressources. Dans cet exemple, nous créons le ElastiCache cluster en spécifiant le nom de la ressource du ElastiCache cluster (`MyElastiCache`), en déclarant son type, puis en configurant les propriétés du cluster. L'exemple fait référence au nom de la ressource du groupe de ElastiCache sécurité créée et définie dans ce fichier de configuration. Ensuite, nous créons un groupe ElastiCache de sécurité. Nous définissons le nom de cette ressource, nous déclarons son type, puis nous ajoutons une description du groupe de sécurité. Enfin, nous avons défini les règles d'entrée pour le groupe de ElastiCache sécurité afin d'autoriser l'accès uniquement depuis les instances du groupe de ElastiCache sécurité (`MyCacheSecurityGroup`) et du groupe de sécurité Elastic Beanstalk (). `AWSEBSecurityGroup` Le nom du paramètre, `AWSEBSecurityGroup`, est un nom de ressource fixe fourni par Elastic Beanstalk. Vous devez ajouter des règles d'entrée `AWSEBSecurityGroup` à votre groupe de ElastiCache sécurité pour que votre application Elastic Beanstalk puisse se connecter aux instances de votre cluster. ElastiCache

```
#This sample requires you to create a separate configuration file that defines the
 custom option settings for CacheCluster properties.

Resources:
  MyElastiCache:
    Type: AWS::ElastiCache::CacheCluster
    Properties:
      CacheNodeType:
         Fn::GetOptionSetting:
             OptionName : CacheNodeType
             DefaultValue: cache.m1.small
      NumCacheNodes:
           Fn::GetOptionSetting:
             OptionName : NumCacheNodes
             DefaultValue: 1
      Engine:
           Fn::GetOptionSetting:
             OptionName : Engine
             DefaultValue: memcached
      CacheSecurityGroupNames:
        - Ref: MyCacheSecurityGroup
  MyCacheSecurityGroup:
    Type: AWS::ElastiCache::SecurityGroup
    Properties:
      Description: "Lock cache down to webserver access only"
  MyCacheSecurityGroupIngress:
    Type: AWS::ElastiCache::SecurityGroupIngress
```

```
    Properties:
      CacheSecurityGroupName:
        Ref: MyCacheSecurityGroup
      EC2SecurityGroupName:
        Ref: AWSEBSecurityGroup
```

Pour plus d'informations sur les ressources utilisées dans cet exemple de fichier de configuration, consultez les références suivantes :

- AWS::ElastiCache::CacheCluster
- AWS::ElastiCache::SecurityGroup
- AWS::ElastiCache : SecurityGroupIngress

Créez un fichier de configuration distinct nommé `options.config` et définissez les paramètres d'option personnalisés.

```
option_settings:
  "aws:elasticbeanstalk:customoption":
     CacheNodeType : cache.m1.small
     NumCacheNodes : 1
     Engine : memcached
```

Ces lignes indiquent à Elastic Beanstalk d'obtenir les valeurs des propriétés,, et Engine à CacheNodeTypepartir NumCacheNodes des valeurs CacheNodeType, et Engine dans un fichier de configuration (options.config dans notre exemple) qui contient une section option_settings avec une section aws:elasticbeanstalk:customoption contenant une paire nom-valeur contenant la valeur réelle à utiliser. NumCacheNodes Dans l'exemple ci-dessus, cela signifie que les éléments cache.m1.small, 1 et memcached seraient utilisés pour les valeurs. Pour plus d'informations sur `Fn::GetOptionSetting`, consultez Fonctions.

EC2-VPC (par défaut)

Cet exemple ajoute un ElastiCache cluster Amazon à un environnement avec des instances lancées sur la plateforme EC2-VPC. Plus précisément, les informations de cette section s'appliquent à un scénario où EC2 lance des instances dans le VPC par défaut. Toutes les propriétés figurant dans cet exemple correspondent aux propriétés requises minimales qui doivent être définies pour chaque type de ressource. Pour plus d'informations sur le VPC et les VPCs sous-réseaux par défaut, consultez la section Votre VPC et vos sous-réseaux par défaut.

> **Note**
>
> Cet exemple crée AWS des ressources qui peuvent vous être facturées. Pour plus d'informations sur la AWS tarification, consultezhttps://aws.amazon.com/pricing/. Certains services font partie du niveau d'utilisation AWS gratuit. Si vous êtes un nouveau client, vous pouvez essayer ces services gratuitement. Pour plus d'informations, consultez https://aws.amazon.com/free/.

Pour utiliser cet exemple, procédez comme suit :

1. Créez un répertoire `.ebextensions` dans le répertoire de niveau supérieur de votre bundle de fichiers source.
2. Créez deux fichiers de configuration avec l'extension `.config`, puis placez-les dans votre répertoire `.ebextensions`. Un fichier de configuration définit les ressources et l'autre fichier de configuration définit les options.
3. Déployez votre application sur Elastic Beanstalk.

   YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Ensuite, attribuez le nom `elasticache.config` au fichier de configuration de ressources. Pour créer le ElastiCache cluster, cet exemple indique le nom de la ressource du ElastiCache cluster (`MyElastiCache`), déclare son type, puis configure les propriétés du cluster. L'exemple fait référence à l'ID de la ressource de groupe de sécurité que nous créons et définissons dans ce fichier de configuration.

Ensuite, nous créons un groupe de sécurité EC2. Nous définissons le nom de cette ressource, déclarons son type, ajoutons une description et définissons les règles de trafic entrant pour le groupe de sécurité afin de n'autoriser l'accès qu'à partir des instances incluses dans le groupe de sécurité Elastic Beanstalk (`AWSEBSecurityGroup`). (Le nom du paramètre`AWSEBSecurityGroup`, est un nom de ressource fixe fourni par Elastic Beanstalk. Vous devez ajouter des règles d'entrée `AWSEBSecurityGroup` à votre groupe de ElastiCache sécurité pour que votre application Elastic Beanstalk puisse se connecter aux instances de votre cluster.) ElastiCache

Les règles de trafic entrant pour le groupe de sécurité EC2 définissent également le protocole IP et les numéros de port sur lesquels les nœuds de cache peuvent accepter des connexions. Pour Redis, le numéro de port par défaut est 6379.

```
#This sample requires you to create a separate configuration file that defines the
 custom option settings for CacheCluster properties.

Resources:
  MyCacheSecurityGroup:
    Type: "AWS::EC2::SecurityGroup"
    Properties:
      GroupDescription: "Lock cache down to webserver access only"
      SecurityGroupIngress :
        - IpProtocol : "tcp"
          FromPort :
            Fn::GetOptionSetting:
              OptionName : "CachePort"
              DefaultValue: "6379"
          ToPort :
            Fn::GetOptionSetting:
              OptionName : "CachePort"
              DefaultValue: "6379"
          SourceSecurityGroupName:
            Ref: "AWSEBSecurityGroup"
  MyElastiCache:
    Type: "AWS::ElastiCache::CacheCluster"
    Properties:
      CacheNodeType:
        Fn::GetOptionSetting:
          OptionName : "CacheNodeType"
          DefaultValue : "cache.t2.micro"
      NumCacheNodes:
        Fn::GetOptionSetting:
          OptionName : "NumCacheNodes"
          DefaultValue : "1"
      Engine:
        Fn::GetOptionSetting:
          OptionName : "Engine"
          DefaultValue : "redis"
      VpcSecurityGroupIds:
        -
          Fn::GetAtt:
            - MyCacheSecurityGroup
```

```
            - GroupId

Outputs:
  ElastiCache:
    Description : "ID of ElastiCache Cache Cluster with Redis Engine"
    Value :
      Ref : "MyElastiCache"
```

Pour plus d'informations sur les ressources utilisées dans cet exemple de fichier de configuration, consultez les références suivantes :

- AWS::ElastiCache::CacheCluster
- AWS::EC2::SecurityGroup

Ensuite, attribuez le nom `options.config` au fichier de configuration des options et définissez les paramètres d'option personnalisés.

```
option_settings:
  "aws:elasticbeanstalk:customoption":
    CacheNodeType : cache.t2.micro
    NumCacheNodes : 1
    Engine : redis
    CachePort : 6379
```

Avec ces lignes de code, Elastic Beanstalk récupère les valeurs des propriétés `CacheNodeType`, `NumCacheNodes`, `Engine` et `CachePort` à partir des valeurs `CacheNodeType`, `NumCacheNodes`, `Engine` et `CachePort` figurant dans un fichier de configuration (`options.config` dans notre exemple). Ce fichier inclut une section `aws:elasticbeanstalk:customoption` (sous `option_settings`) qui contient des paires nom-valeur incluant les valeurs réelles à utiliser. Dans l'exemple précédent, les éléments `cache.t2.micro`, `1`, `redis` et `6379` seraient utilisés pour les valeurs. Pour plus d'informations sur `Fn::GetOptionSetting`, consultez Fonctions.

EC2-VPC (personnalisé)

Si vous créez un VPC personnalisé sur la plateforme EC2-VPC et que vous le spécifiez comme VPC dans lequel EC2 lance les instances, le processus d'ajout d'un ElastiCache cluster Amazon à votre environnement est différent de celui d'un VPC par défaut. La principale différence est que vous devez créer un groupe de sous-réseaux pour le ElastiCache cluster. Toutes les propriétés figurant dans cet

exemple correspondent aux propriétés requises minimales qui doivent être définies pour chaque type de ressource.

> **Note**
>
> Cet exemple crée AWS des ressources qui peuvent vous être facturées. Pour plus d'informations sur la AWS tarification, consultezhttps://aws.amazon.com/pricing/. Certains services font partie du niveau d'utilisation AWS gratuit. Si vous êtes un nouveau client, vous pouvez essayer ces services gratuitement. Pour plus d'informations, consultez https://aws.amazon.com/free/.

Pour utiliser cet exemple, procédez comme suit :

1. Créez un répertoire `.ebextensions` dans le répertoire de niveau supérieur de votre bundle de fichiers source.
2. Créez deux fichiers de configuration avec l'extension `.config`, puis placez-les dans votre répertoire `.ebextensions`. Un fichier de configuration définit les ressources et l'autre fichier de configuration définit les options.
3. Déployez votre application sur Elastic Beanstalk.

   YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Ensuite, attribuez le nom `elasticache.config` au fichier de configuration de ressources. Pour créer le ElastiCache cluster, cet exemple indique le nom de la ressource du ElastiCache cluster (`MyElastiCache`), déclare son type, puis configure les propriétés du cluster. Les propriétés de l'exemple font référence au nom du groupe de sous-réseaux du ElastiCache cluster ainsi qu'à l'ID de la ressource du groupe de sécurité que nous créons et définissons dans ce fichier de configuration.

Ensuite, nous créons un groupe de sécurité EC2. Nous définissons le nom de cette ressource, déclarons son type, ajoutons une description et l'ID de VPC, et définissons les règles de trafic entrant pour le groupe de sécurité afin de n'autoriser l'accès qu'à partir des instances incluses dans le groupe de sécurité Elastic Beanstalk (`AWSEBSecurityGroup`). (Le nom du paramètre`AWSEBSecurityGroup`, est un nom de ressource fixe fourni par Elastic Beanstalk. Vous devez ajouter des règles d'entrée `AWSEBSecurityGroup` à votre groupe de ElastiCache sécurité

pour que votre application Elastic Beanstalk puisse se connecter aux instances de votre cluster.) ElastiCache

Les règles de trafic entrant pour le groupe de sécurité EC2 définissent également le protocole IP et les numéros de port sur lesquels les nœuds de cache peuvent accepter des connexions. Pour Redis, le numéro de port par défaut est 6379. Enfin, cet exemple crée un groupe de sous-réseaux pour le ElastiCache cluster. Nous définissons le nom de cette ressource, déclarons son type et ajoutons la description et l'ID du sous-réseau dans le groupe de sous-réseaux.

> **Note**
>
> Nous vous recommandons d'utiliser des sous-réseaux privés pour le ElastiCache cluster. Pour plus d'informations sur un VPC avec un sous-réseau privé, consultez https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Scenario2.html.

```
#This sample requires you to create a separate configuration file that defines the
 custom option settings for CacheCluster properties.

Resources:
  MyElastiCache:
    Type: "AWS::ElastiCache::CacheCluster"
    Properties:
      CacheNodeType:
        Fn::GetOptionSetting:
          OptionName : "CacheNodeType"
          DefaultValue : "cache.t2.micro"
      NumCacheNodes:
        Fn::GetOptionSetting:
          OptionName : "NumCacheNodes"
          DefaultValue : "1"
      Engine:
        Fn::GetOptionSetting:
          OptionName : "Engine"
          DefaultValue : "redis"
      CacheSubnetGroupName:
        Ref: "MyCacheSubnets"
      VpcSecurityGroupIds:
        - Ref: "MyCacheSecurityGroup"
  MyCacheSecurityGroup:
    Type: "AWS::EC2::SecurityGroup"
```

```
    Properties:
      GroupDescription: "Lock cache down to webserver access only"
      VpcId:
        Fn::GetOptionSetting:
          OptionName : "VpcId"
      SecurityGroupIngress :
        - IpProtocol : "tcp"
          FromPort :
            Fn::GetOptionSetting:
              OptionName : "CachePort"
              DefaultValue: "6379"
          ToPort :
            Fn::GetOptionSetting:
              OptionName : "CachePort"
              DefaultValue: "6379"
          SourceSecurityGroupId:
            Ref: "AWSEBSecurityGroup"
  MyCacheSubnets:
    Type: "AWS::ElastiCache::SubnetGroup"
    Properties:
      Description: "Subnets for ElastiCache"
      SubnetIds:
        Fn::GetOptionSetting:
          OptionName : "CacheSubnets"
Outputs:
  ElastiCache:
    Description : "ID of ElastiCache Cache Cluster with Redis Engine"
    Value :
      Ref : "MyElastiCache"
```

Pour plus d'informations sur les ressources utilisées dans cet exemple de fichier de configuration, consultez les références suivantes :

- AWS::ElastiCache::CacheCluster
- AWS::EC2::SecurityGroup
- AWS::ElastiCache::SubnetGroup

Ensuite, attribuez le nom `options.config` au fichier de configuration des options et définissez les paramètres d'option personnalisés.

> **Note**
>
> Dans l'exemple suivant, remplacez les exemples de valeurs `CacheSubnets` et `VpcId` par vos propres sous-réseaux et VPC.

```
option_settings:
  "aws:elasticbeanstalk:customoption":
    CacheNodeType : cache.t2.micro
    NumCacheNodes : 1
    Engine : redis
    CachePort : 6379
    CacheSubnets:
      - subnet-1a1a1a1a
      - subnet-2b2b2b2b
      - subnet-3c3c3c3c
    VpcId: vpc-4d4d4d4d
```

Avec ces lignes de code, Elastic Beanstalk récupère les valeurs des propriétés `CacheNodeType`, `NumCacheNodes`, `Engine`, `CachePort`, `CacheSubnets` et `VpcId` à partir des valeurs `CacheNodeType`, `NumCacheNodes`, `Engine`, `CachePort`, `CacheSubnets` et `VpcId` figurant dans un fichier de configuration (`options.config` dans notre exemple). Ce fichier inclut une section `aws:elasticbeanstalk:customoption` (sous `option_settings`) qui contient des paires nom-valeur incluant des exemples de valeurs. Dans l'exemple ci-dessus, les éléments `cache.t2.micro`, `1`, `redis`, `6379`, `subnet-1a1a1a1a`, `subnet-2b2b2b2b`, `subnet-3c3c3c3c` et `vpc-4d4d4d4d` seraient utilisés pour les valeurs. Pour plus d'informations sur `Fn::GetOptionSetting`, consultez Fonctions.

Exemple : SQS et CloudWatch SNS

Dans cet exemple, une file d'attente Amazon SQS et une alarme relative à la longueur de la file d'attente sont ajoutées à l'environnement. Les propriétés que vous voyez dans cet exemple correspondent aux propriétés minimales requises que vous devez définir pour chacune de ces ressources. Vous pouvez télécharger l'exemple sur SQS, SNS et. CloudWatch

> **Note**
>
> Cet exemple crée AWS des ressources qui peuvent vous être facturées. Pour plus d'informations sur la AWS tarification, consultezhttps://aws.amazon.com/pricing/. Certains

services font partie du niveau d'utilisation AWS gratuit. Si vous êtes un nouveau client, vous pouvez essayer ces services gratuitement. Pour plus d'informations, consultez https://aws.amazon.com/free/.

Pour utiliser cet exemple, procédez comme suit :

1. Créez un répertoire `.ebextensions` dans le répertoire de niveau supérieur de votre bundle de fichiers source.
2. Créez deux fichiers de configuration avec l'extension `.config`, puis placez-les dans votre répertoire `.ebextensions`. Un fichier de configuration définit les ressources et l'autre fichier de configuration définit les options.
3. Déployez votre application sur Elastic Beanstalk.

   YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Créez un fichier de configuration (par exemple, sqs.config) qui définit les ressources. Dans cet exemple, nous créons une file d'attente SQS et définissons la propriété `VisbilityTimeout` dans la ressource `MySQSQueue`. Ensuite, nous créons une rubrique (`Topic`) SNS et nous indiquons que l'e-mail doit être envoyé à `someone@example.com` lorsque l'alarme se déclenche. Enfin, nous créons une CloudWatch alarme si la file d'attente dépasse 10 messages. Dans la propriété `Dimensions`, nous spécifions le nom de la dimension et la valeur représentant la mesure de la dimension. Nous utilisons `Fn::GetAtt` pour renvoyer la valeur de `QueueName` à partir de `MySQSQueue`.

```
#This sample requires you to create a separate configuration file to define the custom
 options for the SNS topic and SQS queue.
Resources:
  MySQSQueue:
    Type: AWS::SQS::Queue
    Properties:
      VisibilityTimeout:
        Fn::GetOptionSetting:
          OptionName: VisibilityTimeout
          DefaultValue: 30
  AlarmTopic:
    Type: AWS::SNS::Topic
    Properties:
```

```
      Subscription:
        - Endpoint:
            Fn::GetOptionSetting:
              OptionName: AlarmEmail
              DefaultValue: "nobody@amazon.com"
          Protocol: email
  QueueDepthAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmDescription: "Alarm if queue depth grows beyond 10 messages"
      Namespace: "AWS/SQS"
      MetricName: ApproximateNumberOfMessagesVisible
      Dimensions:
        - Name: QueueName
          Value : { "Fn::GetAtt" : [ "MySQSQueue", "QueueName"] }
      Statistic: Sum
      Period: 300
      EvaluationPeriods: 1
      Threshold: 10
      ComparisonOperator: GreaterThanThreshold
      AlarmActions:
        - Ref: AlarmTopic
      InsufficientDataActions:
        - Ref: AlarmTopic

Outputs :
  QueueURL:
    Description : "URL of newly created SQS Queue"
    Value : { Ref : "MySQSQueue" }
  QueueARN :
    Description : "ARN of newly created SQS Queue"
    Value : { "Fn::GetAtt" : [ "MySQSQueue", "Arn"]}
  QueueName :
    Description : "Name newly created SQS Queue"
    Value : { "Fn::GetAtt" : [ "MySQSQueue", "QueueName"]}
```

Pour plus d'informations sur les ressources utilisées dans cet exemple de fichier de configuration, consultez les références suivantes :

- AWS::SQS::Queue
- AWS::SNS::Topic
- AWS::CloudWatch::Alarm

Créez un fichier de configuration distinct nommé `options.config` et définissez les paramètres d'option personnalisés.

```
option_settings:
  "aws:elasticbeanstalk:customoption":
     VisibilityTimeout : 30
     AlarmEmail : "nobody@example.com"
```

Ces lignes indiquent à Elastic Beanstalk d'obtenir les valeurs des propriétés et du point de terminaison d'abonnement à VisibilityTimeout partir des valeurs VisibilityTimeout du point de terminaison d'abonnement dans un fichier de configuration (options.config dans notre exemple) qui contient une section option_settings avec une section aws:elasticbeanstalk:customoption contenant une paire nom-valeur contenant la valeur réelle à utiliser. Dans l'exemple ci-dessus, cela signifie que les valeurs 30 et « nobody@amazon.com » seraient utilisées. Pour plus d'informations sur `Fn::GetOptionSetting`, consultez the section called “Fonctions”.

Exemple : DynamoDB, et SNS CloudWatch

Ce fichier de configuration définit la table DynamoDB en tant que gestionnaire de session pour une application basée sur PHP utilisant le SDK pour PHP 2. AWS Pour utiliser cet exemple, vous devez disposer d'un profil d'instance IAM, qui est ajouté aux instances de votre environnement et utilisé pour accéder à la table DynamoDB.

Vous pouvez télécharger ici l'exemple que nous utiliserons dans cette étape : Exemple de prise en charge de session DynamoDB. L'exemple contient les fichiers suivants :

- Exemple d'application, `index.php`
- Un fichier de configuration`dynamodb.config`, pour créer et configurer une table DynamoDB et d' AWS autres ressources et pour installer le logiciel sur les instances EC2 qui hébergent l'application dans un environnement Elastic Beanstalk
- Un fichier de configuration `options.config` qui remplace les valeurs par défaut dans `dynamodb.config` par des paramètres propres à cette installation spécifique

`index.php`

```
<?php

// Include the SDK using the Composer autoloader
require '../vendor/autoload.php';
```

```
use Aws\DynamoDb\DynamoDbClient;

// Grab the session table name and region from the configuration file
list($tableName, $region) = file(__DIR__ . '/../sessiontable');
$tableName = rtrim($tableName);
$region = rtrim($region);

// Create a DynamoDB client and register the table as the session handler
$dynamodb = DynamoDbClient::factory(array('region' => $region));
$handler = $dynamodb->registerSessionHandler(array('table_name' => $tableName,
 'hash_key' => 'username'));

// Grab the instance ID so we can display the EC2 instance that services the request
$instanceId = file_get_contents("http://169.254.169.254/latest/meta-data/instance-id");
?>
<h1>Elastic Beanstalk PHP Sessions Sample</h1>
<p>This sample application shows the integration of the Elastic Beanstalk PHP
container and the session support for DynamoDB from the AWS SDK for PHP 2.
Using DynamoDB session support, the application can be scaled out across
multiple web servers. For more details, see the
<a href="https://aws.amazon.com/php/">PHP Developer Center</a>.</p>

<form id="SimpleForm" name="SimpleForm" method="post" action="index.php">
<?php
echo 'Request serviced from instance ' . $instanceId . '<br/>';
echo '<br/>';

if (isset($_POST['continue'])) {
  session_start();
  $_SESSION['visits'] = $_SESSION['visits'] + 1;
  echo 'Welcome back ' . $_SESSION['username'] . '<br/>';
  echo 'This is visit number ' . $_SESSION['visits'] . '<br/>';
  session_write_close();
  echo '<br/>';
  echo '<input type="Submit" value="Refresh" name="continue" id="continue"/>';
  echo '<input type="Submit" value="Delete Session" name="killsession"
 id="killsession"/>';
} elseif (isset($_POST['killsession'])) {
  session_start();
  echo 'Goodbye ' . $_SESSION['username'] . '<br/>';
  session_destroy();
  echo 'Username: <input type="text" name="username" id="username" size="30"/><br/>';
  echo '<br/>';
```

```
  echo '<input type="Submit" value="New Session" name="newsession" id="newsession"/>';
} elseif (isset($_POST['newsession'])) {
  session_start();
  $_SESSION['username'] = $_POST['username'];
  $_SESSION['visits'] = 1;
  echo 'Welcome to a new session ' . $_SESSION['username'] . '<br/>';
  session_write_close();
  echo '<br/>';
  echo '<input type="Submit" value="Refresh" name="continue" id="continue"/>';
  echo '<input type="Submit" value="Delete Session" name="killsession"
 id="killsession"/>';
} else {
  echo 'To get started, enter a username.<br/>';
  echo '<br/>';
  echo 'Username: <input type="text" name="username" id="username" size="30"/><br/>';
  echo '<input type="Submit" value="New Session" name="newsession" id="newsession"/>';
}
?>
</form>
```

.ebextensions/dynamodb.config

```
Resources:
  SessionTable:
    Type: AWS::DynamoDB::Table
    Properties:
      KeySchema:
        HashKeyElement:
          AttributeName:
            Fn::GetOptionSetting:
              OptionName : SessionHashKeyName
              DefaultValue: "username"
          AttributeType:
            Fn::GetOptionSetting:
              OptionName : SessionHashKeyType
              DefaultValue: "S"
      ProvisionedThroughput:
        ReadCapacityUnits:
          Fn::GetOptionSetting:
            OptionName : SessionReadCapacityUnits
            DefaultValue: 1
        WriteCapacityUnits:
          Fn::GetOptionSetting:
```

```
            OptionName : SessionWriteCapacityUnits
            DefaultValue: 1

  SessionWriteCapacityUnitsLimit:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmDescription: { "Fn::Join" : ["", [{ "Ref" : "AWSEBEnvironmentName" }, "
write capacity limit on the session table." ]]}
      Namespace: "AWS/DynamoDB"
      MetricName: ConsumedWriteCapacityUnits
      Dimensions:
        - Name: TableName
          Value: { "Ref" : "SessionTable" }
      Statistic: Sum
      Period: 300
      EvaluationPeriods: 12
      Threshold:
          Fn::GetOptionSetting:
            OptionName : SessionWriteCapacityUnitsAlarmThreshold
            DefaultValue: 240
      ComparisonOperator: GreaterThanThreshold
      AlarmActions:
        - Ref: SessionAlarmTopic
      InsufficientDataActions:
        - Ref: SessionAlarmTopic

  SessionReadCapacityUnitsLimit:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmDescription: { "Fn::Join" : ["", [{ "Ref" : "AWSEBEnvironmentName" }, " read
capacity limit on the session table." ]]}
      Namespace: "AWS/DynamoDB"
      MetricName: ConsumedReadCapacityUnits
      Dimensions:
        - Name: TableName
          Value: { "Ref" : "SessionTable" }
      Statistic: Sum
      Period: 300
      EvaluationPeriods: 12
      Threshold:
          Fn::GetOptionSetting:
            OptionName : SessionReadCapacityUnitsAlarmThreshold
            DefaultValue: 240
      ComparisonOperator: GreaterThanThreshold
```

```
      AlarmActions:
        - Ref: SessionAlarmTopic
      InsufficientDataActions:
        - Ref: SessionAlarmTopic

  SessionThrottledRequestsAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmDescription: { "Fn::Join" : ["", [{ "Ref" : "AWSEBEnvironmentName" }, ":
 requests are being throttled." ]]}
      Namespace: AWS/DynamoDB
      MetricName: ThrottledRequests
      Dimensions:
        - Name: TableName
          Value: { "Ref" : "SessionTable" }
      Statistic: Sum
      Period: 300
      EvaluationPeriods: 1
      Threshold:
        Fn::GetOptionSetting:
          OptionName: SessionThrottledRequestsThreshold
          DefaultValue: 1
      ComparisonOperator: GreaterThanThreshold
      AlarmActions:
        - Ref: SessionAlarmTopic
      InsufficientDataActions:
        - Ref: SessionAlarmTopic

  SessionAlarmTopic:
    Type: AWS::SNS::Topic
    Properties:
      Subscription:
        - Endpoint:
            Fn::GetOptionSetting:
              OptionName: SessionAlarmEmail
              DefaultValue: "nobody@amazon.com"
          Protocol: email

files:
  "/var/app/sessiontable":
    mode: "000444"
    content: |
      `{"Ref" : "SessionTable"}`
      `{"Ref" : "AWS::Region"}`
```

```
  "/var/app/composer.json":
    mode: "000744"
    content:
      {
        "require": {
           "aws/aws-sdk-php": "*"
        }
      }

container_commands:
 "1-install-composer":
   command: "cd /var/app; curl -s http://getcomposer.org/installer | php"
 "2-install-dependencies":
   command: "cd /var/app; php composer.phar install"
 "3-cleanup-composer":
   command: "rm -Rf /var/app/composer.*"
```

Dans le modèle de fichier de configuration, nous commençons par créer la table DynamoDB et configurer la structure de clé primaire pour la table et les unités de capacités afin d'allouer des ressources suffisantes pour offrir le débit demandé. Ensuite, nous créons des CloudWatch alarmes pour `WriteCapacity` et`ReadCapacity`. Nous créons une rubrique SNS qui envoie un e-mail à « nobody@amazon.com » si le seuil d'alarme est dépassé.

Après avoir créé et configuré nos AWS ressources pour notre environnement, nous devons personnaliser les instances EC2. Nous utilisons cette `files` clé pour transmettre les détails de la table DynamoDB aux instances EC2 de notre environnement et pour ajouter un « require » dans `composer.json` le fichier du SDK AWS pour PHP 2. Enfin, nous exécutons des commandes de conteneur pour installer Composer, les dépendances requises, puis nous supprimons le programme d'installation.

`.ebextensions/options.config`

```
option_settings:
  "aws:elasticbeanstalk:customoption":
     SessionHashKeyName                     : username
     SessionHashKeyType                     : S
     SessionReadCapacityUnits               : 1
     SessionReadCapacityUnitsAlarmThreshold  : 240
     SessionWriteCapacityUnits              : 1
     SessionWriteCapacityUnitsAlarmThreshold : 240
     SessionThrottledRequestsThreshold      : 1
```

```
    SessionAlarmEmail                       : me@example.com
```

Remplacez la SessionAlarmEmail valeur par l'e-mail dans lequel vous souhaitez que les notifications d'alarme soient envoyées. Le fichier `options.config` contient les valeurs utilisées pour certaines des variables définies dans `dynamodb.config`. Par exemple, `dynamodb.config` contient les lignes suivantes :

```
Subscription:
  - Endpoint:
      Fn::GetOptionSetting:
        OptionName: SessionAlarmEmail
        DefaultValue: "nobody@amazon.com"
```

Ces lignes indiquent à Elastic Beanstalk d'obtenir la valeur de la propriété Endpoint à SessionAlarmEmailpartir de la valeur d'un **`options.config`** fichier de configuration (dans notre exemple d'application) qui contient une section option_settings avec une section aws:elasticbeanstalk:customoption contenant une paire nom-valeur contenant la valeur réelle à utiliser. Dans l'exemple ci-dessus, la valeur SessionAlarmEmailserait affectée à ce moyen`nobody@amazon.com`.

Pour plus d'informations sur les CloudFormation ressources utilisées dans cet exemple, consultez les références suivantes :

- AWS::DynamoDB::Table
- AWS::CloudWatch::Alarm
- AWS::SNS::Topic

Utilisation des configurations enregistrées par Elastic Beanstalk

Vous pouvez enregistrer la configuration de votre environnement en tant qu'objet dans Amazon Simple Storage Service (Amazon S3) pouvant être appliqué dans d'autres environnements au cours de la création d'environnement, ou appliqués à un environnement en cours d'exécution. Les configurations enregistrées sont des modèles au format YAML qui définissent la version de plateforme, le niveau, les paramètres d'option de configuration et les balises.

Vous pouvez appliquer les balises à une configuration enregistrée lorsque vous la créez, et modifier des balises existantes de configurations enregistrées. Les balises appliquées à une configuration

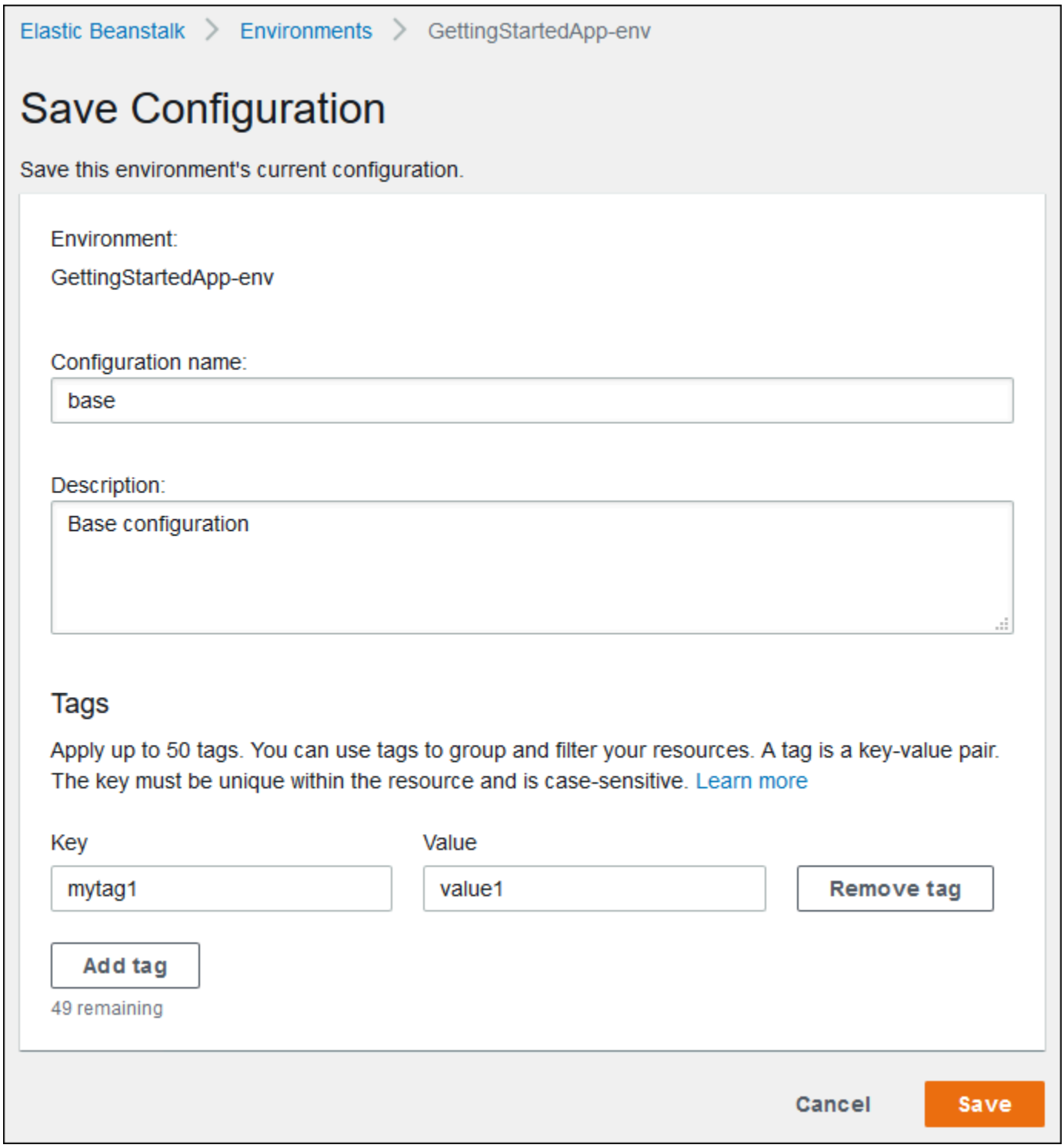
enregistrée ne sont pas liées à des balises spécifiées dans une configuration enregistrée à l'aide de la clé `Tags:`. Ces dernières sont appliquées à un environnement lorsque vous appliquez la configuration enregistrée à l'environnement. Pour en savoir plus, consultez Balisage de configurations enregistrées.

> **Note**
>
> Vous pouvez créer et appliquer des configurations enregistrées à vos environnements Elastic Beanstalk de plusieurs manières. Il s'agit notamment de la console Elastic Beanstalk, de l'interface de ligne EB et AWS CLI.
> Consultez les rubriques suivantes pour obtenir des exemples d'autres méthodes de création et d'application de configurations enregistrées :
>
> - Définition d'options de configuration avant la création de l'environnement
> - Définition des options de configuration lors de la création de l'environnement
> - Définition des options de configuration après la création de l'environnement

Créez une configuration enregistrée à partir de l'état actuel de votre environnement dans la console de gestion Elastic Beanstalk.

Pour enregistrer la configuration d'un environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Save configuration (Enregistrer la configuration).
4. Utilisez le formulaire à l'écran pour nommer la configuration enregistrée. Le cas échéant, fournissez une brève description, et ajouter les clés et valeurs de balise.
5. Choisissez Enregistrer.



La configuration enregistrée inclut tous les paramètres que vous avez appliqués à l'environnement avec la console ou tout autre client qui utilise l'API Elastic Beanstalk. Vous pouvez ensuite appliquer la configuration enregistrée à votre environnement à une date ultérieure pour la restaurer à son état antérieur, ou l'appliquer à un nouvel environnement pendant la création de l'environnement.

Vous pouvez télécharger une configuration à l'aide de la the section called “eb config” commande EB CLI, comme illustré dans l'exemple suivant. *NAME*est le nom de votre configuration enregistrée.

```
eb config get NAME
```

Pour appliquer une configuration enregistrée lors de la création de l'environnement (console Elastic Beanstalk)

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.
3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Saved configurations (Configurations enregistrées).
4. Sélectionnez la configuration enregistrée que vous souhaitez appliquer, puis choisissez Launch environment (Lancer l'environnement).
5. Suivez les étapes de l'assistant pour créer votre environnement.

Les configurations enregistrées ne comprennent pas les paramètres appliqués avec les fichiers de configuration dans le code source de votre application. Si le même paramètre est appliqué à la fois dans un fichier de configuration et dans une configuration enregistrée, le paramètre dans la configuration enregistrée est prioritaire. De même, les options spécifiées dans la console Elastic Beanstalk remplacent les options dans les configurations enregistrées. Pour de plus amples informations, veuillez consulter Priorité.

Les configurations enregistrées sont stockées dans le compartiment S3 Elastic Beanstalk, dans un dossier nommé en fonction de votre application. Par exemple, les configurations pour une application nommée `my-app` dans la région us-west-2 pour le compte n° 123456789012 sont disponibles à l'emplacement `s3://elasticbeanstalk-us-west-2-123456789012/resources/templates/my-app/`.

Affichez le contenu d'une configuration enregistrée en l'ouvrant dans un éditeur de texte. L'exemple de configuration suivant montre la configuration d'un environnement de serveur web lancé avec la console de gestion Elastic Beanstalk.

```
EnvironmentConfigurationMetadata:
```

```
  Description: Saved configuration from a multicontainer Docker environment created
 with the Elastic Beanstalk Management Console
  DateCreated: '1520633151000'
  DateModified: '1520633151000'
Platform:
  PlatformArn: arn:aws:elasticbeanstalk:us-east-2::platform/Java 8 running on 64bit
 Amazon Linux/2.5.0
OptionSettings:
  aws:elasticbeanstalk:command:
    BatchSize: '30'
    BatchSizeType: Percentage
  aws:elasticbeanstalk:sns:topics:
    Notification Endpoint: me@example.com
  aws:elb:policies:
    ConnectionDrainingEnabled: true
    ConnectionDrainingTimeout: '20'
  aws:elb:loadbalancer:
    CrossZone: true
  aws:elasticbeanstalk:environment:
    ServiceRole: aws-elasticbeanstalk-service-role
  aws:elasticbeanstalk:application:
    Application Healthcheck URL: /
  aws:elasticbeanstalk:healthreporting:system:
    SystemType: enhanced
  aws:autoscaling:launchconfiguration:
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role
    InstanceType: t2.micro
    EC2KeyName: workstation-uswest2
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateType: Health
    RollingUpdateEnabled: true
EnvironmentTier:
  Type: Standard
  Name: WebServer
AWSConfigurationTemplateVersion: 1.1.0.0
Tags:
  Cost Center: WebApp Dev
```

Vous pouvez modifier le contenu d'une configuration enregistrée et l'enregistrer au même emplacement dans Amazon S3. Toute configuration enregistrée avec le bon format stockée à l'emplacement approprié peut être appliquée à un environnement avec la console de gestion Elastic Beanstalk.

Les clés suivantes sont prises en charge.

- AWSConfigurationTemplateVersion(obligatoire) — Version du modèle de configuration (1.1.0.0).

```
AWSConfigurationTemplateVersion: 1.1.0.0
```

- Plateforme – Amazon Resource Name (ARN) de la version de plateforme de l'environnement. Vous pouvez spécifier la plateforme par ARN ou par nom de pile de solutions.

```
Platform:
  PlatformArn: arn:aws:elasticbeanstalk:us-east-2::platform/Java 8 running on 64bit
 Amazon Linux/2.5.0
```

- SolutionStack— Le nom complet de la pile de solutions utilisée pour créer l'environnement.

```
SolutionStack: 64bit Amazon Linux 2017.03 v2.5.0 running Java 8
```

- OptionSettings— Paramètres des options de configuration à appliquer à l'environnement. Par exemple, l'entrée suivante définit le type d'instance sur t2.micro.

```
OptionSettings:
  aws:autoscaling:launchconfiguration:
    InstanceType: t2.micro
```

- Balises – Jusqu'à 47 balises à appliquer aux ressources créées dans l'environnement.

```
Tags:
  Cost Center: WebApp Dev
```

- EnvironmentTier— Le type d'environnement à créer. Pour un environnement de serveur web, vous pouvez exclure cette section (le serveur web est la valeur par défaut). Pour un environnement de travail, utilisez les informations suivantes.

```
EnvironmentTier:
  Name: Worker
  Type: SQS/HTTP
```

> **Note**
>
> Vous pouvez créer et appliquer des configurations enregistrées à vos environnements Elastic Beanstalk de plusieurs manières. Il s'agit notamment de la console Elastic Beanstalk, de l'interface de ligne EB et AWS CLI.
> Consultez les rubriques suivantes pour obtenir des exemples d'autres méthodes de création et d'application de configurations enregistrées :
>
> - Définition d'options de configuration avant la création de l'environnement
> - Définition des options de configuration lors de la création de l'environnement
> - Définition des options de configuration après la création de l'environnement

Balisage de configurations enregistrées

Vous pouvez appliquer des balises à vos configurations AWS Elastic Beanstalk enregistrées. Les balises sont des paires clé-valeur associées à des AWS ressources. Pour plus d'informations sur le balisage des ressources Elastic Beanstalk, les cas d'utilisation, les contraintes de clé et de valeur de balise, et les types de ressources pris en charge, consultez Étiquette des ressources d'application Elastic Beanstalk.

Vous pouvez spécifier des balises lorsque vous créez une configuration enregistrée. Dans une configuration enregistrée, vous pouvez ajouter ou supprimer des balises, ainsi que mettre à jour les valeurs des balises existantes. Vous pouvez ajouter jusqu'à 50 balises à chaque configuration enregistrée.

Ajout de balises lors de la création d'une configuration enregistrée

Lorsque vous utilisez la console Elastic Beanstalk pour enregistrer une configuration, vous pouvez spécifier des clés et des valeurs de balise sur la page Save Configuration (Enregistrer la configuration).

Si vous utilisez l'interface de ligne de commande EB pour enregistrer une configuration, utilisez l'option `--tags` avec eb config pour ajouter des balises.

```
~/workspace/my-app$ eb config --tags mytag1=value1,mytag2=value2
```

Avec le AWS CLI ou d'autres clients basés sur une API, ajoutez des balises en utilisant le `--tags` paramètre de la commande. create-configuration-template

```
$ aws elasticbeanstalk create-configuration-template \
      --tags Key=mytag1,Value=value1 Key=mytag2,Value=value2 \
      --application-name my-app --template-name my-template --solution-stack-
name solution-stack
```

Gestion des balises d'une configuration enregistrée existante

Vous pouvez ajouter, mettre à jour et supprimer des balises dans une configuration Elastic Beanstalk enregistrée existante.

Pour gérer les balises d'une configuration enregistrée à l'aide de la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.
3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Saved configurations (Configurations enregistrées).
4. Sélectionnez la configuration enregistrée que vous souhaitez gérer.
5. Choisissez Actions, puis Manage Tags (Gérer les identifications).
6. Utilisez le formulaire à l'écran pour ajouter, mettre à jour ou supprimer des identifications.
7. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Si vous utilisez l'interface de ligne de commande EB pour mettre à jour votre configuration enregistrée, utilisez eb tags pour ajouter, mettre à jour, supprimer ou répertorier des balises.

Par exemple, la commande suivante répertorie les balises dans une configuration enregistrée.

```
~/workspace/my-app$ eb tags --list --resource "arn:aws:elasticbeanstalk:us-east-2:my-
account-id:configurationtemplate/my-app/my-template"
```

La commande suivante met à jour l'identification `mytag1` et supprime l'identification `mytag2`.

```
~/workspace/my-app$ eb tags --update mytag1=newvalue --delete mytag2 \
      --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-
id:configurationtemplate/my-app/my-template"
```

Pour obtenir une liste complète des options et d'autres exemples, consultez `eb tags`.

Avec le AWS CLI ou d'autres clients basés sur une API, utilisez la list-tags-for-resource commande pour répertorier les balises d'une configuration enregistrée.

```
$ aws elasticbeanstalk list-tags-for-resource --resource-arn
 "arn:aws:elasticbeanstalk:us-east-2:my-account-id:configurationtemplate/my-app/my-
template"
```

Utilisez la commande update-tags-for-resource pour ajouter, mettre à jour ou supprimer des balises dans une configuration enregistrée.

```
$ aws elasticbeanstalk update-tags-for-resource \
      --tags-to-add Key=mytag1,Value=newvalue --tags-to-remove mytag2 \
      --resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-
id:configurationtemplate/my-app/my-template"
```

Spécifiez les identifications à ajouter et les identifications à mettre à jour dans le paramètre `--tags-to-add` de update-tags-for-resource. Une identification inexistante est ajoutée et la valeur d'une identification existante est mise à jour.

> **Note**
>
> Pour utiliser certaines AWS CLI commandes et certaines de l'interface de ligne de commande EB avec une configuration enregistrée d'Elastic Beanstalk, vous avez besoin de l'ARN de la configuration enregistrée. Pour construire les ARN, vous devez d'abord utiliser la commande suivante pour récupérer la configuration enregistrée.
>
> ```
> $ aws elasticbeanstalk describe-applications --application-names my-app
> ```
>
> Recherchez la clé `ConfigurationTemplates` dans la sortie de la commande. Cet élément affiche le nom de la configuration enregistrée. Utilisez ce nom où *my-template* est spécifié dans les commandes mentionnées sur cette page.

Manifeste d'environnement (**env.yaml**)

Vous pouvez inclure un manifeste d'environnement au format YAML à la racine du groupe source de votre application pour configurer le nom de l'environnement, la pile de solutions et les liens d'environnement à utiliser lors de la création de votre environnement.

Ce format de fichier inclut la prise en charge pour les groupes de l'environnement. Pour utiliser des groupes, précisez le nom de l'environnement dans le manifeste avec un symbole + à la fin. Lorsque vous créez ou mettez à jour l'environnement, spécifiez le nom du groupe avec `--group-name` (AWS CLI) ou `--env-group-suffix` (interface de ligne de commande EB). Pour de plus amples informations sur les groupes, veuillez consulter Création et mise à jour de groupes d'environnements Elastic Beanstalk.

L'exemple de manifeste suivant définit un environnement de serveur web avec un lien vers un composant d'environnement de travail dont il dépend. Le manifeste utilise des groupes pour autoriser la création de plusieurs environnements avec le même groupe source :

`~/myapp/frontend/env.yaml`

```
AWSConfigurationTemplateVersion: 1.1.0.0
SolutionStack: 64bit Amazon Linux 2015.09 v2.0.6 running Multi-container Docker 1.7.1
 (Generic)
OptionSettings:
  aws:elasticbeanstalk:command:
    BatchSize: '30'
    BatchSizeType: Percentage
  aws:elasticbeanstalk:sns:topics:
    Notification Endpoint: me@example.com
  aws:elb:policies:
    ConnectionDrainingEnabled: true
    ConnectionDrainingTimeout: '20'
  aws:elb:loadbalancer:
    CrossZone: true
  aws:elasticbeanstalk:environment:
    ServiceRole: aws-elasticbeanstalk-service-role
  aws:elasticbeanstalk:application:
    Application Healthcheck URL: /
  aws:elasticbeanstalk:healthreporting:system:
    SystemType: enhanced
  aws:autoscaling:launchconfiguration:
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role
    InstanceType: t2.micro
    EC2KeyName: workstation-uswest2
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateType: Health
    RollingUpdateEnabled: true
Tags:
  Cost Center: WebApp Dev
```

```
CName: front-A08G28LG+
EnvironmentName: front+
EnvironmentLinks:
  "WORKERQUEUE" : "worker+"
```

Les clés suivantes sont prises en charge.

- AWSConfigurationTemplateVersion(obligatoire) — Version du modèle de configuration (1.1.0.0).

  ```
  AWSConfigurationTemplateVersion: 1.1.0.0
  ```

- Plateforme – Amazon Resource Name (ARN) de la version de plateforme de l'environnement. Vous pouvez spécifier la plateforme par ARN ou par nom de pile de solutions.

  ```
  Platform:
    PlatformArn: arn:aws:elasticbeanstalk:us-east-2::platform/Java 8 running on 64bit
   Amazon Linux/2.5.0
  ```

- SolutionStack— Le nom complet de la pile de solutions utilisée pour créer l'environnement.

  ```
  SolutionStack: 64bit Amazon Linux 2017.03 v2.5.0 running Java 8
  ```

- OptionSettings— Paramètres des options de configuration à appliquer à l'environnement. Par exemple, l'entrée suivante définit le type d'instance sur t2.micro.

  ```
  OptionSettings:
    aws:autoscaling:launchconfiguration:
      InstanceType: t2.micro
  ```

- Balises – Jusqu'à 47 balises à appliquer aux ressources créées dans l'environnement.

  ```
  Tags:
    Cost Center: WebApp Dev
  ```

- EnvironmentTier— Le type d'environnement à créer. Pour un environnement de serveur web, vous pouvez exclure cette section (le serveur web est la valeur par défaut). Pour un environnement de travail, utilisez les informations suivantes.

  ```
  EnvironmentTier:
    Name: Worker
    Type: SQS/HTTP
  ```

- CName— Le CNAME de l'environnement. Incluez un caractère + à la fin du nom pour activer des groupes.

  ```
  CName: front-A08G28LG+
  ```

- EnvironmentName— Le nom de l'environnement à créer. Incluez un caractère + à la fin du nom pour activer des groupes.

  ```
  EnvironmentName: front+
  ```

  Avec les groupes activés, vous devez spécifier un nom de groupe lorsque vous créez les environnements. Elastic Beanstalk ajoute le nom du groupe au nom de l'environnement avec un trait d'union. Par exemple, avec le nom d'environnement `front+` et le nom de groupe `dev`, Elastic Beanstalk crée l'environnement nommé `front-dev`.

- EnvironmentLinks— Une carte des noms de variables et des noms d'environnement des dépendances. L'exemple suivant fait de l'environnement `worker+` une dépendance et indique à Elastic Beanstalk d'enregistrer les informations de lien sur une variable nommée `WORKERQUEUE`.

  ```
  EnvironmentLinks:
    "WORKERQUEUE" : "worker+"
  ```

  La valeur de la variable de lien varie en fonction du type de l'environnement lié. Pour un environnement de serveur web, le lien est le CNAME de l'environnement. Pour un environnement de travail, le lien est le nom de la file d'attente Amazon Simple Queue Service (Amazon SQS) de l'environnement.

Les CNameEnvironmentLinkstouches EnvironmentNameet peuvent être utilisées pour créer des groupes d'environnements et des liens vers d'autres environnements. Ces fonctionnalités sont actuellement prises en charge lors de l'utilisation de l'interface de ligne de commande EB AWS CLI ou d'un SDK.

Utilisation d'une image machine Amazon (AMI) personnalisée dans votre environnement Elastic Beanstalk

Cette section explique dans quels cas envisager d'utiliser une AMI personnalisée et fournit les procédures de configuration et de gestion de l'AMI personnalisée dans votre environnement. Lorsque

vous créez un AWS Elastic Beanstalk environnement, vous pouvez spécifier une Amazon Machine Image (AMI) à utiliser à la place de l'AMI Elastic Beanstalk standard incluse dans la version de votre plateforme. Une AMI personnalisée peut améliorer les délais de provisionnement lorsque des instances sont lancées dans votre environnement si vous devez installer un grand nombre de logiciels non inclus dans la norme AMIs.

L'utilisation de fichiers de configuration est efficace pour personnaliser votre environnement rapidement et de manière cohérente. Cependant, l'application des configurations peut commencer à prendre beaucoup de temps lors de la création et des mises à jour de l'environnement. Si vous passez beaucoup de temps sur la configuration serveur dans les fichiers de configuration, vous pouvez résoudre le problème en créant une AMI personnalisée qui dispose déjà du logiciel et de la configuration dont vous avez besoin.

En outre, une AMI personnalisée vous permet d'apporter des modifications aux composants de bas niveau, tels que le noyau Linux, qui sont difficiles à mettre en œuvre et dont l'application dans les fichiers de configuration prend beaucoup de temps. Pour créer une AMI personnalisée, lancez une AMI de plateforme Elastic Beanstalk dans Amazon EC2, personnalisez le logiciel et la configuration selon vos besoins, puis arrêtez l'instance et enregistrez une AMI à partir de cette dernière.

Création d'une AMI personnalisée

Vous pouvez utiliser EC2 Image Builder pour créer et gérer des fichiers AMIs personnalisés comme alternative à ces procédures. Pour plus d'informations, consultez le guide de l'utilisateur d'Image Builder.

Pour identifier l'AMI Elastic Beanstalk de base

1. Dans une fenêtre de commande, exécutez une commande similaire à celle-ci. Pour plus d'informations, consultez describe-platform-version dans la Référence des commandes de l'AWS CLI .

   Spécifiez la AWS région dans laquelle vous souhaitez utiliser votre AMI personnalisée et remplacez l'ARN et le numéro de version de la plateforme par la plateforme Elastic Beanstalk sur laquelle repose votre application.

   Example- Système d'exploitation Mac / Linux

   ```
   $ aws elasticbeanstalk describe-platform-version --region us-east-2 \
   ```

```
      --platform-arn "arn:aws:elasticbeanstalk:us-east-2::platform/Node.js 20
 running on 64bit Amazon Linux 2023/6.1.7" \
      --query PlatformDescription.CustomAmiList
[
    {
        "VirtualizationType": "pv",
        "ImageId": ""
    },
    {
        "VirtualizationType": "hvm",
        "ImageId": "ami-020ae06fdda6a0f66"
    }
]
```

Example- Système d'exploitation Windows

```
C:\> aws elasticbeanstalk describe-platform-version --region us-east-2 --platform-
arn"arn:aws:elasticbeanstalk:us-east-2::platform/
IIS 10.0 running on 64bit Windows Server 2022/2.15.3" --query
 PlatformDescription.CustomAmiList
[
    {
        "VirtualizationType": "pv",
        "ImageId": ""
    },
    {
        "VirtualizationType": "hvm",
        "ImageId": "ami-020ae06fdda6a0f66"
    }
]
```

2. Notez la valeur `ImageId` qui ressemble à `ami-020ae06fdda6a0f66` dans le résultat.

La valeur est l'AMI Elastic Beanstalk standard correspondant à la version de plateforme, à l'architecture AWS de l'instance EC2 et à la région pertinentes pour votre application. Si vous devez créer AMIs pour plusieurs plateformes, architectures ou AWS régions, répétez ce processus pour identifier l'AMI de base appropriée pour chaque combinaison.

> **Note**
>
> Ne créez pas d'AMI à partir d'une instance qui a été lancée dans un environnement Elastic Beanstalk. Elastic Beanstalk apporte des modifications aux instances au cours de la mise en service, ce qui peut générer des problèmes dans l'AMI enregistrée. En outre, si vous enregistrez une image à partir d'une instance d'un environnement Elastic Beanstalk, la version de votre application ayant été déployée dans l'instance sera transformée en partie fixe de l'image.

Pour Linux, vous pouvez également créer une AMI personnalisée à partir d'une AMI de la communauté qui n'a pas été publiée par Elastic Beanstalk. Vous pouvez utiliser la dernière AMI Linux Amazon comme point de départ. Lorsque vous lancez un environnement avec une AMI Linux non gérée par Elastic Beanstalk, Elastic Beanstalk tente d'installer des logiciels de plateforme (langage, cadre, serveur proxy, etc.) et des composants supplémentaires pour prendre en charge des fonctionnalités telles que les rapports sur l'état amélioré.

> **Note**
>
> La personnalisation AMIs basée sur Windows Server nécessite le `describe-platform-version` retour de l'AMI Elastic Beanstalk d'origine, comme indiqué précédemment à l'étape 1.

Bien qu'Elastic Beanstalk puisse utiliser une AMI qui n'est pas gérée par Elastic Beanstalk, l'augmentation du temps de mise en service résultant de l'installation de composants manquants par Elastic Beanstalk peut réduire ou éliminer les avantages de la création d'une AMI personnalisée. D'autres distributions Linux peuvent fonctionner, avec des solutions de dépannage, mais elles ne sont pas officiellement prises en charge. Si votre application nécessite une distribution Linux spécifique, une alternative consiste à créer une image Docker et à l'exécuter sur la plateforme Docker ou la plateforme Docker multi-conteneurs d'Elastic Beanstalk.

Pour créer une AMI personnalisée

1. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
2. Choisissez Launch Instances (Lancer les instances).

3. Si vous avez identifié une AMI Elastic Beanstalk de base (en utilisant `describe-platform-version`) ou une AMI Amazon Linux, entrez son ID dans la zone de recherche. Appuyez ensuite sur Entrée.

   Vous pouvez également explorer la liste d'une autre AMI de communauté qui répond à vos besoins.

   > **Note**
   >
   > Nous vous recommandons de choisir une AMI qui utilise la virtualisation HVM. Ils AMIs indiquent le type de virtualisation : hvm dans leur description.
   > Pour plus d'informations, consultez la section Types de virtualisation dans le guide de l'utilisateur Amazon EC2.

4. Choisissez Sélectionner pour sélectionner l'AMI.
5. Sélectionnez un type d'instance, puis choisissez Suivant : Configurer les détails de l'instance.
6. (Pour les plateformes Amazon Linux AMI (AL1) retirées) Ignorez cette étape si votre environnement s'exécute sur une plate-forme Linux prise en charge ou sur une plate-forme Windows.

   Développez la section Détails avancés et collez le texte suivant dans le champ Données utilisateur.

   ```
   #cloud-config
     repo_releasever: repository version number
     repo_upgrade: none
   ```

   Le numéro de version du référentiel correspond à l'année et au mois de la version dans le nom de l'AMI. Par exemple, sur la AMIs base de la version de mars 2015 d'Amazon, Linux possède un numéro de version du référentiel`2015.03`. Pour une image Elastic Beanstalk, cela correspond à la date indiquée dans le nom de la pile de solution pour votre version de plateforme basée sur l'AMI Amazon Linux (antérieure à Amazon Linux 2).

   > **Note**
   >
   > Le `repo_releasever` paramètre configure la lock-on-launch fonctionnalité d'une AMI Amazon Linux. L'AMI utilise alors une version de référentiel fixe et spécifique lors de son lancement. Cette fonctionnalité n'est pas prise en charge sur Amazon Linux 2. Ne la

spécifiez pas si votre environnement utilise une branche de plateforme Amazon Linux 2 actuelle. Ce paramètre est requis si vous utilisez une AMI personnalisée avec Elastic Beanstalk uniquement sur les branches de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2).
Ce paramètre `repo_upgrade` désactive l'installation automatique des mises à jour de sécurité. Il est nécessaire d'utiliser une AMI personnalisée avec Elastic Beanstalk.

7. Suivez les étapes de l'assistant pour lancer l'instance EC2. Lorsque vous y êtes invité, sélectionnez une paire de clés à laquelle vous avez accès afin de pouvoir vous connecter à l'instance pour suivre les prochaines étapes.
8. Connectez-vous à l'instance avec SSH ou RDP.
9. Effectuez toutes les personnalisations de votre choix.
10. (Plateformes Windows) Exécutez le service EC2 Config Sysprep. Pour plus d'informations sur EC2 Config, consultez Configuration d'une instance Windows à l'aide du service EC2 Config. Assurez-vous que Sysprep est configuré pour générer un mot de passe aléatoire pouvant être extrait de AWS Management Console.
11. Dans la console Amazon EC2, arrêtez l'instance EC2. Ensuite, dans le menu Instance Actions (Actions d'instance), choisissez Créer l'image (EBS AMI).
12. Pour éviter d'encourir des AWS frais supplémentaires, mettez fin à l'instance EC2.

Pour utiliser votre AMI personnalisée dans un environnement Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Pour l'ID d’AMI, entrez votre ID d’AMI personnalisé.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Lorsque vous créez un nouvel environnement avec l'AMI personnalisée, vous devez utiliser la même version de plateforme que celle utilisée comme base pour créer l'AMI.

Gestion d'un environnement à l'aide d'une AMI personnalisée

Mises à jour de plateforme

Lors de l'utilisation d'une AMI personnalisée, Elastic Beanstalk continuera à utiliser la même AMI personnalisée dans un environnement lorsque la version de sa plate-forme est mise à jour, que la mise à jour soit appliquée manuellement ou via des mises à jour de plate-forme gérées. L'environnement ne sera pas réinitialisé pour utiliser l'AMI d'origine de la nouvelle version de la plateforme.

Nous vous recommandons de créer une nouvelle AMI personnalisée basée sur l'AMI standard de la nouvelle version de la plateforme. Cela permettra d'appliquer les correctifs disponibles dans la nouvelle version de la plateforme et de minimiser les échecs de déploiement dus à des versions de packages ou de bibliothèques incompatibles.

Pour plus d'informations sur la création d'une nouvelle AMI personnalisée, consultez la section Création d'une AMI personnalisée précédente de cette rubrique.

Supprimer une AMI personnalisée

Si vous souhaitez supprimer une AMI personnalisée d'un environnement et la réinitialiser pour qu'elle utilise l'AMI de base correspondant à la version de plate-forme de l'environnement, utilisez la commande CLI suivante.

```
aws elasticbeanstalk update-environment \
  --application-name my-application \
  --environment-name my-environment \
  --region us-east-1 \
  --options-to-remove Namespace=aws:autoscaling:launchconfiguration,OptionName=ImageId
```

> **Note**
>
> Pour éviter toute interruption de service, testez votre application avec une AMI standard avant d'appliquer cette modification à votre environnement de production.

Nettoyage d'une AMI personnalisée

Lorsque vous avez fini d'utiliser une AMI personnalisée et que vous n'en avez plus besoin pour lancer des environnements Elastic Beanstalk, nettoyez-la afin de réduire les coûts de stockage. Le

nettoyage d'une AMI personnalisée implique l'annulation de son enregistrement dans Amazon EC2 et la suppression des autres ressources associées. Pour de plus amples informations, veuillez consulter Annulation de l'inscription de votre AMI Linux ou Annuler l'inscription de votre AMI Windows.

Conservation de l'accès à une Amazon Machine Image (AMI) pour une plateforme retirée

Elastic Beanstalk définit le statut de la branche de plateforme sur retiré lorsque le système d'exploitation ou le composant principal utilisé par la branche atteint sa fin de vie. L'AMI Elastic Beanstalk de base pour la branche de plateforme peut également être rendue privée afin d'empêcher l'utilisation de cette AMI. out-of-date Les environnements AMIs qui ont été rendus privés ne pourront plus lancer d'instances.

Si vous ne parvenez pas à migrer votre application vers un environnement pris en charge avant son retrait, votre environnement peut être concerné. Il peut être nécessaire de mettre à jour un environnement pour une branche de plateforme Beanstalk, dont l'AMI Elastic Beanstalk de base a été rendue privée. Une autre approche est possible. Vous pouvez mettre à jour un environnement existant basé sur une copie de l'AMI Elastic Beanstalk de base utilisée par votre environnement.

Cette rubrique fournit une procédure et un script autonome pour mettre à jour un environnement existant basé sur une copie de l'AMI Elastic Beanstalk de base utilisée par votre environnement. Après avoir migré votre application vers une plateforme prise en charge, vous pouvez continuer à utiliser les procédures standard de maintenance de votre application et des environnements pris en charge.

Étapes manuelles

Pour mettre à jour un environnement basé sur une copie de l'AMI Elastic Beanstalk de base

1. Déterminez l'AMI utilisée par votre environnement. Cette commande renvoie l'AMI utilisée par l'environnement Elastic Beanstalk que vous spécifiez dans les paramètres. La valeur renvoyée est utilisée comme source-ami-idà l'étape suivante.

   Dans une fenêtre de commande, exécutez une commande similaire à celle-ci. Pour plus d’informations, consultez describe-configuration-settings dans la Référence des commandes de l’AWS CLI .

Spécifiez la AWS région qui stocke l'AMI source que vous souhaitez copier. Remplacez le nom de l'application et le nom de l'environnement par ceux basés sur l'AMI source. Entrez le texte du paramètre de requête comme indiqué.

Example

```
>aws elasticbeanstalk describe-configuration-settings \
  --application-name my-application \
  --environment-name my-environment \
  --region us-east-2 \
  --query "ConfigurationSettings[0].OptionSettings[?OptionName=='ImageId'] |
 [0].Value"
```

2. Copiez l'AMI dans votre compte. Cette commande renvoie la nouvelle AMI qui résulte de la copie de source-ami-idcelle renvoyée à l'étape précédente.

> **Note**
>
> Veillez à noter le nouvel identifiant d'AMI généré par cette commande. Vous devrez le saisir à l'étape suivante, en le remplaçant copied-ami-iddans l'exemple de commande.

Dans une fenêtre de commande, exécutez une commande similaire à celle-ci. Pour plus d'informations, voir la rubrique copy-object dans la Référence des commandes AWS CLI .

Spécifiez la AWS région de l'AMI source que vous souhaitez copier (--source-region) et la région dans laquelle vous souhaitez utiliser votre nouvelle AMI personnalisée (--region). source-ami-id Remplacez-le par l'AMI de l'image que vous copiez. Le source-ami-ida été renvoyé par la commande à l'étape précédente. new-ami-nameRemplacez-le par un nom décrivant la nouvelle AMI dans la région de destination. Le script qui suit cette procédure génère le nouveau nom de l'AMI en ajoutant la chaîne « Copy of » au début du nom dusource-ami-id.

```
>aws ec2 copy-image \
    --region us-east-2 \
    --source-image-id source-ami-id \
    --source-region us-east-2 \
    --name new-ami-name
```

3. Mettez à jour un environnement pour utiliser l'AMI copiée. Une fois exécutée, la commande renvoie le statut de l'environnement.

   Dans une fenêtre de commande, exécutez une commande similaire à celle-ci. Pour plus d'informations, voir la rubrique update-environment dans la Référence des commandes AWS CLI .

   Spécifiez la AWS région de l'environnement et de l'application que vous devez mettre à jour. Remplacez le nom de l'application et le nom de l'environnement par ceux que vous devez associer copied-ami-idà l'étape précédente. Pour le paramètre --option-setttings, remplacez-le par l'identifiant de *copied-ami-id* l'AMI que vous avez noté dans le résultat de la commande précédente.

   ```
   >aws elasticbeanstalk update-environment \
     --application-name my-application \
     --environment-name my-environment \
     --region us-east-2 \
     --option-settings
    "Namespace=aws:autoscaling:launchconfiguration,OptionName=ImageId,Value=copied-ami-id"
   ```

> **Note**
>
> Nettoyez votre AMI personnalisée lorsque vous n'en avez plus besoin pour lancer des environnements Elastic Beanstalk afin de limiter les coûts de stockage. Pour plus d'informations, voir Nettoyage d'une AMI personnalisée.

Script autonome

Le script suivant fournit les mêmes résultats que les étapes manuelles précédentes. Téléchargez le script en cliquant sur le lien suivant : copy_ami_and_update_env.zip.

Source du script : copy_ami_and_update_env.sh

```
#!/bin/bash

set -ue
```

```
USAGE="This script is used to copy an AMI used by your Elastic Beanstalk environment
 into your account to use in your environment.\n\n"
USAGE+="Usage:\n\n"
USAGE+="./$(basename $0) [OPTIONS]\n"
USAGE+="OPTIONS:\n"
USAGE+="\t--application-name <application-name>\tThe name of your Elastic Beanstalk
 application.\n"
USAGE+="\t--environment-name <environment-name>\tThe name of your Elastic Beanstalk
 environment.\n"
USAGE+="\t--region <region> \t\t\tThe AWS region your Elastic Beanstalk environment is
 deployed to.\n"
USAGE+="\n\n"
USAGE+="Script Usage Example(s):\n"
USAGE+="./$(basename $0) --application-name my-application --environment-name my-
environment --region us-east-1\n"

if [ $# -eq 0 ]; then
  echo -e $USAGE
  exit
fi

while [[ $# -gt 0 ]]; do
  case $1 in
    --application-name)       APPLICATION_NAME="$2"; shift ;;
    --environment-name)       ENVIRONMENT_NAME="$2"; shift ;;
    --region)                 REGION="$2"; shift ;;
    *)                        echo "Unknown option $1" ; echo -e $USAGE ; exit ;;
  esac
  shift
done

aws_cli_version="$(aws --version)"
if [ $? -ne 0 ]; then
  echo "aws CLI not found. Please install it: https://docs.aws.amazon.com/cli/latest/
userguide/getting-started-install.html. Exiting."
  exit 1
fi
echo "Using aws CLI version: ${aws_cli_version}"

account=$(aws sts get-caller-identity --query "Account" --output text)
echo "Using account ${account}"

environment_ami_id=$(aws elasticbeanstalk describe-configuration-settings \
  --application-name "$APPLICATION_NAME" \
```

```
  --environment-name "$ENVIRONMENT_NAME" \
  --region "$REGION" \
  --query "ConfigurationSettings[0].OptionSettings[?OptionName=='ImageId'] | [0].Value"
 \
  --output text)
echo "Image associated with environment ${ENVIRONMENT_NAME} is ${environment_ami_id}"

owned_image=$(aws ec2 describe-images \
  --owners self \
  --image-ids "$environment_ami_id" \
  --region "$REGION" \
  --query "Images[0]" \
  --output text)
if [ "$owned_image" != "None" ]; then
  echo "${environment_ami_id} is already owned by account ${account}. Exiting."
  exit
fi

source_image_name=$(aws ec2 describe-images \
  --image-ids "$environment_ami_id" \
  --region "$REGION" \
  --query "Images[0].Name" \
  --output text)
if [ "$source_image_name" = "None" ]; then
  echo "Cannot find ${environment_ami_id}. Please contact AWS support if you need
 additional help: https://aws.amazon.com/support."
  exit 1
fi

copied_image_name="Copy of ${source_image_name}"
copied_ami_id=$(aws ec2 describe-images \
  --owners self \
  --filters Name=name,Values="${copied_image_name}" \
  --region "$REGION" \
  --query "Images[0].ImageId" \
  --output text)
if [ "$copied_ami_id" != "None" ]; then
  echo "Detected that ${environment_ami_id} has already been copied by account
 ${account}. Skipping image copy."
else
  echo "Copying ${environment_ami_id} to account ${account} with name
 ${copied_image_name}"
  copied_ami_id=$(aws ec2 copy-image \
    --source-image-id "$environment_ami_id" \
```

```
    --source-region "$REGION" \
    --name "$copied_image_name" \
    --region "$REGION" \
    --query "ImageId" \
    --output text)
  echo "New AMI ID is ${copied_ami_id}"

  echo "Waiting for ${copied_ami_id} to become available"
  aws ec2 wait image-available \
    --image-ids "$copied_ami_id" \
    --region "$REGION"
  echo "${copied_ami_id} is now available"
fi

echo "Updating environment ${ENVIRONMENT_NAME} to use ${copied_ami_id}"
environment_status=$(aws elasticbeanstalk update-environment \
  --application-name "$APPLICATION_NAME" \
  --environment-name "$ENVIRONMENT_NAME" \
  --option-settings
 "Namespace=aws:autoscaling:launchconfiguration,OptionName=ImageId,Value=
${copied_ami_id}" \
  --region "$REGION" \
  --query "Status" \
  --output text)
echo "Environment ${ENVIRONMENT_NAME} is now ${environment_status}"

echo "Waiting for environment ${ENVIRONMENT_NAME} update to complete"
aws elasticbeanstalk wait environment-updated \
  --application-name "$APPLICATION_NAME" \
  --environment-names "$ENVIRONMENT_NAME" \
  --region "$REGION"
echo "Environment ${ENVIRONMENT_NAME} update complete"
```

Note

Vous devez l'avoir AWS CLI installé pour exécuter le script. Pour obtenir des instructions d'installation, voir la rubrique Installation ou mise à jour de la dernière version de la AWS CLI du Guide de l'utilisateur de la AWS Command Line Interface .
Après avoir installé le AWS CLI, vous devez également le configurer pour utiliser le AWS compte propriétaire de l'environnement. Pour plus d'informations, voir la rubrique Configuration de la AWS CLI du Guide de l'utilisateur de la AWS Command Line Interface

. Le compte doit également disposer des autorisations nécessaires pour créer une AMI et mettre à jour l'environnement Elastic Beanstalk.

Ces étapes décrivent le processus suivi par le script.

1. Notez le compte en cours d'utilisation.
2. Déterminez quelle AMI est utilisée par l'environnement (AMI source).
3. Vérifiez si l'AMI source appartient déjà au compte. Si oui, quittez.
4. Déterminez le nom de l'AMI source afin qu'il puisse être utilisé dans le nom de la nouvelle AMI. Cela permet également de confirmer l'accès à l'AMI source.
5. Vérifiez si l'AMI source a déjà été copiée sur le compte. Cela se fait en recherchant AMIs le nom de l'AMI copiée appartenant au compte. Si le nom de l'AMI a été modifié entre deux exécutions de script, l'image sera de nouveau copiée.
6. Si l'AMI source n'a pas encore été copiée, copiez-la sur le compte et patientez jusqu'à ce que la nouvelle AMI soit disponible.
7. Mettez à jour la configuration de l'environnement de sorte à utiliser la nouvelle AMI.
8. Patientez jusqu'à la fin de la mise à jour de l'environnement.

Après avoir extrait le script du fichier copy_ami_and_update_env.zip, exécutez-le comme indiqué dans l'exemple suivant. Remplacez le nom de l'application et le nom de l'environnement de l'exemple par vos propres valeurs.

```
>sh copy_ami_and_update_env.sh \
  --application-name my-application \
  --environment-name my-environment \
  --region us-east-1
```

> **Note**
>
> Nettoyez votre AMI personnalisée lorsque vous n'en avez plus besoin pour lancer des environnements Elastic Beanstalk afin de limiter les coûts de stockage. Pour plus d'informations, voir Nettoyage d'une AMI personnalisée.

Service de fichiers statiques

Pour améliorer les performances, vous pouvez configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Elastic Beanstalk prend en charge la configuration du proxy pour servir des fichiers statiques sur la plupart des branches de plate-forme basées sur Amazon Linux 2. La seule exception est Docker.

> **Note**
>
> Sur les plates-formes Python et Ruby, Elastic Beanstalk configure certains dossiers de fichiers statiques par défaut. Pour plus de détails, consultez les sections de configuration de fichiers statiques pour Python et Ruby. Vous pouvez configurer des dossiers supplémentaires comme expliqué sur cette page.

Configurer les fichiers statiques à l'aide de la console

Pour configurer le serveur proxy afin de servir des fichiers statiques

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Faites défiler la page jusqu'à la section Logiciel de plateforme et localisez le groupe Fichiers statiques.

    a. Pour ajouter un mappage de fichiers statiques, sélectionnez Ajouter des fichiers statiques. Dans la ligne supplémentaire qui s'affiche, vous devez saisir un chemin pour servir les fichiers statiques et le répertoire qui contient les fichiers statiques à servir.

- Dans le champ Chemin, commencez le nom du chemin par une barre oblique (/) (par exemple, « /images »).
- Dans le champ Répertoire, spécifiez un nom de répertoire situé à la racine du code source de votre application. Ne le faites pas commencer par une barre oblique (par exemple, « static/image-files »).

> **Note**
>
> Si la section Static Files (fichiers statiques) n'apparaît pas, vous devez ajouter au moins un mappage à l'aide d'un fichier de configuration. Pour plus de détails, consultez the section called “Configurer des fichiers statiques à l'aide des options de configuration” sur cette page.

b. Pour supprimer un mappage, sélectionnez Supprimer.

6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Configurer des fichiers statiques à l'aide des options de configuration

Vous pouvez utiliser un fichier de configuration pour configurer des chemins de fichiers statiques et des emplacements de répertoires à l'aide des options de configuration. Vous pouvez ajouter un fichier de configuration au bundle source de votre application et le déployer lors de la création de l'environnement ou d'un déploiement ultérieur.

Si votre environnement utilise une branche de plateforme basée sur Amazon Linux 2, utilisez l'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles`.

L'exemple de fichier de configuration suivant indique au serveur proxy de servir les fichiers du dossier `statichtml` sur le chemin `/html` et les fichiers du dossier `staticimages` sur le chemin `/images`.

Example.ebextensions/static-files.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:staticfiles:
    /html: statichtml
    /images: staticimages
```

Si votre environnement Elastic Beanstalk utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires suivantes :

Espaces de noms spécifiques à la plateforme AMI Amazon Linux

Sur les branches de la plateforme AMI Amazon Linux, les espaces de noms de configuration de fichiers statiques varient selon la plateforme. Pour plus d'informations, consultez l'une des pages suivantes :

- Espaces de noms de la configuration Go
- Espaces de noms de la configuration Java SE
- Espaces de noms de la configuration Tomcat
- Espaces de noms de la configuration Node.js
- Espaces de noms de la configuration Python

Configuration de HTTPS pour votre environnement Elastic Beanstalk

Les rubriques de cette section expliquent comment configurer le protocole HTTPS pour votre environnement Elastic Beanstalk. HTTPS est une nécessité pour n'importe quelle application qui transmet des informations de connexion ou des données utilisateur.

Si vous avez acheté et configuré un nom de domaine personnalisé pour votre environnement Elastic Beanstalk, vous pouvez utiliser HTTPS pour permettre aux utilisateurs de se connecter à votre site web en toute sécurité.

Si vous ne possédez pas un nom de domaine, vous pouvez toujours utiliser HTTPS avec un certificat auto-signé à des fins de développement et de test. Pour de plus amples informations, veuillez consulter Certificats de serveur.

Configuration de la terminaison HTTPS au niveau de l'équilibreur de charge

Un équilibreur de charge distribue les demandes aux instances EC2 qui exécutent votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet. Le moyen le plus simple d'utiliser le protocole HTTPS dans un environnement multi-instance Elastic Beanstalk consiste à configurer un écouteur sécurisé pour l'équilibreur de charge. La connexion entre le client et l'équilibreur de charge reste sécurisée. Vous pouvez donc configurer l'équilibreur

de charge pour mettre fin au protocole HTTPS. Les connexions principales entre l'équilibreur de charge et les instances EC2 utilisent le protocole HTTP. Aucune configuration supplémentaire des instances n'est donc requise. Pour obtenir des instructions détaillées sur la configuration d'un écouteur sécurisé, consultezConfiguration de la terminaison HTTPS au niveau de l'équilibreur de charge.

Configuration de la terminaison HTTPS sur l'instance EC2

Si vous exécutez votre application dans un environnement d'instance unique, ou que vous avez besoin de sécuriser la connexion intégralement vers les instances EC2 derrière l'équilibreur de charge, vous pouvez configurer le serveur proxy qui s'exécute sur l'instance pour résilier HTTPS. La configuration de vos instances pour résilier des connexions HTTPS nécessite l'utilisation de fichiers de configuration pour modifier le logiciel s'exécutant sur les instances et pour modifier les groupes de sécurité pour autoriser des connexions sécurisées. Pour de plus amples informations, veuillez consulter Configuration de la terminaison HTTPS sur l'instance.

Configuration du protocole HTTPS end-to-end

Pour le end-to-end protocole HTTPS dans un environnement d'équilibrage de charge, vous pouvez combiner la terminaison de l'instance et de l'équilibreur de charge pour chiffrer les deux connexions. Par défaut, si vous configurez l'équilibreur de charge pour acheminer le trafic à l'aide de HTTPS, il va faire confiance à n'importe quel certificat qui lui est présenté par les instances backend. Pour une sécurité optimale, vous pouvez attacher des stratégies à l'équilibreur de charge qui l'empêchent de se connecter aux instances qui ne présentent pas un certificat public auquel il fait confiance. Pour de plus amples informations, veuillez consulter Configuration du end-to-end chiffrement dans un environnement Elastic Beanstalk à charge équilibrée.

Configuration du protocole HTTPS avec TCP Passthrough

Vous pouvez également configurer l'équilibreur de charge pour relayer le trafic HTTPS sans le déchiffrer. Pour de plus amples informations, veuillez consulter Configuration de l'équilibreur de charge de votre environnement pour TCP Passthrough.

> **Note**
>
> Est-ce qu'il y a des serpents ? Un exemple d'application GitHub inclut des fichiers de configuration et des instructions pour chaque méthode de configuration du protocole HTTPS avec une application Web Tomcat. Consultez le fichier readme et les instructions HTTPS pour plus de détails.

Rubriques

- Certificats de serveur
- Configuration de la terminaison HTTPS au niveau de l'équilibreur de charge
- Configuration de la terminaison HTTPS sur l'instance
- Configuration du end-to-end chiffrement dans un environnement Elastic Beanstalk à charge équilibrée
- Configuration de l'équilibreur de charge de votre environnement pour TCP Passthrough
- Configuration de la redirection HTTP vers HTTPS

Certificats de serveur

Cette rubrique décrit les différents types de certificats que vous pouvez utiliser pour configurer le protocole HTTPS et les circonstances dans lesquelles ils doivent être appliqués. Les sous-rubriques de cette section fournissent des instructions pour créer votre propre certificat et comment le télécharger.

AWS Certificate Manager (ACM)

ACM est l'outil préféré pour mettre en service, gérer et déployer vos certificats de serveur. Vous pouvez le faire par programmation ou à l'aide du. AWS CLI Avec ACM, vous pouvez créer gratuitement un certificat fiable pour vos noms de domaine.

Les certificats ACM ne peuvent être utilisés qu'avec les équilibreurs de AWS charge et les CloudFront distributions Amazon, et ACM n'est disponible que dans certaines régions. AWS Pour utiliser un certificat ACM avec Elastic Beanstalk, veuillez consulter Configuration de la terminaison HTTPS au niveau de l'équilibreur de charge. Pour plus d'informations sur ACM, consultez le guide de l'AWS Certificate Manager utilisateur.

> **Note**
>
> Pour obtenir la liste des régions dans lesquelles ACM est disponible, consultez la section Points de terminaison et quotas ACM dans le. Référence générale d'Amazon Web Services

Si ACM n'est pas disponible dans votre AWS région, vous pouvez télécharger un certificat tiers ou auto-signé et une clé privée vers Gestion des identités et des accès AWS (IAM). Vous pouvez utiliser

le AWS CLI pour télécharger le certificat. Les certificats stockés dans IAM peuvent être utilisés avec les équilibreurs de charge et CloudFront les distributions. Pour de plus amples informations, veuillez consulter Chargement d'un certificat dans IAM.

Certificat d'une tierce partie

Si ACM n'est pas disponible dans votre région, vous pouvez acheter un certificat de confiance auprès d'un tiers. Un certificat tiers peut être utilisé pour déchiffrer le trafic HTTPS au niveau de votre équilibreur de charge, sur les instances backend, ou les deux.

Certificat auto-signé

Pour le développement et les tests, vous pouvez créer et signer un certificat vous-même avec des outils open source. Les certificats auto-signés sont gratuits et faciles à créer, mais ne peuvent pas être utilisés pour le déchiffrement frontal sur les sites publics. Si vous essayez d'utiliser un certificat auto-signé pour une connexion HTTPS à un client, le navigateur de l'utilisateur affiche une erreur indiquant que votre site web n'est pas sécurisé. Vous pouvez, toutefois, utiliser un certificat auto-signé pour sécuriser les connexions backend sans problème.

Création et signature d'un certificat X509

Vous pouvez créer un certificat X509 pour votre application avec `OpenSSL`. OpenSSL est une bibliothèque open source standard qui prend en charge un large éventail de fonctions de chiffrement, dont la création et la signature de certificats X509. Pour plus d'informations sur OpenSSL, consultez le site http://www.openssl.org.

> **Note**
>
> Vous ne devez créer un certificat localement que si vous souhaitez utiliser HTTPS dans un environnement à instance unique ou effectuer un nouveau chiffrement sur le backend avec un certificat auto-signé. Si vous possédez un nom de domaine, vous pouvez créer un certificat AWS et l'utiliser gratuitement dans un environnement à charge équilibrée en utilisant AWS Certificate Manager (ACM). Pour obtenir les instructions, consultez Demande de certificat dans le Guide de l'utilisateur AWS Certificate Manager .

Exécutez `openssl version` dans la ligne de commande pour voir si OpenSSL est déjà installé. Si ce n'est pas le cas, vous pouvez créer et installer le code source en suivant les instructions du GitHub dépôt public ou utiliser votre gestionnaire de paquets préféré. OpenSSL étant également

installé sur les images Linux d'Elastic Beanstalk, une alternative rapide consiste à se connecter à une instance dans un environnement d'exécution à l'aide de la commande de l'EB CLI : EC2 eb ssh

```
~/eb$ eb ssh
[ec2-user@ip-255-55-55-255 ~]$ openssl version
OpenSSL 1.0.1k-fips 8 Jan 2015
```

Vous devez créer une clé privée RSA pour générer votre demande de signature de certificat (CSR). Pour créer votre clé privée, utilisez la commande openssl genrsa :

```
[ec2-user@ip-255-55-55-255 ~]$ openssl genrsa 2048 > privatekey.pem
Generating RSA private key, 2048 bit long modulus
.....................................................................................
+++
..............+++
e is 65537 (0x10001)
```

privatekey.pem

> Nom du fichier dans lequel vous souhaitez enregistrer la clé privée. Généralement, la commande openssl genrsa affiche le contenu de la clé privée à l'écran, mais cette commande redirige le résultat vers un fichier. Choisissez un nom de fichier et stockez le fichier dans un emplacement sécurisé de façon à pouvoir le récupérer ultérieurement. Si vous perdez votre clé privée, vous ne pourrez pas utiliser votre certificat.

Un fichier CSR est un fichier que vous pouvez envoyer à une autorité de certification (CA) pour demander un certificat de serveur numérique. Pour créer un fichier CSR, utilisez la commande openssl req :

```
$ openssl req -new -key privatekey.pem -out csr.pem
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
```

Saisissez les informations demandées et appuyez sur Entrée. Le tableau suivant décrit les champs et fournit des exemples pour chacun d'entre eux.

Nom	Description	Exemple
Nom du pays	Abréviation ISO de deux lettres de votre pays.	US = États-Unis
État ou Province	Nom de l'état ou de la province où votre organisation se situe. Vous ne pouvez pas abréger ce nom.	Washington
Nom de la localité	Nom de la ville où votre organisation se situe.	Seattle
Nom de l'organisation	Nom légal complet de votre organisation. N'abrégez pas le nom de votre organisation.	Exemple d'entreprise
Unité organisationnelle	(Facultatif) Informations supplémentaires sur l'organisation.	Marketing
Nom commun	Nom de domaine complet de votre site web. Il doit correspondre au nom de domaine que les utilisateurs voient lorsqu'ils visitent votre site (sinon, des erreurs de certificat s'affichent).	www.exemple.com
Adresse e-mail	Adresse e-mail de l'administrateur du site.	quelquun@exemple.com

Vous pouvez envoyer la demande de signature à un tiers pour la signature, ou la signer vous-même à des fins de test et de développement. Les certificats auto-signés peuvent également être utilisés pour le protocole HTTPS principal entre un équilibreur de charge et des instances. EC2

Pour signer le certificat, utilisez la commande openssl x509. L'exemple suivant utilise la clé privée de l'étape précédente (*privatekey.pem*) et la demande de signature (*csr.pem*) pour créer un certificat public nommé *public.crt* valide pendant des *365* jours.

```
$ openssl x509 -req -days 365 -in csr.pem -signkey privatekey.pem -out public.crt
Signature ok
subject=/C=us/ST=washington/L=seattle/O=example corporation/OU=marketing/
CN=www.example.com/emailAddress=someone@example.com
Getting Private key
```

Conservez la clé privée et le certificat public pour une utilisation ultérieure. Vous pouvez supprimer la demande de signature. Veillez à toujours stocker la clé privée dans un emplacement sécurisé et à ne pas l'ajouter à votre code source.

Pour utiliser le certificat avec la plateforme Windows Server, vous devez le convertir au format PFX. Utilisez la commande suivante pour créer un certificat PFX à partir de la clé privée et des fichiers de certificat public :

```
$ openssl pkcs12 -export -out example.com.pfx -inkey privatekey.pem -in public.crt
Enter Export Password: password
Verifying - Enter Export Password: password
```

Maintenant que vous disposez d'un certificat, vous pouvez le charger dans IAM pour l'utiliser avec un équilibreur de charge, ou configurer les instances de votre environnement pour suspendre HTTPS.

Chargement d'un certificat dans IAM

Pour utiliser votre certificat avec l'équilibreur de charge de votre environnement Elastic Beanstalk, téléchargez le certificat et la clé privée vers (IAM). Gestion des identités et des accès AWS Vous pouvez utiliser un certificat stocké dans IAM avec les équilibreurs de charge Elastic Load Balancing et les distributions Amazon CloudFront .

> **Note**
>
> AWS Certificate Manager (ACM) est l'outil préféré pour approvisionner, gérer et déployer vos certificats de serveur. Pour plus d'informations sur une demande de certificat ACM, consultez Request a Certificate (Demande de certificat) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur AWS Certificate Manager). Pour plus d'informations sur l'importation de certificats tiers dans ACM, consultez Importing Certificates (Importation de certificats) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur AWS Certificate Manager). Utilisez IAM pour télécharger un certificat uniquement si ACM n'est pas disponible dans votre AWS région.

Vous pouvez utiliser le AWS Command Line Interface (AWS CLI) pour télécharger votre certificat. La commande suivante télécharge un certificat auto-signé nommé *`https-cert.crt`* avec une clé privée nommée : *`private-key.pem`*

```
$ aws iam upload-server-certificate --server-certificate-name elastic-beanstalk-x509 --
certificate-body file://https-cert.crt --private-key file://private-key.pem
{
    "ServerCertificateMetadata": {
        "ServerCertificateId": "AS5YBEIONO2Q7CAIHKNGC",
        "ServerCertificateName": "elastic-beanstalk-x509",
        "Expiration": "2017-01-31T23:06:22Z",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:server-certificate/elastic-beanstalk-x509",
        "UploadDate": "2016-02-01T23:10:34.167Z"
    }
}
```

Le `file://` préfixe indique de AWS CLI charger le contenu d'un fichier dans le répertoire courant. *`elastic-beanstalk-x509`*spécifie le nom pour appeler le certificat dans IAM.

Si vous avez acheté un certificat auprès d'une autorité de certification et reçu un fichier de chaîne de certificats, chargez-le également en incluant l'option `--certificate-chain` :

```
$ aws iam upload-server-certificate --server-certificate-name elastic-beanstalk-x509 --
certificate-chain file://certificate-chain.pem --certificate-body file://https-cert.crt
 --private-key file://private-key.pem
```

Notez l'Amazon Resource Name (ARN) de votre certificat. Vous l'utiliserez lors de la mise à jour des paramètres de configuration de votre équilibreur de charge pour utiliser HTTPS.

> **Note**
>
> Un certificat chargé dans IAM reste stocké même s'il n'est plus utilisé dans aucun équilibreur de charge de l'environnement. Il contient des données sensibles. Lorsque vous n'avez plus besoin du certificat pour aucun environnement, veillez à le supprimer. Pour de plus amples informations sur la suppression d'un certificat dans IAM, veuillez consulter https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_server-certs.html#delete-server-certificate.

Pour de plus amples informations sur les certificats de serveur dans IAM, veuillez consulter Utilisation des certificats de serveur dans le Guide de l'utilisateur IAM.

Stockage sécurisé des clés privées dans Amazon S3

La clé privée qui vous permet de vous connecter à votre certificat public est privée et ne doit pas être validée dans le code source. Vous pouvez éviter de stocker les clés privées dans les fichiers de configuration en les chargeant dans Amazon S3 et en configurant Elastic Beanstalk pour télécharger le fichier à partir d'Amazon S3 lors du déploiement de l'application.

L'exemple suivant décrit les sections Ressourceset fichiers d'un fichier de configuration, qui télécharge un fichier de clé privée à partir d'un compartiment Amazon S3.

Example.ebextensions/privatekey.config

```
Resources:
  AWSEBAutoScalingGroup:
    Metadata:
      AWS::CloudFormation::Authentication:
        S3Auth:
          type: "s3"
          buckets: ["elasticbeanstalk-us-west-2-123456789012"]
          roleName:
            "Fn::GetOptionSetting":
              Namespace: "aws:autoscaling:launchconfiguration"
              OptionName: "IamInstanceProfile"
              DefaultValue: "aws-elasticbeanstalk-ec2-role"
files:
  # Private key
  "/etc/pki/tls/certs/server.key":
    mode: "000400"
    owner: root
    group: root
    authentication: "S3Auth"
    source: https://elasticbeanstalk-us-west-2-123456789012.s3.us-west-2.amazonaws.com/
server.key
```

Remplacez l'URL et le nom de compartiment figurant dans l'exemple par vos propres valeurs. La première entrée du fichier ajoute une méthode d'authentification nommée `S3Auth` aux métadonnées du groupe Auto Scaling de l'environnement. Si vous avez configuré un profil d'instance personnalisé pour votre environnement, il est utilisé. Sinon, c'est la valeur par défaut du fichier `aws-elasticbeanstalk-ec2-role` qui s'applique. Le profil d'instance par défaut est autorisé à lire les données du compartiment de stockage Elastic Beanstalk. Si vous utilisez un autre compartiment, ajoutez des autorisations au profil d'instance.

La deuxième entrée utilise la méthode d'authentification `S3Auth` pour télécharger la clé privée à partir de l'URL spécifiée et pour l'enregistrer dans `/etc/pki/tls/certs/server.key`. Le serveur proxy peut ensuite lire la clé privée à partir de cet emplacement afin de mettre les connexions HTTPS hors service dans l'instance.

Le profil d'instance attribué aux EC2 instances de votre environnement doit être autorisé à lire l'objet clé dans le compartiment spécifié. Assurez-vous que le profil d'instance est autorisé à lire l'objet dans IAM, et que les autorisations sur le compartiment et l'objet n'entraînent pas d'interdiction pour le profil d'instance.

Pour afficher les autorisations d'un compartiment

1. Ouvrez la console de gestion Amazon S3.
2. Choisissez un compartiment.
3. Choisissez Properties, puis Autorisations.
4. Vérifiez que votre compte bénéficie d'une autorisation de lecture sur le compartiment.
5. Si une stratégie de compartiment est attachée, choisissez Stratégie de compartiment pour afficher les autorisations attribuées au compartiment.

Configuration de la terminaison HTTPS au niveau de l'équilibreur de charge

Pour mettre à jour votre AWS Elastic Beanstalk environnement afin qu'il utilise le protocole HTTPS, vous devez configurer un écouteur HTTPS pour l'équilibreur de charge de votre environnement. Deux types d'équilibreur de charge prennent en charge un écouteur HTTPS : l'équilibreur Classic Load Balancer et l'équilibreur Application Load Balancer.

Vous pouvez utiliser la console Elastic Beanstalk ou un fichier de configuration pour configurer un écouteur sécurisé et attribuer le certificat.

> **Note**
>
> Les environnements d'instance unique n'ont pas d'équilibreur de charge et ne prennent pas en charge la terminaison HTTPS au niveau de l'équilibreur de charge.

Configuration d'un écouteur sécurisé à l'aide de la console Elastic Beanstalk

Pour attribuer un certificat à l'équilibreur de charge de votre environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Load balancer (Équilibreur de charge), choisissez Edit (Modifier).

   > **Note**
   >
   > Si la catégorie de configuration Load balancer (Équilibreur de charge) ne dispose pas du bouton Edit (Modifier), cela signifie que votre environnement ne dispose pas d'un équilibreur de charge.

5. Sur la page Modifier l'équilibreur de charge, la procédure varie selon le type d'équilibreur de charge associé à votre environnement.
   - Equilibreur de charge classique
     a. Choisissez Add listener (Ajouter un écouteur).
     b. Dans la boîte de dialogue Classic Load Balancer listener (Écouteur Classic Load Balancer), configurez les paramètres suivants :
        - Pour Port d'écoute, tapez le port du trafic entrant, généralement `443`.
        - Pour Protocole d'écoute, choisissez HTTPS.
        - Pour Port de l'instance, tapez `80`.
        - Pour Protocole de l'instance, choisissez HTTP.
        - Pour SSL Certificate (Certificat SSL), sélectionnez votre certificat.
     c. Cliquez sur Add (Ajouter).
   - Application Load Balancer
     a. Choisissez Add listener (Ajouter un écouteur).

b. Dans la boîte de dialogue Application Load Balancer listener (Écouteur de l'équilibreur Application Load Balancer), configurez les paramètres suivants :

- Pour Port, tapez le port du trafic entrant, généralement `443`.
- Pour Protocole, choisissez HTTPS.
- Pour SSL Certificate (Certificat SSL), sélectionnez votre certificat.

c. Cliquez sur Add (Ajouter).

> **Note**
>
> Pour l'équilibreur Classic Load Balancer et l'équilibreur Application Load Balancer, si le menu déroulant n'affiche aucun certificat, vous devez créer ou charger un certificat pour votre nom de domaine personnalisé dans AWS Certificate Manager (ACM) (de préférence). Vous pouvez également charger un certificat dans IAM à l'aide de l' AWS CLI.

- Network Load Balancer

a. Choisissez Add listener (Ajouter un écouteur).

b. Dans la boîte de dialogueNetwork Load Balancer listener (Écouteur de l'équilibreur Network Load Balancer) pour Port, tapez le port de trafic entrant, généralement `443`.

c. Choisissez Ajouter.

6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Configuration d'un écouteur sécurisé avec un fichier de configuration

Vous pouvez configurer un écouteur sécurisé sur votre équilibreur de charge avec un des fichiers de configuration suivants.

Example.ebextensions/securelistener-clb.config

Utilisez cet exemple lorsque votre environnement dispose d'un équilibreur Classic Load Balancer. L'exemple utilise des options dans l'espace de noms `aws:elb:listener` pour configurer un écouteur HTTPS sur le port 443 avec le certificat spécifié et pour transférer le trafic déchiffré vers les instances de votre environnement sur le port 80.

```
option_settings:
```

```
  aws:elb:listener:443:
    SSLCertificateId: arn:aws:acm:us-east-2:1234567890123:certificate/
###################################
    ListenerProtocol: HTTPS
    InstancePort: 80
```

Remplacez le texte en surbrillance par l'ARN de votre certificat. Le certificat peut être un certificat que vous avez créé ou chargé dans AWS Certificate Manager (ACM) (de préférence), ou un certificat que vous avez chargé sur IAM avec le. AWS CLI

Pour de plus amples informations concernant les options de configuration d'un équilibreur Classic Load Balancer, veuillez consulter Espaces de noms pour la configuration d'un Classic Load Balancer.

Example.ebextensions/securelistener-alb.config

Utilisez cet exemple lorsque votre environnement dispose d'un équilibreur Application Load Balancer. L'exemple utilise les options dans l'espace de noms `aws:elbv2:listener` pour configurer un écouteur HTTPS sur le port 443 avec le certificat spécifié. L'écouteur achemine le trafic vers le processus par défaut.

```
option_settings:
  aws:elbv2:listener:443:
    ListenerEnabled: 'true'
    Protocol: HTTPS
    SSLCertificateArns: arn:aws:acm:us-east-2:1234567890123:certificate/
###################################
```

Example.ebextensions/securelistener-nlb.config

Utilisez cet exemple lorsque votre environnement dispose d'un équilibreur Network Load Balancer. L'exemple utilise les options dans l'espace de noms `aws:elbv2:listener` pour configurer un écouteur sur le port 443. L'écouteur achemine le trafic vers le processus par défaut.

```
option_settings:
  aws:elbv2:listener:443:
    ListenerEnabled: 'true'
```

Configuration d'un groupe de sécurité

Si vous configurez votre équilibreur de charge pour transférer le trafic vers un port de l'instance autre que le port 80, vous devez ajouter une règle à votre groupe de sécurité autorisant le trafic entrant via

le port de l'instance à partir de votre équilibreur de charge. Si vous créez votre environnement dans un VPC personnalisé, Elastic Beanstalk ajoute automatiquement cette règle.

Pour ajouter cette règle, ajoutez une clé `Resources` à un fichier de configuration dans le répertoire `.ebextensions` correspondant à votre application.

L'exemple de fichier de configuration suivante ajoute une règle de trafic entrant au groupe de sécurité `AWSEBSecurityGroup`. Cela permet le trafic sur le port 1000 à partir du groupe de sécurité de l'équilibreur de charge.

Example.ebextensions/sg-ingressfromlb.config

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 1000
      FromPort: 1000
      SourceSecurityGroupId: {"Fn::GetAtt" : ["AWSEBLoadBalancerSecurityGroup",
 "GroupId"]}
```

Configuration de la terminaison HTTPS sur l'instance

Vous pouvez utiliser des fichiers de configuration pour configurer le serveur proxy qui transmet le trafic à votre application afin de mettre des connexions HTTPS hors service. Cette opération est utile si vous souhaitez utiliser HTTPS avec un environnement d'instance unique ou si vous configurez votre équilibreur de charge pour transmettre le trafic sans le déchiffrer.

Pour activer le protocole HTTPS, vous devez autoriser le trafic entrant sur le port 443 à destination de l' EC2 instance sur laquelle s'exécute votre application Elastic Beanstalk. Pour ce faire, utilisez la `Resources` clé du fichier de configuration pour ajouter une règle pour le port 443 aux règles d'entrée du groupe de sécurité du AWSEBSecurity groupe.

L'extrait suivant ajoute une règle de trafic entrant au groupe de sécurité `AWSEBSecurityGroup` qui ouvre le port 443 pour tout le trafic, pour un environnement d'instance unique :

`.ebextensions/https-instance-securitygroup.config`

```
Resources:
  sslSecurityGroupIngress:
```

```
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Dans un environnement à charge équilibrée Amazon Virtual Private Cloud (Amazon VPC) par défaut, vous pouvez modifier cette stratégie pour accepter uniquement le trafic provenant de l'équilibreur de charge. Consultez Configuration du end-to-end chiffrement dans un environnement Elastic Beanstalk à charge équilibrée pour obtenir un exemple.

Plateformes

- Résiliation de connexions HTTPS sur des instances EC2 exécutant Docker
- Résiliation du protocole HTTPS sur les EC2 instances exécutant Go
- Résiliation des connexions HTTPS sur des instances EC2 exécutant Java SE
- Suspension des connexions HTTPS sur des instances EC2 exécutant Node.js
- Mettre fin au protocole HTTPS sur les EC2 instances exécutant PHP
- Suspension des connexions HTTPS sur des instances EC2 exécutant Python
- Suspension des connexions HTTPS sur des instances EC2 exécutant Ruby
- Suspension des connexions HTTPS sur des instances EC2 exécutant Tomcat
- Résiliation des connexions HTTPS sur les instances Amazon EC2 exécutant .NET Core sous Linux
- Résiliation du protocole HTTPS sur les EC2 instances Amazon exécutant .NET

Résiliation de connexions HTTPS sur des instances EC2 exécutant Docker

Pour les conteneurs Docker, vous utilisez un fichier de configuration pour activer HTTPS.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant le certificat et la clé privée comme demandé, puis enregistrez-les dans le répertoire `.ebextensions` de votre bundle de fichiers source. Le fichier de configuration effectue les tâches suivantes :

- La clé `files` crée les fichiers suivants sur l'instance :
  `/etc/nginx/conf.d/https.conf`

    Configure le serveur nginx. Ce fichier est chargé lorsque le service nginx démarre.

`/etc/pki/tls/certs/server.crt`

Crée le fichier de certificat sur l'instance. *`certificate file contents`*Remplacez-le par le contenu de votre certificat.

> **Note**
>
> YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

```
    -----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Crée le fichier de clé privée sur l'instance. *`private key contents`*Remplacez-le par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

Example.ebextensions/https-instance.config

```
files:
  /etc/nginx/conf.d/https.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      # HTTPS Server
```

```
    server {
      listen 443;
      server_name localhost;

      ssl on;
      ssl_certificate /etc/pki/tls/certs/server.crt;
      ssl_certificate_key /etc/pki/tls/certs/server.key;

      ssl_session_timeout 5m;

      ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
      ssl_prefer_server_ciphers on;

      location / {
        proxy_pass http://docker;
        proxy_http_version 1.1;

        proxy_set_header Connection "";
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;
      }
    }

/etc/pki/tls/certs/server.crt:
  mode: "000400"
  owner: root
  group: root
  content: |
    -----BEGIN CERTIFICATE-----
    certificate file contents
    -----END CERTIFICATE-----

/etc/pki/tls/certs/server.key:
  mode: "000400"
  owner: root
  group: root
  content: |
    -----BEGIN RSA PRIVATE KEY-----
    private key contents # See note below.
    -----END RSA PRIVATE KEY-----
```

> **Note**
>
> Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et vérifié qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, veuillez consulter Stockage sécurisé des clés privées dans Amazon S3.

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une CloudFormation fonction et y ajoute une règle.

Example extensions .eb/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Dans un environnement à répartition de charge équilibrée, vous configurez l'équilibreur de charge pour qu'il fasse passer le trafic sécurisé sans le toucher, soit pour le déchiffrer et le rechiffrer à des fins de chiffrement. end-to-end

Résiliation du protocole HTTPS sur les EC2 instances exécutant Go

Pour les types de conteneurs Go, vous activez HTTPS avec un fichier de configuration et un fichier de configuration nginx qui configure le serveur nginx pour utiliser HTTPS.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant les espaces réservés pour le certificat et la clé privée comme demandé, puis enregistrez-le dans le répertoire `.ebextensions` de votre bundle de fichiers source. Le fichier de configuration effectue les tâches suivantes :

- La clé `Resources` active le port 443 sur le groupe de sécurité utilisé par l'instance de votre environnement.

- La clé `files` crée les fichiers suivants sur l'instance :

  `/etc/pki/tls/certs/server.crt`

  Crée le fichier de certificat sur l'instance. *`certificate file contents`*Remplacez-le par le contenu de votre certificat.

  > **Note**
  >
  > YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

  Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

  ```
      -----BEGIN CERTIFICATE-----
  certificate file contents
  -----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----
  first intermediate certificate
  -----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----
  second intermediate certificate
  -----END CERTIFICATE-----
  ```

  `/etc/pki/tls/certs/server.key`

  Crée le fichier de clé privée sur l'instance. *`private key contents`*Remplacez-le par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

Example.ebextensions/https-instance.config

```
files:
  /etc/pki/tls/certs/server.crt:
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
```

```
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----
```

> **Note**
>
> Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et vérifié qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, veuillez consulter Stockage sécurisé des clés privées dans Amazon S3.

Placez les éléments suivants dans un fichier avec l'extension `.conf` dans le répertoire `.ebextensions/nginx/conf.d/` de votre bundle de fichiers source (par exemple, `.ebextensions/nginx/conf.d/https.conf`). *app_port*Remplacez-le par le numéro de port que votre application écoute. Cet exemple configure le serveur nginx pour écouter sur le port 443 à l'aide de SSL. Pour plus d'informations sur ces fichiers de configuration sur la plateforme Go, consultez Configuration du serveur proxy.

Example. ebextensions/nginx/conf.d/https.conf

```
# HTTPS server

server {
    listen       443;
    server_name  localhost;

    ssl                  on;
    ssl_certificate      /etc/pki/tls/certs/server.crt;
    ssl_certificate_key  /etc/pki/tls/certs/server.key;

    ssl_session_timeout  5m;

    ssl_protocols  TLSv1 TLSv1.1 TLSv1.2;
    ssl_prefer_server_ciphers   on;
```

```
    location / {
        proxy_pass  http://localhost:app_port;
        proxy_set_header   Connection "";
        proxy_http_version 1.1;
        proxy_set_header        Host            $host;
        proxy_set_header        X-Real-IP       $remote_addr;
        proxy_set_header        X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header        X-Forwarded-Proto https;
    }
}
```

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une CloudFormation fonction et y ajoute une règle.

Example extensions .eb/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Dans un environnement à répartition de charge équilibrée, vous configurez l'équilibreur de charge pour qu'il fasse passer le trafic sécurisé sans le toucher, soit pour le déchiffrer et le rechiffrer à des fins de chiffrement. end-to-end

Résiliation des connexions HTTPS sur des instances EC2 exécutant Java SE

Pour les types de conteneurs Java SE, vous activez HTTPS avec un fichier de configuration .ebextensions, ainsi qu'un fichier de configuration nginx qui configure le serveur nginx pour utiliser HTTPS.

Toutes les AL2 plateformes AL2023/prennent en charge une fonctionnalité de configuration de proxy uniforme. Pour plus d'informations sur la configuration du serveur proxy sur les versions de votre plate-forme exécutant AL2023/AL2, consultezConfiguration du proxy inverse.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant les espaces réservés pour le certificat et la clé privée comme indiqué, puis enregistrez-le dans le répertoire `.ebextensions`. Le fichier de configuration effectue les tâches suivantes :

- La clé `files` crée les fichiers suivants sur l'instance :

  `/etc/pki/tls/certs/server.crt`

  Crée le fichier de certificat sur l'instance. *`certificate file contents`*Remplacez-le par le contenu de votre certificat.

  > **Note**
  >
  > YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

  Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

  ```
      -----BEGIN CERTIFICATE-----
  certificate file contents
  -----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----
  first intermediate certificate
  -----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----
  second intermediate certificate
  -----END CERTIFICATE-----
  ```

  `/etc/pki/tls/certs/server.key`

  Crée le fichier de clé privée sur l'instance. *`private key contents`*Remplacez-le par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

Example.ebextensions/https-instance.config

```
files:
```

```
  /etc/pki/tls/certs/server.crt:
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----
```

> **Note**
>
> Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et vérifié qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, veuillez consulter Stockage sécurisé des clés privées dans Amazon S3.

Placez les éléments suivants dans un fichier avec l'extension `.conf` dans le répertoire `.ebextensions/nginx/conf.d/` de votre bundle de fichiers source (par exemple, `.ebextensions/nginx/conf.d/https.conf`). *`app_port`*Remplacez-le par le numéro de port que votre application écoute. Cet exemple configure le serveur nginx pour écouter sur le port 443 à l'aide de SSL. Pour plus d'informations sur ces fichiers de configuration sur la plateforme Java SE, consultez Configuration du serveur proxy.

Example. ebextensions/nginx/conf.d/https.conf

```
# HTTPS server

server {
    listen       443;
    server_name  localhost;

    ssl                  on;
    ssl_certificate      /etc/pki/tls/certs/server.crt;
    ssl_certificate_key  /etc/pki/tls/certs/server.key;

    ssl_session_timeout  5m;
```

```
    ssl_protocols  TLSv1 TLSv1.1 TLSv1.2;
    ssl_prefer_server_ciphers   on;

    location / {
        proxy_pass  http://localhost:app_port;
        proxy_set_header   Connection "";
        proxy_http_version 1.1;
        proxy_set_header        Host            $host;
        proxy_set_header        X-Real-IP       $remote_addr;
        proxy_set_header        X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header        X-Forwarded-Proto https;
    }
}
```

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une CloudFormation fonction et y ajoute une règle.

Example extensions .eb/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Dans un environnement à répartition de charge équilibrée, vous configurez l'équilibreur de charge pour qu'il fasse passer le trafic sécurisé sans le toucher, soit pour le déchiffrer et le rechiffrer à des fins de chiffrement. end-to-end

Suspension des connexions HTTPS sur des instances EC2 exécutant Node.js

L'exemple de fichier de configuration suivant étend la configuration nginx par défaut pour écouter sur le port 443 et SSL/TLS mettre fin aux connexions avec un certificat public et une clé privée.

Si vous avez configuré votre environnement pour les rapports améliorés sur l'état, vous devez configurer nginx pour générer les journaux d'accès. Pour ce faire, supprimez la mise en commentaire

du bloc de lignes sous le commentaire où il est écrit `# For enhanced health...` en supprimant les caractères # de tête.

Example.ebextensions/https-instance.config

```
files:
  /etc/nginx/conf.d/https.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      # HTTPS server

      server {
          listen       443;
          server_name  localhost;

          ssl                  on;
          ssl_certificate      /etc/pki/tls/certs/server.crt;
          ssl_certificate_key  /etc/pki/tls/certs/server.key;

          ssl_session_timeout  5m;

          ssl_protocols  TLSv1 TLSv1.1 TLSv1.2;
          ssl_prefer_server_ciphers   on;

          # For enhanced health reporting support, uncomment this block:

          #if ($time_iso8601 ~ "^(\d{4})-(\d{2})-(\d{2})T(\d{2})") {
          #    set $year $1;
          #    set $month $2;
          #    set $day $3;
          #    set $hour $4;
          #}
          #access_log /var/log/nginx/healthd/application.log.$year-$month-$day-$hour
 healthd;
          #access_log  /var/log/nginx/access.log  main;

          location / {
              proxy_pass  http://nodejs;
              proxy_set_header   Connection "";
              proxy_http_version 1.1;
              proxy_set_header        Host            $host;
              proxy_set_header        X-Real-IP       $remote_addr;
```

```
            proxy_set_header        X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header        X-Forwarded-Proto https;
        }
    }

/etc/pki/tls/certs/server.crt:
  mode: "000400"
  owner: root
  group: root
  content: |
    -----BEGIN CERTIFICATE-----
    certificate file contents
    -----END CERTIFICATE-----

/etc/pki/tls/certs/server.key:
  mode: "000400"
  owner: root
  group: root
  content: |
    -----BEGIN RSA PRIVATE KEY-----
    private key contents # See note below.
    -----END RSA PRIVATE KEY-----
```

La clé `files` crée les fichiers suivants sur l'instance :

`/etc/nginx/conf.d/https.conf`

Configure le serveur nginx. Ce fichier est chargé lorsque le service nginx démarre.

`/etc/pki/tls/certs/server.crt`

Crée le fichier de certificat sur l'instance. *`certificate file contents`*Remplacez-le par le contenu de votre certificat.

> **Note**
>
> YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

```
    -----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Crée le fichier de clé privée sur l'instance. *`private key contents`*Remplacez-le par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

> **Note**
>
> Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et vérifié qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, veuillez consulter Stockage sécurisé des clés privées dans Amazon S3.

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une CloudFormation fonction et y ajoute une règle.

Example extensions .eb/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
```

```
    FromPort: 443
    CidrIp: 0.0.0.0/0
```

Dans un environnement à répartition de charge équilibrée, vous configurez l'équilibreur de charge pour qu'il fasse passer le trafic sécurisé sans le toucher, soit pour le déchiffrer et le rechiffrer à des fins de chiffrement. end-to-end

Mettre fin au protocole HTTPS sur les EC2 instances exécutant PHP

Pour les types de conteneurs PHP, vous utilisez un fichier de configuration pour permettre au serveur HTTP Apache d'utiliser HTTPS.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant le certificat et la clé privée comme demandé, puis enregistrez-les dans le répertoire `.ebextensions` de votre bundle de fichiers source.

Le fichier de configuration effectue les tâches suivantes :

- La clé `packages` utilise yum pour installer `mod24_ssl`.
- La clé `files` crée les fichiers suivants sur l'instance :

  `/etc/httpd/conf.d/ssl.conf`

  Configure le serveur Apache. Ce fichier se charge lorsque le service Apache démarre.

  `/etc/pki/tls/certs/server.crt`

  Crée le fichier de certificat sur l'instance. *`certificate file contents`*Remplacez-le par le contenu de votre certificat.

  > **Note**
  >
  > YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

  Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

```
    -----BEGIN CERTIFICATE-----
  certificate file contents
  -----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----
  first intermediate certificate
  -----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----
  second intermediate certificate
  -----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Crée le fichier de clé privée sur l'instance. *`private key contents`*Remplacez-le par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

Example.ebextensions/https-instance.config

```
packages:
  yum:
    mod24_ssl : []

files:
  /etc/httpd/conf.d/ssl.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      LoadModule ssl_module modules/mod_ssl.so
      Listen 443
      <VirtualHost *:443>
        <Proxy *>
          Order deny,allow
          Allow from all
        </Proxy>

        SSLEngine             on
        SSLCertificateFile    "/etc/pki/tls/certs/server.crt"
        SSLCertificateKeyFile "/etc/pki/tls/certs/server.key"
        SSLCipherSuite        EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH
        SSLProtocol           All -SSLv2 -SSLv3
        SSLHonorCipherOrder   On
```

```
        SSLSessionTickets     Off

        Header always set Strict-Transport-Security "max-age=63072000;
includeSubdomains; preload"
        Header always set X-Frame-Options DENY
        Header always set X-Content-Type-Options nosniff

        ProxyPass / http://localhost:80/ retry=0
        ProxyPassReverse / http://localhost:80/
        ProxyPreserveHost on
        RequestHeader set X-Forwarded-Proto "https" early

      </VirtualHost>

 /etc/pki/tls/certs/server.crt:
   mode: "000400"
   owner: root
   group: root
   content: |
     -----BEGIN CERTIFICATE-----
     certificate file contents
     -----END CERTIFICATE-----

 /etc/pki/tls/certs/server.key:
   mode: "000400"
   owner: root
   group: root
   content: |
     -----BEGIN RSA PRIVATE KEY-----
     private key contents # See note below.
     -----END RSA PRIVATE KEY-----
```

> **Note**
>
> Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et vérifié qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, veuillez consulter Stockage sécurisé des clés privées dans Amazon S3.

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une CloudFormation fonction et y ajoute une règle.

Example extensions .eb/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Dans un environnement à répartition de charge équilibrée, vous configurez l'équilibreur de charge pour qu'il fasse passer le trafic sécurisé sans le toucher, soit pour le déchiffrer et le rechiffrer à des fins de chiffrement. end-to-end

Suspension des connexions HTTPS sur des instances EC2 exécutant Python

Pour les types de conteneurs Python utilisant Apache HTTP Server avec l'interface WSGI (Web Server Gateway Interface), vous utilisez un fichier de configuration pour permettre l'Apache HTTP Server d'utiliser HTTPS.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant le document certificat et clé privée comme demandé, puis enregistrez-le dans le répertoire `.ebextensions` de votre bundle source. Le fichier de configuration effectue les tâches suivantes :

- La clé `packages` utilise yum pour installer `mod_ssl`.
- La clé `files` crée les fichiers suivants sur l'instance :

  `/etc/httpd/conf.d/ssl.conf`

  Configure le serveur Apache. Si votre application n'est pas nommée `application.py`, remplacez le texte en surbrillance dans la valeur pour `WSGIScriptAlias` par le chemin d'accès local à votre application. Par exemple, une application Django peut être sur `django/wsgi.py`. L'emplacement doit correspondre à la valeur de l'option `WSGIPath` que vous avez définie pour votre environnement.

En fonction des besoins de votre application, vous aurez peut-être également besoin d'ajouter d'autres répertoires au paramètre python-path.

`/etc/pki/tls/certs/server.crt`

Crée le fichier de certificat sur l'instance. *`certificate file contents`*Remplacez-le par le contenu de votre certificat.

> **Note**
>
> YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

```
    -----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Crée le fichier de clé privée sur l'instance. *`private key contents`*Remplacez-le par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

- La clé `container_commands` arrête le service httpd une fois que tout a été configuré de telle sorte que le service utilise le nouveau certificat et le nouveau fichier `https.conf`.

> **Note**
>
> L'exemple fonctionne uniquement dans des environnements utilisant la plateforme Python.

Example.ebextensions/https-instance.config

```
packages:
  yum:
    mod_ssl: []

files:
  /etc/httpd/conf.d/ssl.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      LoadModule wsgi_module modules/mod_wsgi.so
      WSGIPythonHome /var/app/venv/staging-LQM1lest
      WSGISocketPrefix run/wsgi
      WSGIRestrictEmbedded On
      Listen 443
      <VirtualHost *:443>
        SSLEngine on
        SSLCertificateFile "/etc/pki/tls/certs/server.crt"
        SSLCertificateKeyFile "/etc/pki/tls/certs/server.key"

        Alias /static/ /var/app/current/static/
        <Directory /var/app/current/static>
        Order allow,deny
        Allow from all
        </Directory>

        WSGIScriptAlias / /var/app/current/application.py

        <Directory /var/app/current>
        Require all granted
        </Directory>

        WSGIDaemonProcess wsgi-ssl processes=1 threads=15 display-name=%{GROUP} \
          python-path=/var/app/current \
          python-home=/var/app/venv/staging-LQM1lest \
          home=/var/app/current \
```

```
          user=webapp \
          group=webapp
        WSGIProcessGroup wsgi-ssl

      </VirtualHost>

  /etc/pki/tls/certs/server.crt:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----

container_commands:
  01killhttpd:
    command: "killall httpd"
  02waitforhttpddeath:
    command: "sleep 3"
```

> **Note**
>
> Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et vérifié qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, veuillez consulter Stockage sécurisé des clés privées dans Amazon S3.

> **Remarque pour les environnements Amazon Linux 2023**
>
> Sur Amazon Linux 2023, il `mod_wsgi` doit être installé séparément car il n'est pas disponible dans les référentiels de packages. Pour les instructions d'installation, consultez mod_wsgi sur PyPI.

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une CloudFormation fonction et y ajoute une règle.

Example extensions .eb/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Dans un environnement à répartition de charge équilibrée, vous configurez l'équilibreur de charge pour qu'il fasse passer le trafic sécurisé sans le toucher, soit pour le déchiffrer et le rechiffrer à des fins de chiffrement. end-to-end

Suspension des connexions HTTPS sur des instances EC2 exécutant Ruby

Pour les types de conteneur Ruby, la façon dont vous activez HTTPS dépend du type de serveur d'applications utilisé.

Rubriques

- Configuration de HTTPS pour Ruby avec Puma
- Configuration de HTTPS pour Ruby avec Passenger

Configuration de HTTPS pour Ruby avec Puma

Pour les types de conteneurs Ruby qui utilisent Puma en tant que serveur d'applications, vous utilisez un fichier de configuration pour activer HTTPS.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant le certificat et la clé privée comme demandé, puis enregistrez-les dans le répertoire `.ebextensions` de votre bundle de fichiers source. Le fichier de configuration effectue les tâches suivantes :

- La clé `files` crée les fichiers suivants sur l'instance :

  `/etc/nginx/conf.d/https.conf`

  Configure le serveur nginx. Ce fichier est chargé lorsque le service nginx démarre.

  `/etc/pki/tls/certs/server.crt`

  Crée le fichier de certificat sur l'instance. *`certificate file contents`*Remplacez-le par le contenu de votre certificat.

  > **Note**
  >
  > YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

  Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

```
    -----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Crée le fichier de clé privée sur l'instance. *`private key contents`*Remplacez-le par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

Example.ebextensions/https-instance.config

```
files:
  /etc/nginx/conf.d/https.conf:
    content: |
      # HTTPS server

      server {
          listen       443;
          server_name  localhost;

          ssl                  on;
          ssl_certificate      /etc/pki/tls/certs/server.crt;
          ssl_certificate_key  /etc/pki/tls/certs/server.key;

          ssl_session_timeout  5m;

          ssl_protocols  TLSv1 TLSv1.1 TLSv1.2;
          ssl_prefer_server_ciphers   on;

          location / {
              proxy_pass  http://my_app;
              proxy_set_header        Host            $host;
              proxy_set_header        X-Forwarded-For $proxy_add_x_forwarded_for;
              proxy_set_header        X-Forwarded-Proto https;
          }

          location /assets {
            alias /var/app/current/public/assets;
            gzip_static on;
            gzip on;
            expires max;
            add_header Cache-Control public;
          }

          location /public {
            alias /var/app/current/public;
```

```
            gzip_static on;
            gzip on;
            expires max;
            add_header Cache-Control public;
          }
      }

  /etc/pki/tls/certs/server.crt:
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----
```

> **Note**
>
> Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et vérifié qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, veuillez consulter Stockage sécurisé des clés privées dans Amazon S3.

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une CloudFormation fonction et y ajoute une règle.

Example extensions .eb/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
```

```
      CidrIp: 0.0.0.0/0
```

Dans un environnement à répartition de charge équilibrée, vous configurez l'équilibreur de charge pour qu'il fasse passer le trafic sécurisé sans le toucher, soit pour le déchiffrer et le rechiffrer à des fins de chiffrement. end-to-end

Configuration de HTTPS pour Ruby avec Passenger

Pour les types de conteneurs Ruby qui utilisent Passenger en tant que serveur d'applications, vous utilisez un fichier de configuration et un fichier JSON pour activer HTTPS.

Pour configurer HTTPS pour Ruby avec Passenger

1. Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant le certificat et la clé privée comme demandé, puis enregistrez-les dans le répertoire `.ebextensions` de votre bundle de fichiers source. Le fichier de configuration effectue les tâches suivantes :

   - La clé `files` crée les fichiers suivants sur l'instance :

     `/etc/pki/tls/certs/server.crt`

     Crée le fichier de certificat sur l'instance. *`certificate file contents`*Remplacez-le par le contenu de votre certificat.

     > **Note**
     >
     > YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

     Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

     ```
         -----BEGIN CERTIFICATE-----
     certificate file contents
     -----END CERTIFICATE-----
     -----BEGIN CERTIFICATE-----
     first intermediate certificate
     -----END CERTIFICATE-----
     ```

```
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Crée le fichier de clé privée sur l'instance. *`private key contents`*Remplacez-le par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

Example Extrait .Ebextensions pour la configuration de HTTPS pour Ruby avec Passenger

```
files:
  /etc/pki/tls/certs/server.crt:
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----
```

> **Note**
>
> Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et vérifié qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, veuillez consulter Stockage sécurisé des clés privées dans Amazon S3.

2. Créez un fichier texte et ajoutez le JSON suivant au fichier. Enregistrez-le dans le répertoire racine de votre groupe source avec le nom `passenger-standalone.json`. Ce fichier JSON configure Passenger pour utiliser HTTPS.

> **Important**
>
> Ce fichier JSON ne doit pas contenir une marque d'ordre d'octet (BOM). Si tel est le cas, la bibliothèque Passenger JSON ne lira pas le fichier correctement et le service Passenger ne démarrera pas.

Example passenger-standalone.json

```
{
  "ssl" : true,
  "ssl_port" : 443,
  "ssl_certificate" : "/etc/pki/tls/certs/server.crt",
  "ssl_certificate_key" : "/etc/pki/tls/certs/server.key"
}
```

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une CloudFormation fonction et y ajoute une règle.

Example extensions .eb/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Dans un environnement à répartition de charge équilibrée, vous configurez l'équilibreur de charge pour qu'il fasse passer le trafic sécurisé sans le toucher, soit pour le déchiffrer et le rechiffrer à des fins de chiffrement. end-to-end

Suspension des connexions HTTPS sur des instances EC2 exécutant Tomcat

Pour les types de conteneurs Tomcat, vous devez utiliser un fichier de configuration afin d'autoriser Apache HTTP Server à utiliser HTTPS lorsqu'il agit en tant que proxy inverse pour Tomcat.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant le certificat et la clé privée comme demandé, puis enregistrez-les dans le répertoire `.ebextensions` de votre bundle de fichiers source. Le fichier de configuration effectue les tâches suivantes :

- La clé `files` crée les fichiers suivants sur l'instance :

  `/etc/pki/tls/certs/server.crt`

  Crée le fichier de certificat sur l'instance. *`certificate file contents`*Remplacez-le par le contenu de votre certificat.

  > **Note**
  >
  > YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

  `/etc/pki/tls/certs/server.key`

  Crée le fichier de clé privée sur l'instance. *`private key contents`*Remplacez-le par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

  `/opt/elasticbeanstalk/hooks/appdeploy/post/99_start_httpd.sh`

  Crée un script de hook post-déploiement pour redémarrer le service httpd.

Example.ebextensions/https-instance.config

```
files:
  /etc/pki/tls/certs/server.crt:
    mode: "000400"
    owner: root
    group: root
    content: |
```

```
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----

  /opt/elasticbeanstalk/hooks/appdeploy/post/99_start_httpd.sh:
    mode: "000755"
    owner: root
    group: root
    content: |
      #!/usr/bin/env bash
      sudo service httpd restart
```

Vous devez également configurer le serveur proxy de votre environnement pour écouter sur le port 443. La configuration Apache 2.4 suivante ajoute un écouteur sur le port 443. Pour en savoir plus, veuillez consulter la section Configuration du serveur proxy.

Example. ebextensions/httpd/conf.d/ssl.conf

```
Listen 443
<VirtualHost *:443>
  ServerName server-name
  SSLEngine on
  SSLCertificateFile "/etc/pki/tls/certs/server.crt"
  SSLCertificateKeyFile "/etc/pki/tls/certs/server.key"

  <Proxy *>
    Require all granted
  </Proxy>
  ProxyPass / http://localhost:8080/ retry=0
  ProxyPassReverse / http://localhost:8080/
  ProxyPreserveHost on

  ErrorLog /var/log/httpd/elasticbeanstalk-ssl-error_log
```

```
</VirtualHost>
```

Votre fournisseur de certificats peut inclure des certificats intermédiaires, que vous pouvez installer pour améliorer la compatibilité avec les clients mobiles. Configurez Apache avec un bundle de certificats intermédiaires en ajoutant ce qui suit à votre fichier de configuration SSL (consultez Extension et remplacement de la configuration Apache par défaut — Amazon Linux AMI () AL1 pour l'emplacement) :

- Dans le contenu du fichier `ssl.conf`, spécifiez le fichier de chaîne :

```
SSLCertificateKeyFile "/etc/pki/tls/certs/server.key"
SSLCertificateChainFile "/etc/pki/tls/certs/gd_bundle.crt"
SSLCipherSuite        EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH
```

- Ajoutez une nouvelle entrée à la clé `files` avec le contenu des certificats intermédiaires :

```
files:
  /etc/pki/tls/certs/gd_bundle.crt:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN CERTIFICATE-----
      First intermediate certificate
      -----END CERTIFICATE-----
      -----BEGIN CERTIFICATE-----
      Second intermediate certificate
      -----END CERTIFICATE-----
```

> **Note**
>
> Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et vérifié qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, veuillez consulter Stockage sécurisé des clés privées dans Amazon S3.

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une CloudFormation fonction et y ajoute une règle.

Example extensions .eb/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Dans un environnement à répartition de charge équilibrée, vous configurez l'équilibreur de charge pour qu'il fasse passer le trafic sécurisé sans le toucher, soit pour le déchiffrer et le rechiffrer à des fins de chiffrement. end-to-end

Résiliation des connexions HTTPS sur les instances Amazon EC2 exécutant .NET Core sous Linux

Pour les types de conteneur .NET Core sous Linux, vous activez HTTPS avec un fichier de configuration `.ebextensions` et un fichier de configuration nginx qui configure le serveur nginx pour utiliser HTTPS.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant les espaces réservés pour le certificat et la clé privée comme indiqué, puis enregistrez-le dans le répertoire `.ebextensions`. Le fichier de configuration effectue les tâches suivantes :

- La clé `files` crée les fichiers suivants sur l'instance :

  `/etc/pki/tls/certs/server.crt`

  Crée le fichier de certificat sur l'instance. *`certificate file contents`*Remplacez-le par le contenu de votre certificat.

> **Note**
>
> YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

```
    -----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Crée le fichier de clé privée sur l'instance. *`private key contents`*Remplacez-le par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

Example.ebextensions/https-instance.config

```
files:
  /etc/pki/tls/certs/server.crt:
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
```

```
-----END RSA PRIVATE KEY-----
```

> **Note**
>
> Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et vérifié qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, veuillez consulter Stockage sécurisé des clés privées dans Amazon S3.

Placez les éléments suivants dans un fichier avec l'extension `.conf` dans le répertoire `.platform/nginx/conf.d/` de votre bundle de fichiers source (par exemple, `.platform/nginx/conf.d/https.conf`). *`app_port`*Remplacez-le par le numéro de port que votre application écoute. Cet exemple configure le serveur nginx pour écouter sur le port 443 à l'aide de SSL. Pour de plus amples informations sur les fichiers de configuration sur la plateforme .NET Core sous Linux, veuillez consulter the section called “Serveur proxy”.

Example. platform/nginx/conf.d/https.conf

```
# HTTPS server

server {
    listen       443 ssl;
    server_name  localhost;

    ssl_certificate      /etc/pki/tls/certs/server.crt;
    ssl_certificate_key  /etc/pki/tls/certs/server.key;

    ssl_session_timeout  5m;

    ssl_protocols  TLSv1 TLSv1.1 TLSv1.2;
    ssl_prefer_server_ciphers   on;

    location / {
        proxy_pass  http://localhost:app_port;
        proxy_set_header   Connection "";
        proxy_http_version 1.1;
        proxy_set_header        Host            $host;
        proxy_set_header        X-Real-IP       $remote_addr;
        proxy_set_header        X-Forwarded-For $proxy_add_x_forwarded_for;
```

```
        proxy_set_header        X-Forwarded-Proto https;
    }
}
```

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une CloudFormation fonction et y ajoute une règle.

Example extensions .eb/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Dans un environnement à répartition de charge équilibrée, vous configurez l'équilibreur de charge pour qu'il fasse passer le trafic sécurisé sans le toucher, soit pour le déchiffrer et le rechiffrer à des fins de chiffrement. end-to-end

Résiliation du protocole HTTPS sur les EC2 instances Amazon exécutant .NET

Le fichier de configuration suivant crée et exécute un PowerShell script Windows qui exécute les tâches suivantes :

- Vérifie la présence d'un certificat HTTPS existant lié au port 443.
- Obtient le certificat PFX à partir d'un compartiment Amazon S3.

> **Note**
>
> Ajoutez une `AmazonS3ReadOnlyAccess` politique au `aws-elasticbeanstalk-ec2-role` pour accéder au certificat SSL dans le compartiment Amazon S3.

- Obtient le mot de passe de AWS Secrets Manager.

> **Note**
>
> Ajoutez une instruction `aws-elasticbeanstalk-ec2-role` qui autorise l'`secretsmanager:GetSecretValue`action pour le secret qui contient le mot de passe du certificat

- Installe le certificat.
- Lie le certificat au port 443.

> **Note**
>
> Pour supprimer le point de terminaison HTTP (port 80), incluez la commande `Remove-WebBinding` sous la section Remove the HTTP binding de l'exemple.

Example extensions .eb/ .config https-instance-dotnet

```
files:
  "C:\\certs\\install-cert.ps1":
    content: |
      import-module webadministration
      ## Settings - replace the following values with your own
      $bucket = "amzn-s3-demo-bucket"  ## S3 bucket name
      $certkey = "example.com.pfx"    ## S3 object key for your PFX certificate
      $secretname = "example_secret"  ## AWS Secrets Manager name for a secret that
 contains the certificate's password
      ##

      # Set variables
      $certfile = "C:\cert.pfx"
      $pwd = Get-SECSecretValue -SecretId $secretname | select -expand SecretString

      # Clean up existing binding
      if ( Get-WebBinding "Default Web Site" -Port 443 ) {
        Echo "Removing WebBinding"
        Remove-WebBinding -Name "Default Web Site" -BindingInformation *:443:
      }
      if ( Get-Item -path IIS:\SslBindings\0.0.0.0!443 ) {
        Echo "Deregistering WebBinding from IIS"
        Remove-Item -path IIS:\SslBindings\0.0.0.0!443
```

```
      }

      # Download certificate from S3
      Read-S3Object -BucketName $bucket -Key $certkey -File $certfile

      # Install certificate
      Echo "Installing cert..."
      $securepwd = ConvertTo-SecureString -String $pwd -Force -AsPlainText
      $cert = Import-PfxCertificate -FilePath $certfile cert:\localMachine\my -Password
 $securepwd

      # Create site binding
      Echo "Creating and registering WebBinding"
      New-WebBinding -Name "Default Web Site" -IP "*" -Port 443 -Protocol https
      New-Item -path IIS:\SslBindings\0.0.0.0!443 -value $cert -Force

      ## Remove the HTTP binding
      ## (optional) Uncomment the following line to unbind port 80
      # Remove-WebBinding -Name "Default Web Site" -BindingInformation *:80:
      ##

      # Update firewall
      netsh advfirewall firewall add rule name="Open port 443" protocol=TCP
 localport=443 action=allow dir=OUT

commands:
  00_install_ssl:
    command: powershell -NoProfile -ExecutionPolicy Bypass -file C:\\certs\\install-
cert.ps1
```

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une CloudFormation fonction et y ajoute une règle.

Example extensions .eb/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
```

```
    FromPort: 443
    CidrIp: 0.0.0.0/0
```

Dans un environnement à répartition de charge équilibrée, vous configurez l'équilibreur de charge pour qu'il fasse passer le trafic sécurisé sans le toucher, soit pour le déchiffrer et le rechiffrer à des fins de chiffrement. end-to-end

Configuration du end-to-end chiffrement dans un environnement Elastic Beanstalk à charge équilibrée

La suspension des connexions sécurisées au niveau de l'équilibreur de charge et l'utilisation de HTTP sur le backend peuvent s'avérer suffisantes pour votre application. Le trafic réseau entre les ressources AWS ne peut pas être écouté par les instances qui ne font pas partie de la connexion, même si elles sont exécutées sous le même compte.

Toutefois, si vous développez une application qui doit respecter des réglementations externes strictes, vous pouvez être contraint de sécuriser toutes les connexions réseau. Pour respecter ces obligations, vous pouvez utiliser la console Elastic Beanstalk ou les fichiers de configuration afin que l'équilibreur de charge de votre environnement Elastic Beanstalk se connecte aux instances backend en toute sécurité. La procédure suivante se concentre sur les fichiers de configuration.

Commencez par ajouter un écouteur sécurisé à votre équilibreur de charge, si vous ne l'avez pas encore fait.

Vous devez également configurer les instances de votre environnement pour autoriser l'écoute sur le port sécurisé et la suspension des connexions HTTPS. La configuration varie selon la plateforme. Pour obtenir des instructions, consultez Configuration de la terminaison HTTPS sur l'instance. Vous pouvez utiliser un certificat auto-signé pour les EC2 instances sans problème.

Ensuite, configurez l'écouteur pour transférer le trafic à l'aide du protocole HTTPS vers le port sécurisé utilisé par votre application. Utilisez l'un des fichiers de configuration suivants en fonction du type d'équilibreur de charge utilisé par votre environnement.

`.ebextensions/https-reencrypt-clb.config`

Utilisez ce fichier de configuration avec un équilibreur Classic Load Balancer. En plus de configurer l'équilibreur de charge, le fichier de configuration modifie également par défaut la vérification de l'état pour utiliser le port 443 et HTTPS, afin de veiller à ce que l'équilibreur de charge puisse se connecter en toute sécurité.

```
option_settings:
  aws:elb:listener:443:
    InstancePort: 443
    InstanceProtocol: HTTPS
  aws:elasticbeanstalk:application:
    Application Healthcheck URL: HTTPS:443/
```

`.ebextensions/https-reencrypt-alb.config`

Utilisez ce fichier de configuration avec un équilibreur Application Load Balancer.

```
option_settings:
  aws:elbv2:listener:443:
    DefaultProcess: https
    ListenerEnabled: 'true'
    Protocol: HTTPS
  aws:elasticbeanstalk:environment:process:https:
    Port: '443'
    Protocol: HTTPS
```

`.ebextensions/https-reencrypt-nlb.config`

Utilisez ce fichier de configuration avec un équilibreur Network Load Balancer.

```
option_settings:
  aws:elbv2:listener:443:
    DefaultProcess: https
    ListenerEnabled: 'true'
  aws:elasticbeanstalk:environment:process:https:
    Port: '443'
```

L'option `DefaultProcess` est nommée ainsi en raison des équilibreurs de charge Application Load Balancer, qui peuvent avoir des écouteurs autres que ceux par défaut sur le même port pour le trafic vers des chemins spécifiques (voir Application Load Balancer pour plus de détails). Pour un équilibreur de charge Network Load Balancer, l'option spécifie le seul processus cible de cet écouteur.

Dans cet exemple, nous avons nommé le processus `https`, car il écoute le trafic sécurisé (HTTPS). L'écouteur envoie le trafic vers le processus sur le port désigné à l'aide du protocole TCP, car un équilibreur de charge Network Load Balancer fonctionne uniquement avec TCP. C'est tout à fait normal, car le trafic réseau pour HTTP et HTTPS est mis en œuvre par-dessus TCP.

> **Note**
>
> L'interface de ligne de commande (CLI) EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Consultez Valeurs recommandées pour plus de détails.

Dans la tâche suivante, vous devez modifier le groupe de sécurité de l'équilibreur de charge pour autoriser le trafic. Selon l'Amazon Virtual Private Cloud (Amazon VPC) dans lequel vous lancez votre environnement (le VPC par défaut ou un VPC personnalisé), le groupe de sécurité de l'équilibreur de charge varie. Dans un VPC par défaut, Elastic Load Balancing fournit un groupe de sécurité par défaut que tous les équilibreurs de charge peuvent utiliser. Dans un VPC Amazon que vous créez, Elastic Beanstalk crée un groupe de sécurité que l'équilibreur de charge doit utiliser.

Pour prendre en charge les deux scénarios, vous pouvez créer un groupe de sécurité et demander à Elastic Beanstalk de l'utiliser. Le fichier de configuration suivant crée un groupe de sécurité et l'associe à l'équilibreur de charge.

`.ebextensions/https-lbsecuritygroup.config`

```
option_settings:
  # Use the custom security group for the load balancer
  aws:elb:loadbalancer:
    SecurityGroups: '`{ "Ref" : "loadbalancersg" }`'
    ManagedSecurityGroup: '`{ "Ref" : "loadbalancersg" }`'

Resources:
  loadbalancersg:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: load balancer security group
      VpcId: vpc-########
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
```

```
          CidrIp: 0.0.0.0/0
      SecurityGroupEgress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
```

Remplacez le texte en surbrillance par l'ID de votre VPC par défaut ou personnalisé. L'exemple précédent inclut le trafic entrant et sortant via le port 80 pour autoriser les connexions HTTP. Vous pouvez supprimer ces propriétés si vous souhaitez autoriser uniquement les connexions sécurisées.

Enfin, ajoutez des règles de trafic entrant et sortant qui autorisent la communication sur le port 443 entre le groupe de sécurité de l'équilibreur de charge et le groupe de sécurité des instances.

`.ebextensions/https-backendsecurity.config`

```
Resources:
  # Add 443-inbound to instance security group (AWSEBSecurityGroup)
  httpsFromLoadBalancerSG:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      SourceSecurityGroupId: {"Fn::GetAtt" : ["loadbalancersg", "GroupId"]}
  # Add 443-outbound to load balancer security group (loadbalancersg)
  httpsToBackendInstances:
    Type: AWS::EC2::SecurityGroupEgress
    Properties:
      GroupId: {"Fn::GetAtt" : ["loadbalancersg", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      DestinationSecurityGroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
```

Si vous effectuez cette opération en dehors du processus de création du groupe de sécurité, vous pouvez limiter les groupes de sécurité source et de destination sans créer de dépendance circulaire.

Après avoir effectué toutes les tâches précédentes, l'équilibreur de charge se connecte à vos instances backend en toute sécurité à l'aide du protocole HTTPS. L'équilibreur de charge ne s'occupe

pas de savoir si le certificat de votre instance est auto-signé ou a été émis par une autorité de certification compétente : il accepte tous les certificats qui lui sont présentés.

Pour modifier ce comportement, vous pouvez ajouter des stratégies à l'équilibreur de charge en lui demandant de ne faire confiance qu'à un certificat spécifique. Le fichier de configuration suivant crée deux stratégies. La première spécifie un certificat public, et la deuxième indique à l'équilibreur de charge qu'il ne doit faire confiance qu'à ce certificat pour les connexions au port 443 de l'instance.

`.ebextensions/https-backendauth.config`

```
option_settings:
  # Backend Encryption Policy
  aws:elb:policies:backendencryption:
    PublicKeyPolicyNames: backendkey
    InstancePorts:  443
  # Public Key Policy
  aws:elb:policies:backendkey:
    PublicKey: |
      -----BEGIN CERTIFICATE-----
      ################################################################
      ################################################################
      ################################################################
      ################################################################
      ################################################
      -----END CERTIFICATE-----
```

Remplacez le texte surligné par le contenu du certificat public de votre EC2 instance.

Configuration de l'équilibreur de charge de votre environnement pour TCP Passthrough

Si vous ne souhaitez pas que l'équilibreur de charge de votre AWS Elastic Beanstalk environnement déchiffre le trafic HTTPS, vous pouvez configurer l'écouteur sécurisé pour qu'il relaie les demandes aux instances de backend telles quelles.

> **Important**
>
> La configuration de l'équilibreur de charge pour relayer le trafic HTTPS sans le déchiffrer présente un inconvénient. L'équilibreur de charge ne peut pas voir les demandes chiffrées et ne peut donc pas optimiser le routage ni communiquer les mesures de réponse.

Configurez d'abord les EC2 instances de votre environnement pour mettre fin au protocole HTTPS. Testez la configuration sur un environnement d'instance unique pour vous assurer que tout fonctionne avant d'ajouter un équilibreur de charge au mélange.

Ajoutez un fichier de configuration à votre projet pour configurer un auditeur sur le port 443 qui passe des paquets TCP en l'état vers le port 443 sur les instances backend :

`.ebextensions/https-lb-passthrough.config`

```
option_settings:
  aws:elb:listener:443:
    ListenerProtocol: TCP
    InstancePort: 443
    InstanceProtocol: TCP
```

Dans un VPC Amazon Virtual Private Cloud (Amazon VPC) par défaut, vous devez également ajouter une règle au groupe de sécurité des instances pour autoriser le trafic entrant sur 443 à partir de l'équilibreur de charge :

`.ebextensions/https-instance-securitygroup.config`

```
Resources:
  443inboundfromloadbalancer:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      SourceSecurityGroupName: { "Fn::GetAtt": ["AWSEBLoadBalancer",
 "SourceSecurityGroup.GroupName"] }
```

Dans un VPC personnalisé, Elastic Beanstalk met automatiquement à jour la configuration du groupe de sécurité.

Configuration de la redirection HTTP vers HTTPS

Cette rubrique décrit comment gérer le trafic HTTP vers votre application si les utilisateurs finaux continuent de le lancer. Pour ce faire, vous devez configurer la redirection de HTTP vers HTTPS, parfois appelée forçage HTTPS.

Pour configurer la redirection, vous devez d'abord configurer votre environnement pour gérer le trafic HTTPS. Ensuite, vous redirigez le trafic HTTP vers HTTPS. Ces deux étapes sont abordées dans les sous-sections suivantes.

Configurer votre environnement pour gérer le trafic HTTPS

Selon la configuration d'équilibrage de charge de votre environnement, effectuez l'une des opérations suivantes :

- Environnement à charge équilibrée : configurez votre équilibreur de charge pour résilier les connexions HTTPS.
- Environnement à instance unique : configurez votre application pour résilier les connexions HTTPS au niveau de l'instance. Cette configuration dépend de la plateforme de votre environnement.

Rediriger le trafic HTTP vers HTTPS

Pour rediriger le trafic HTTP vers HTTPS pour votre application, vous pouvez configurer les serveurs Web sur les instances de votre environnement ou configurer l'Application Load Balancer de l'environnement.

Configuration des serveurs Web de l'instance

Cette méthode fonctionne sur n'importe quel environnement de serveur Web. Configurez les serveurs Web de vos EC2 instances Amazon pour qu'ils répondent au trafic HTTP avec un statut de réponse de redirection HTTP.

Cette configuration dépend de la plateforme de votre environnement. Recherchez le dossier de votre plateforme dans la `https-redirect`collection sur GitHub, et utilisez l'exemple de fichier de configuration contenu dans ce dossier.

Si votre environnement utilise les vérifications de l'état Elastic Load Balancing, l'équilibreur de charge attend qu'une instance saine réponde aux messages de vérification de l'état HTTP avec des réponses HTTP 200 (OK). Par conséquent, votre serveur web ne devrait pas rediriger ces messages vers HTTPS. Les exemples de fichiers de configuration dans `https-redirect` gèrent cette exigence correctement.

Configuration de l'équilibreur de charge

Cette méthode fonctionne si vous disposez d'un environnement d'équilibrage de charge qui utilise un Application Load Balancer. Un Application Load Balancer peut envoyer des réponses de redirection

au fur et à mesure que le trafic HTTP arrive. Dans ce cas, vous n'avez pas besoin de configurer la redirection sur les instances de votre environnement.

Nous avons deux exemples de fichiers de configuration GitHub qui montrent comment configurer un Application Load Balancer pour la redirection.

- Le fichier de configuration `alb-http-to-https-redirection-full.config` crée un écouteur HTTPS sur le port 443 et modifie l'écouteur de port 80 par défaut pour rediriger le trafic HTTP entrant vers HTTPS.
- Le fichier `alb-http-to-https-redirection.config`de configuration s'attend à ce que l'écouteur 443 soit défini. Pour le définir, vous pouvez utiliser les espaces de noms de configuration standard d'Elastic Beanstalk ou la console Elastic Beanstalk. Ensuite, il prend soin de modifier l'écouteur du port 80 pour la redirection.

Plateformes Elastic Beanstalk

AWS Elastic Beanstalk fournit une variété de plateformes sur lesquelles vous pouvez créer vos applications. Vous concevez votre application web sur l'une de ces plateformes. Elastic Beanstalk déploie votre code sur la version de la plateforme que vous avez sélectionnée pour créer un environnement d'application actif.

Elastic Beanstalk propose des plateformes dans différents langages de programmation, serveurs d'applications, et conteneurs Docker. Certaines plateformes possèdent plusieurs versions prise en charge simultanément.

Rubriques

- Glossaire des plateformes Elastic Beanstalk
- Modèle de responsabilité partagée pour la maintenance de la plateforme Elastic Beanstalk
- Stratégie de prise en charge de la plateforme Elastic Beanstalk
- Calendrier de sortie de la plateforme Elastic Beanstalk
- Plateformes prises en charge par Elastic Beanstalk
- Plateformes Linux Elastic Beanstalk
- Extension des plateformes Linux Elastic Beanstalk

Glossaire des plateformes Elastic Beanstalk

Vous trouverez ci-dessous les principaux termes relatifs aux AWS Elastic Beanstalk plateformes et à leur cycle de vie.

Environnement d’exécution

> Langage de programmation propre au logiciel d'environnement d'exécution (infrastructure, bibliothèques, interpréteur, VM, etc.) requis pour exécuter votre code d'application.

Composants Elastic Beanstalk

> Composants logiciels qu'Elastic Beanstalk ajoute à une plateforme pour activer la fonctionnalité Elastic Beanstalk. Par exemple, l'agent amélioré pour l'état de santé est nécessaire pour recueillir et rapporter les données d'état de santé.

Plateforme

Combinaison entre un système d'exploitation (OS), un environnement d'exécution, un serveur web, un serveur d'applications et les composants Elastic Beanstalk. Les plateformes fournissent des composants qui sont disponibles pour exécuter votre application.

Version de plateforme

Combinaison entre les versions spécifiques d'un système d'exploitation (OS), un environnement d'exécution, un serveur web, un serveur d'applications et les composants Elastic Beanstalk. Vous créez un environnement Elastic Beanstalk basé sur une version de plateforme et déployez votre application sur cette dernière.

Une version de plateforme possède un numéro de version sémantique au format X.Y.Z, où X est la version majeure, Y est la version mineure et Z est la version de correctif.

Une version de plateforme peut se trouver dans l'un des états suivants :

- Recommandé — La dernière version de la plateforme dans une branche de plateforme prise en charge. Cette version contient le plus grand nombre de up-to-date composants et est recommandée pour une utilisation dans les environnements de production. Lorsqu'Elastic Beanstalk publie une nouvelle version de plateforme, celle-ci remplace la version précédente et devient la version de plateforme recommandée pour la branche de plateforme correspondante.
- Non recommandé — Toute version de plate-forme qui n'est pas la dernière version de sa branche de plate-forme. Bien que ces versions puissent rester fonctionnelles, nous vous recommandons vivement de passer à la dernière version de la plateforme. Vous pouvez utiliser les mises à jour gérées de la plateforme pour vous aider à rester up-to-date automatique.

Vous pouvez vérifier si une version de plate-forme est recommandée à l'aide de la commande AWS CLI describe-platform-version et en vérifiant le `PlatformLifecycleState` champ.

Branche de plateforme

Une ligne de versions de plateforme partageant des versions spécifiques (généralement majeures) de certains de leurs composants, notamment le système d'exploitation (OS), l'exécution ou des composants Elastic Beanstalk. Par exemple : Python 3.13 s'exécute sur Amazon Linux 2023 64 bits ; IIS 10.0 s'exécute sur Windows Server 2025 64 bits. Les succursales de la plateforme reçoivent des mises à jour sous la forme de nouvelles versions de la plateforme. Chaque version de plateforme successive dans une branche est une mise à jour de la précédente.

La version recommandée dans chaque branche de plate-forme prise en charge est disponible sans condition pour la création d'un environnement. Les versions précédentes de la plateforme restent accessibles aux comptes dotés d'environnements actifs ou résiliés qui les utilisaient au moment où elles ont été remplacées par une nouvelle version. Les versions précédentes de la plate-forme sont dépourvues de la plupart des up-to-date composants et leur utilisation n'est pas recommandée.

> **Note**
>
> Si vous avez besoin d'accéder aux versions précédentes de la plateforme au-delà de la disponibilité standard décrite ci-dessus, vous pouvez contacter le AWS Support Center pour obtenir de l'aide.

Une branche de plateforme peut se trouver dans l'un des états suivants :

- Prise en charge – Branche de plateforme actuelle. Elle comprend uniquement des composants pris en charge. Les composants pris en charge n'ont pas atteint la fin de vie (EOL), comme indiqué par leurs fournisseurs. Elle bénéficie de mises à jour continues de la plateforme et est recommandée pour une utilisation dans les environnements de production. Pour obtenir une liste de branches de plateformes prises en charge, consultez Plateformes Elastic Beanstalk prises en charge dans le guide Plateformes AWS Elastic Beanstalk .
- Bêta – Version préliminaire, pré-version de la branche de plateforme. Elle est de nature expérimentale. Elle peut bénéficier des mises à jour continues de la plateforme pendant un certain temps, mais ne dispose pas d'une prise en charge à long terme. Une branche de plateforme bêta n'est pas recommandée pour une utilisation dans des environnements de production. Utilisez-la uniquement à des fins d'évaluation. Pour obtenir la liste des branches de plateforme bêta, consultez Versions de plateformes Elastic Beanstalk en bêta publique dans le guide Plateformes AWS Elastic Beanstalk .
- Obsolète : branche de plate-forme dans laquelle un ou plusieurs composants (tels que le moteur d'exécution ou le système d'exploitation) approchent de la fin de vie (EOL) ou ont atteint la fin de vie, comme indiqué par leurs fournisseurs. Alors qu'une branche de plateforme obsolète continue de recevoir de nouvelles versions de plate-forme jusqu'à sa date de retrait, les composants qui ont atteint la fin de leur durée de vie ne sont pas mis à jour. Par exemple, si une version d'exécution atteint la fin de vie, la branche de plate-forme sera marquée comme obsolète mais continuera à recevoir les mises à jour du système d'exploitation jusqu'à la date de mise hors service de la branche de plate-forme. La branche de la plateforme ne continuera

pas à recevoir les mises à jour de la version d'exécution EOL. L'utilisation d'une branche de plateforme obsolète n'est pas recommandée.

- Retiré : branche de la plateforme qui ne reçoit plus aucune mise à jour. Les branches de plateforme retirées ne sont pas disponibles pour créer de nouveaux environnements Elastic Beanstalk à l'aide de la console Elastic Beanstalk. Si votre environnement utilise une branche de plate-forme retirée, vous devez passer à une branche de plate-forme prise en charge pour continuer à recevoir des mises à jour. L'utilisation d'une branche de plateforme retirée n'est pas recommandée. Pour plus de détails sur les succursales de plateforme retirées, consultezthe section called “Stratégie de prise en charge de la plateforme”. Pour consulter la liste des succursales de la plateforme dont la mise à la retraite est prévue, consultez le calendrier des succursales de la plateforme qui prennent leur retraite. Pour consulter les anciennes succursales de la plateforme retirées, consultez l'historique des succursales de la plateforme retirées.

Si votre environnement utilise une branche de plateforme obsolète ou hors service, nous vous recommandons de la mettre à jour vers une version de plateforme dans une branche de plateforme prise en charge. Pour en savoir plus, consultez the section called “Mises à jour de plateforme”.

Vous pouvez vérifier l'état d'une branche de plate-forme à l'aide de la commande AWS CLI describe-platform-version et en vérifiant le `PlatformBranchLifecycleState` champ.

Mise à jour de plateforme

Version d'une nouvelle version de la plateforme qui contient des mises à jour de certains composants de la plateforme : système d'exploitation, environnement d'exécution, serveur Web, serveur d'applications et composants Elastic Beanstalk. Lorsqu'Elastic Beanstalk publie une nouvelle version de plateforme, celle-ci remplace la version précédente et devient la version de plateforme recommandée pour la branche de plateforme correspondante. Les mises à jour de la plateforme suivent la taxonomie sémantique des versions et peuvent comporter trois niveaux :

- Mise à jour majeure – Mise à jour contenant des modifications incompatibles avec les versions de plateforme existantes. Vous devrez peut-être modifier votre application pour qu'elle fonctionne correctement sur une nouvelle version majeure. Une mise à jour majeure a un nouveau numéro de version majeure de plateforme.
- Mise à jour mineure : mise à jour dont les modifications sont rétrocompatibles avec les versions existantes de la plateforme dans la plupart des cas. En fonction de votre application, vous devrez peut-être modifier votre application pour qu'elle fonctionne correctement sur une

nouvelle version mineure. Une mise à jour mineure a un nouveau numéro de version mineure de plateforme.

- Mise à jour corrective – Mise à jour consistant en publications de maintenance (correctifs de bogues, mises à jour de sécurité et améliorations de performances) qui sont rétrocompatibles avec une version de plateforme existante. Une mise à jour de correctif a un nouveau numéro de correctif de version de plateforme.

Mises à jour gérées

Fonctionnalité Elastic Beanstalk qui applique automatiquement les mises à jour correctives et mineures au système d'exploitation (OS), à l'environnement d'exécution, au serveur web, au serveur d'applications et aux composants Elastic Beanstalk pour une version de plateforme prise en charge par Elastic Beanstalk. Une mise à jour gérée applique une version de plateforme plus récente dans la même branche de plateforme à votre environnement. Vous pouvez configurer les mises à jour gérées pour n'appliquer que les mises à jour correctives ou les mises à jour mineures et correctives. Vous pouvez également désactiver entièrement les mises à jour gérées.

Pour de plus amples informations, veuillez consulter Mises à jour gérées de la plateforme.

Modèle de responsabilité partagée pour la maintenance de la plateforme Elastic Beanstalk

AWS et nos clients partagent la responsabilité d'atteindre un niveau élevé de sécurité et de conformité des composants logiciels. Ce modèle de responsabilité partagée permet de réduire votre charge opérationnelle.

Pour plus de détails, consultez le modèle de responsabilité AWS partagée.

AWS Elastic Beanstalk vous aide à mettre en œuvre votre version du modèle de responsabilité partagée en fournissant une fonctionnalité de mises à jour gérées. Cette fonction applique automatiquement les mises à jour correctives et mineures correspondant à une version de plateforme prise en charge par Elastic Beanstalk. Si une mise à jour gérée échoue, Elastic Beanstalk vous informe de l'échec pour que vous en teniez compte et que vous preniez des mesures immédiates.

Pour de plus amples informations, veuillez consulter Mises à jour gérées de la plateforme.

De plus, Elastic Beanstalk effectue les opérations suivantes :

- Publie sa stratégie de prise en charge de la plateforme et son calendrier de mise hors service pour les 12 mois prochains.
- Libère les mises à jour correctives, mineures et majeures des composants du système d'exploitation (OS), du moteur d'exécution, du serveur d'applications et du serveur Web généralement dans les 30 jours suivant leur disponibilité. Elastic Beanstalk est responsable de la création de mises à jour des composants Elastic Beanstalk présents sur ses versions de plateforme prises en charge. Toutes les autres mises à jour sont fournies directement par leurs fournisseurs (propriétaires ou communautés).

Nous annonçons toutes les mises à jour de nos plateformes prises en charge dans les notes de mise à jour du guide Notes de mise à jour AWS Elastic Beanstalk . Nous fournissons également une liste de toutes les plateformes prises en charge et de leurs composants, ainsi qu'un historique des plateformes, dans le guide Plateformes AWS Elastic Beanstalk . Pour de plus amples informations, veuillez consulter Plateformes prises en charge et historique des composants.

Vous êtes chargé d'effectuer les tâches suivantes :

- Mettez à jour tous les composants que vous contrôlez (identifiés en tant que client dans le modèle de responsabilité AWS partagée). Il s'agit notamment d'assurer la sécurité de votre application, de vos données et de tous les composants dont votre application a besoin et que vous avez téléchargé.
- Assurez-vous que vos environnements Elastic Beanstalk s'exécutent sur une version de plateforme prise en charge et migrez tout environnement exécuté sur une version de plateforme hors service vers une version prise en charge.
- Si vous utilisez une image machine Amazon (AMI) personnalisée pour votre environnement Elastic Beanstalk, corrigez, gérez et testez votre AMI personnalisée afin qu'elle reste à jour et compatible avec une version prise en charge de la plateforme Elastic Beanstalk. Pour plus d'informations sur la gestion des environnements avec une AMI personnalisée, consultezUtilisation d'une image machine Amazon (AMI) personnalisée dans votre environnement Elastic Beanstalk.
- Résolvez tous les problèmes qui surviennent lors de l'échec des tentatives de mises à jour gérées et réessayez la mise à jour.
- Appliquez vous-même un correctif au système d'exploitation, à l'environnement d'exécution et au serveur web si vous n'avez pas choisi les mises à jour gérées par Elastic Beanstalk. Pour cela, vous pouvez appliquer les mises à jour de plateforme manuellement ou appliquer directement les correctifs aux composants sur toutes les ressources d'environnement concernées.

- Gérez la sécurité et la conformité de tous les AWS services que vous utilisez en dehors d'Elastic Beanstalk conformément AWS au modèle de responsabilité partagée.

Stratégie de prise en charge de la plateforme Elastic Beanstalk

Elastic Beanstalk prend en charge les succursales de plateformes qui continuent de recevoir des mises à jour mineures et correctives de la part de leurs fournisseurs (propriétaires ou communauté). Pour obtenir une définition complète des termes associés, veuillez consulter Glossaire des plateformes Elastic Beanstalk.

Branches de plate-forme mises hors service

Lorsqu'un composant d'une branche de plateforme prise en charge est marqué comme étant en fin de vie (EOL) par son fournisseur, Elastic Beanstalk marque la branche de plateforme comme étant retirée. Les composants d'une branche de plate-forme sont les suivants : système d'exploitation (OS), version du langage d'exécution, serveur d'applications ou serveur Web.

Une fois qu'une branche de la plateforme est marquée comme supprimée, les règles suivantes s'appliquent :

- Elastic Beanstalk cesse de fournir des mises à jour de maintenance, notamment des mises à jour de sécurité.
- Elastic Beanstalk ne fournit plus de support technique aux branches de plateforme abandonnées.
- Elastic Beanstalk ne met plus la branche plateforme à la disposition des nouveaux clients d'Elastic Beanstalk pour les déploiements dans de nouveaux environnements. Il existe une période de grâce de 90 jours à compter de la publication de la date de retrait pour les clients existants dont les environnements actifs s'exécutent sur des branches de plateforme retirées.

> **Note**
>
> Une branche de plateforme retirée ne sera pas disponible dans la console Elastic Beanstalk. Toutefois, il sera disponible via la AWS CLI CLI EB et l'API EB pour les clients disposant d'environnements existants basés sur la branche de plate-forme retirée. Les clients existants peuvent également utiliser les consoles de l'environnement Clone et de l'environnement Rebuild.

Pour obtenir la liste des branches de la plateforme dont la mise hors service est prévue, consultez la rubrique Calendrier de mise hors service des branches de plateforme suivante consacrée au calendrier de la plateforme Elastic Beanstalk.

Pour plus d'informations sur ce à quoi vous attendre en cas de retrait de la branche plateforme de votre environnement, consultezFAQ sur le retrait de la plateforme.

Au-delà de la période de grâce de 90 jours

Notre politique concernant les succursales de plateforme retirées ne supprime pas l'accès aux environnements ni les ressources. Toutefois, les clients existants qui exécutent un environnement Elastic Beanstalk sur une branche de plateforme abandonnée doivent être conscients des risques que cela comporte. De tels environnements peuvent se retrouver dans une situation imprévisible, car Elastic Beanstalk n'est pas en mesure de fournir des mises à jour de sécurité, un support technique ou des correctifs aux succursales de plateforme abandonnées, le fournisseur ayant marqué la fin de vie de leurs composants par le fournisseur.

Par exemple, une vulnérabilité de sécurité néfaste et critique peut apparaître dans un environnement exécuté sur une branche de plateforme retirée. Ou encore, une action de l'API EB peut cesser de fonctionner pour l'environnement si elle devient incompatible avec le service Elastic Beanstalk au fil du temps. La possibilité de ce type de risques augmente avec la durée d'activité d'un environnement basé sur une branche de plateforme retirée. Pour continuer à profiter des améliorations importantes proposées par les fournisseurs des composants en matière de sécurité, de performances et de fonctionnalités, nous vous encourageons vivement à mettre à jour tous vos environnements Elastic Beanstalk vers une version de plateforme prise en charge.

Si votre application rencontre des problèmes lors de son exécution sur une branche de plate-forme abandonnée et que vous ne parvenez pas à la migrer vers une plate-forme prise en charge, vous devrez envisager d'autres solutions. Les solutions de contournement consistent à encapsuler l'application dans une image Docker pour l'exécuter en tant que conteneur Docker. Cela permettrait à un client d'utiliser n'importe laquelle de nos solutions Docker, telles que nos plateformes AL2 Elastic AL2 Beanstalk 023/ Docker, ou d'autres services basés sur Docker tels qu'Amazon ECS ou Amazon EKS. Les alternatives autres que Docker incluent notre AWS CodeDeploy service, qui permet une personnalisation complète des environnements d'exécution que vous souhaitez.

Calendrier de sortie de la plateforme Elastic Beanstalk

Outre la publication mensuelle en cadence des nouvelles versions des branches de la plateforme, notre maintenance des versions inclut également les processus suivants :

- Publication de nouvelles branches de plate-forme : celles-ci introduisent généralement une nouvelle version majeure d'un langage d'exécution, d'un système d'exploitation ou d'un serveur d'applications.
- Retrait des succursales de la plateforme — Nous devons retirer une succursale de la plateforme lorsque l'un de ses composants atteint la fin de vie (EOL). Pour plus d'informations sur notre politique à l'égard des succursales retirées, voir Stratégie de prise en charge de la plateforme Elastic Beanstalk

Rubriques

- Ressources de planification
- Prochaines versions des branches de la plateforme
- Calendrier de mise hors service des branches de plateforme
- Historique des branches de plateforme retirées
- Historique du serveur et du système d'exploitation retirés

Ressources de planification

Les ressources suivantes peuvent vous aider à planifier la maintenance et le support de votre application exécutée sur une plateforme Elastic Beanstalk.

- AWS Elastic Beanstalk Guide des plateformes — Ce guide fournit une liste détaillée des composants pour chacune de nos branches de plateforme. Il fournit également un historique de la plateforme par date de sortie avec les mêmes détails. Ce guide peut vous informer lorsque des composants spécifiques de la branche de votre plateforme sont modifiés. Si votre application commence à se comporter différemment, vous pouvez également faire référence à la date de l'événement dans le guide des plateformes pour voir si des modifications de plate-forme ont pu affecter votre application.
- AWS Elastic Beanstalk Notes de version — Nos notes de publication annoncent toutes les versions mineures et majeures de notre plateforme. Cela inclut les mises à jour mensuelles de notre plateforme, les mises à jour de sécurité, les correctifs et les annonces de départ à la retraite. Vous pouvez vous abonner à nos flux RSS à partir de la documentation des notes de publication.

Prochaines versions des branches de la plateforme

Le tableau suivant répertorie les prochaines filiales de la plateforme Elastic Beanstalk et leur date de sortie cible. Ces dates sont provisoires et sujettes à modification.

Version d'exécution/branche de plate-forme	Système d’exploitation	Date de sortie cible
Python 3.15	Amazon Linux 2023	Novembre 2026
Node.js 26	Amazon Linux 2023	Novembre 2026
.NET 11	Amazon Linux 2023	décembre 2026
PHP 8.6	Amazon Linux 2023	janvier 2027
Ruby 4.1	Amazon Linux 2023	février 2027

Calendrier de mise hors service des branches de plateforme

Le tableau suivant répertorie les succursales de la plateforme Elastic Beanstalk dont la mise hors service est prévue, car certains de leurs composants atteignent leur fin de vie (EOL). Toutes les succursales AL2 basées sur des plateformes ont une date de départ à la retraite au plus tard le 30 juin 2026, date à laquelle Amazon Linux 2 atteindra sa fin de vie. Pour plus d'informations sur Amazon Linux 2, consultez Amazon Linux 2 FAQs.

Pour une liste plus détaillée des branches de plateforme retirées, y compris leurs composants spécifiques, voir les versions de plate-forme retirées dans le guide des AWS Elastic Beanstalk plateformes.

Version d'exécution/branche de plate-forme	Date de départ à la retraite cible	
PHP 8.1 AL2023	31 mars 2026	
PHP 8.1 AL2	31 mars 2026	
Docker AL2	30 juin 2026	

Version d'exécution/branche de plate-forme	Date de départ à la retraite cible
ECS AL2	30 juin 2026
Go 1 AL2	30 juin 2026
Corretto 8 AL2	30 juin 2026
Corretto 11 AL2	30 juin 2026
Corretto 17 AL2	30 juin 2026
Corretto 8 with Tomcat 9 AL2	30 juin 2026
Corretto 11 with Tomcat 9 AL2	30 juin 2026
.NET Core AL2	30 juin 2026
Python 3.9 AL2023	31 juillet 2026
Ruby 3.2 AL2023	31 juillet 2026
Node.js 20 AL2023	31 juillet 2026
IIS 10.0 on Windows Server 2016 (& Core)	30 septembre 2026
PHP 8.2 AL2023	31 mars 2027
.NET 9 AL2023	31 mars 2027
.NET 8 AL2023	31 mars 2027

Historique des branches de plateforme retirées

Les tableaux suivants répertorient les branches de la plateforme Elastic Beanstalk dont le statut est déjà retiré. Vous pouvez consulter un historique détaillé de ces branches de plateforme et de leurs composants dans le guide « Historique des AWS Elastic Beanstalk plateformes ».

Amazon Linux 2023 (AL2023)

Version d'exécution/branche de plate-forme	Date de retrait		
.NET 6 AL2023	8 avril 2025		
Node.js 18 AL2023	11 août 2025		

Amazon Linux (2AL2)

Version d'exécution/branche de plate-forme	Date de retrait		
Corretto 11 with Tomcat 8.5 AL2	10 octobre 2024		
Corretto 8 with Tomcat 8.5 AL2	10 octobre 2024		
Corretto 11 with Tomcat 7 AL2	29 juin 2022		
Corretto 8 with Tomcat 7 AL2	29 juin 2022		
Node.js 18 AL2	11 août 2025		
Node.js 16 AL2	10 octobre 2024		
Node.js 14 AL2	10 octobre 2024		
Node.js 12 AL2	23 décembre 2022		
Node.js 10 AL2	29 juin 2022		
PHP 8.0 AL2	10 octobre 2024		

Version d'exécution/branche de plate-forme	Date de retrait		
PHP 7.4 AL2	9 juin 2023		
PHP 7.3 AL2	29 juin 2022		
PHP 7.2 AL2	29 juin 2022		
Python 3.8 AL2	8 avril 2025		
Python 3.7 AL2	10 octobre 2024		
Ruby 3.0 AL2	10 octobre 2024		
Ruby 2.7 AL2	10 octobre 2024		
Ruby 2.6 AL2	23 décembre 2022		
Ruby 2.5 AL2	29 juin 2022		

AMI Amazon Linux (AL1)

Version d'exécution/branche de plate-forme	Date de retrait		
Single Container Docker	18 juillet 2022		
Multicontainer Docker	18 juillet 2022		
Preconfigured Docker - GlassFish 5.0 with Java 8	18 juillet 2022		

Version d'exécution/branche de plate-forme	Date de retrait
Go 1	18 juillet 2022
Java 8	18 juillet 2022
Java 7	18 juillet 2022
Java 8 with Tomcat 8.5	18 juillet 2022
Java 7 with Tomcat 7	18 juillet 2022
Node.js	18 juillet 2022
PHP 7.2 - 7.3	18 juillet 2022
Python 3.6	18 juillet 2022
Ruby 2,4, 2.5, 2.6 with Passenger	18 juillet 2022
Ruby 2.4, 2.5, 2.6 with Puma	18 juillet 2022
Go 1.3–1.10	31 octobre 2020
Java 6	31 octobre 2020
Node.js 4.x–8.x	31 octobre 2020
PHP 5.4–5.6	31 octobre 2020
PHP 7.0–7.1	31 octobre 2020
Python 2.6, 2.7, 3.4	31 octobre 2020

Version d'exécution/branche de plate-forme	Date de retrait		
Ruby 1.9.3	31 octobre 2020		
Ruby 2.0–2.3	31 octobre 2020		

> **Note**
>
> Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour de plus amples informations, veuillez consulter FAQ sur le retrait de la plateforme.

Windows Server

Version d'exécution/branche de plate-forme	Date de retrait		
IIS 8.5 s'exécutant sur Windows Server (et Core) 2012 R2 64 bits	4 décembre 2023		
IIS 8.5 s'exécutant sur Windows Server (& Core) 2012 R2 64 bits version 0.1.0	29 juin 2022		
IIS 8.5 s'exécutant sur Windows Server (& Core) 2012 R2 64 bits version 1.2.0	29 juin 2022		

Version d'exécution/branche de plate-forme	Date de retrait		
IIS 10.0 s'exécutant sur Windows Server 2016 (& Core) 64 bits version 1.2.0	29 juin 2022		
IIS 8 s'exécutant sur la branche de plateforme Windows Server 2012 R1 64 bits	22 juin 2022		
IIS 8 s'exécutant sur Windows Server 2012 R1 64 version 0.1.0	22 juin 2022		
IIS 8 s'exécutant sur Windows Server 2012 R1 64 version 1.2.0	22 juin 2022		

> **Note**
>
> Pour plus d'informations sur le retrait des branches de la plateforme Windows 2012 R2, voir Mise hors service des branches de plateforme Windows Server 2012 R2 dans les Notes de mise à jour AWS Elastic Beanstalk .

Historique du serveur et du système d'exploitation retirés

Les tableaux suivants fournissent un historique des systèmes d'exploitation, des serveurs d'applications et des serveurs Web qui ne sont plus pris en charge par les plateformes Elastic Beanstalk. Toutes les branches de la plateforme qui utilisaient ces composants sont désormais supprimées. Les dates reflètent la date de retrait de la dernière branche de la plateforme Elastic Beanstalk qui incluait le composant.

Systèmes d'exploitation

Version du système d'exploitation	Date de mise hors service de la plateforme		
Windows Server 2012 R2 running IIS 8.5	4 décembre 2023		
Windows Server Core 2012 R2 running IIS 8.5	4 décembre 2023		
AMI Amazon Linux (AL1)	18 juillet 2022		
Windows Server 2(	22 juin 2022		
Windows Server 2(	28 octobre 2019		

Serveurs d’applications

Version du serveur d'applications	Date de mise hors service de la plateforme		
Tomcat 7	29 juin 2022 pour les plateformes Amazon Linux (2AL2)		

Version du serveur d'applications	Date de mise hors service de la plateforme		
	18 juillet 2022 pour les plateformes Amazon Linux AMI (AL1)		
Tomcat 8	31 octobre 2020		
Tomcat 6	31 octobre 2020		

Serveurs Web

Version du serveur web	Date de mise hors service de la plateforme		
IIS 8 exécutant Windows Server 64 bits	22 juin 2022		
Serveur HTTP Apache 2.2	31 octobre 2020		
Nginx 1.12.2	31 octobre 2020		

Plateformes prises en charge par Elastic Beanstalk

AWS Elastic Beanstalk fournit une variété de plateformes sur lesquelles vous pouvez créer vos applications. Vous concevez votre application web sur l'une de ces plateformes. Elastic Beanstalk déploie votre code sur la version de la plateforme que vous avez sélectionnée pour créer un environnement d'application actif.

Elastic Beanstalk fournit les ressources nécessaires pour exécuter votre application, y compris une ou plusieurs instances Amazon. EC2 La pile logicielle exécutée sur les EC2 instances Amazon dépend de la version de plateforme spécifique que vous avez sélectionnée pour votre environnement.

Le nom de la pile de solutions pour une branche de plate-forme

Vous pouvez utiliser le nom de la pile de solutions pour une version de branche de plate-forme donnée afin de lancer un environnement avec l'EB CLI, l'API Elastic Beanstalk ou la CLI.AWS Le guide AWS Elastic Beanstalk des plateformes répertorie le nom de la pile de solutions sous la version de la branche de plate-forme dans les sections Plateformes prises en charge par Elastic Beanstalk et historique de la plate-forme.

Pour récupérer tous les noms de pile de solutions que vous pouvez utiliser pour créer un environnement, utilisez l'ListAvailableSolutionStacksAPI ou `aws elasticbeanstalk list-available-solution-stacks`la AWS CLI.

Vous pouvez personnaliser et configurer le logiciel dont dépend votre application dans votre plateforme. Pour en savoir plus, veuillez consulter Personnalisation du logiciel sur des serveurs Linux et Personnalisation du logiciel sur des serveurs Windows. Des notes de mise à jour détaillées sont disponibles pour les mises à jour récentes dans le document suivant : AWS Elastic Beanstalk Notes de mise à jour.

Plateformes prises en charge et historique des composants

Le guide AWS Elastic Beanstalk des plateformes répertorie toutes les versions actuelles des branches de plate-forme dans la section Plateformes prises en charge par Elastic Beanstalk. Le guide des plateformes répertorie également un historique des plateformes pour chaque plate-forme, qui inclut une liste des versions précédentes de la plateforme des succursales. Pour consulter l'historique de chaque plateforme, sélectionnez l'un des liens suivants.

- Docker
- Go
- Java SE
- Tomcat (exécutant Java SE)
- .NET Core sous Linux
- .NET sous Windows Server
- Node.js
- PHP
- Python
- Ruby

Plateformes Linux Elastic Beanstalk

Les plateformes Linux Elastic Beanstalk fournissent de nombreuses fonctionnalités prêtes à l'emploi. Vous pouvez étendre les plateformes de plusieurs façons pour prendre en charge votre application. Pour en savoir plus, consultez the section called “Extension des plateformes Linux”.

La plupart des plateformes prises en charge par Elastic Beanstalk se basent sur le système d'exploitation Linux. Plus précisément, ces plateformes sont basées sur Amazon Linux, une distribution Linux fournie par AWS. Les plateformes Linux Elastic Beanstalk utilisent des instances Amazon Elastic Compute Cloud EC2 (Amazon), qui exécutent Amazon Linux.

Rubriques

- Versions d'Amazon Linux prises en charge
- Liste des plateformes Linux Elastic Beanstalk
- Flux de travail (workflow) de déploiement d'instance
- Flux de déploiement d'instance pour ECS s'exécutant sur Amazon Linux 2 et versions ultérieures
- Outils de script de plateforme pour vos environnements Elastic Beanstalk

Versions d'Amazon Linux prises en charge

AWS Elastic Beanstalk prend en charge les plateformes basées sur Amazon Linux 2 et Amazon Linux 2023.

Pour plus d'informations sur Amazon Linux 2 et Amazon Linux 2023, consultez :

- Amazon Linux 2 — Amazon Linux dans le guide de EC2 l'utilisateur Amazon.
- Amazon Linux 2023 – Qu'est-ce qu'Amazon Linux 2023 ? dans le Guide de l'utilisateur Amazon Linux 2023

Pour de plus amples informations sur les versions de plateforme prises en charge, veuillez consulter Plateformes prises en charge par Elastic Beanstalk.

> **Note**
>
> Vous pouvez migrer votre application depuis une branche AL1 Elastic AL2 Beanstalk ou une branche de plateforme vers AL2 une branche de plateforme 023 équivalente. Pour de plus

amples informations, veuillez consulter Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Amazon Linux 2023

AWS a annoncé la disponibilité générale d'Amazon Linux 2023 en mars 2023. Le Guide de l'utilisateur Amazon Linux 2023 résume les principales différences entre Amazon Linux 2 et Amazon Linux 2023. Pour plus d'informations, consultez Comparaison entre Amazon Linux 2 et Amazon Linux 2023 (français non garanti) dans le guide de l'utilisateur.

Il existe un degré élevé de compatibilité entre les plateformes Elastic Beanstalk Amazon Linux 2 et Amazon Linux 2023. Il y a cependant quelques différences à noter :

- Version 1 du service de métadonnées d'instance (IMDSv1) : le paramètre de l'IMDSv1option Disable est défini par défaut `true` sur les plateformes AL2 023. Par défaut, c'est `false` sur AL2 les plateformes.
- outil d'instance pkg-repo — L'pkg-repooutil n'est pas disponible pour les environnements exécutés sur AL2 les plateformes 023. Toutefois, vous pouvez toujours appliquer manuellement les mises à jour du package et du système d'exploitation à une instance AL2 023. Pour plus d'informations, consultez Gestion des packages et des mises à jour du système d'exploitation (français non garanti) dans le Guide de l'utilisateur Amazon Linux 2023.
- HTTPd Configuration Apache — Le `httpd.conf` fichier Apache pour les plateformes AL2 023 comporte des paramètres de configuration différents de ceux des AL2 plateformes suivantes :
  - Interdire l'accès à l'ensemble du système de fichiers du serveur par défaut. Ces paramètres sont décrits dans la section Protection par défaut des fichiers du serveur de la page Conseils sur la sécurité du site web d'Apache.
  - Refusez l'accès à la configuration de `.htaccess` dans tous les annuaires, à l'exception de ceux spécifiquement activés. Ce paramètre est décrit dans la section Protection de la configuration du système de la page Conseils sur la sécurité du site web d'Apache. La page Tutoriel du serveur HTTP Apache : fichiers .htaccess indique que ce paramètre peut contribuer à améliorer les performances.
  - Refuser l'accès aux fichiers portant le modèle de nom `.ht*`. Ce paramètre empêche les clients web de visualiser les fichiers `.htaccess` et `.htpasswd`.

Vous pouvez modifier les paramètres de configuration ci-dessus en fonction de votre environnement. Pour de plus amples informations, veuillez consulter Configuration d'Apache HTTPD.

- Prise en charge des variables d'environnement multilignes — Les plateformes AL2 023 prennent en charge les valeurs multilignes pour les variables d'environnement et les secrets dans les configurations de service Systemd. Les plateformes Amazon Linux 2 ne prennent pas en charge les valeurs de variables d'environnement multilignes. Cette amélioration vous permet d'utiliser des secrets multilignes et des valeurs de configuration sur les plateformes AL2 023. Pour plus d'informations sur l'utilisation des variables d'environnement et des secrets, consultezthe section called “Valeurs multilignes”.
- CloudWatch transfert de journal personnalisé — L'agent CloudWatch Logs obsolète (`awslogs`package) n'est pas disponible sur les plateformes AL2 023. Si vous avez des configurations personnalisées de transfert de journal qui installent et utilisent l'`awslogs`agent obsolète, vous devez mettre à jour vos fichiers de configuration pour utiliser l' CloudWatch agent unifié lors de la migration d'Amazon Linux 2 vers 023. AL2 Pour de plus amples informations, veuillez consulter the section called “Streaming de fichiers journaux personnalisés”.

Différences spécifiques à la plateforme

Outre les différences de système d'exploitation de base, il existe des différences spécifiques à la plate-forme entre les plateformes d'exécution Amazon Linux 2 et AL2 023 :

- Branchement de plate-forme .NET — La stratégie de branchement de la plate-forme .NET diffère entre Amazon Linux 2 et AL2 023. Sur Amazon Linux 2, la plate-forme .NET Core gère une fenêtre rotative des versions principales de .NET au sein d'une seule branche de plate-forme. Le AL2 023, chaque branche de plate-forme est épinglée à une version majeure de .NET spécifique (par exemple, .NET 9, .NET 10).

  Si vous déployez des applications dépendantes du framework (applications qui s'appuient sur le runtime .NET installé sur la plate-forme), vous devez sélectionner une branche de plate-forme correspondant à la version .NET cible de votre application. Si vous déployez des applications autonomes (applications qui regroupent leur propre environnement d'exécution .NET), vous pouvez utiliser n'importe quelle branche de la plate-forme .NET AL2 023, quelle que soit la version .NET de votre application, car celle-ci ne dépend pas du runtime installé sur la plate-forme. Pour de plus amples informations, veuillez consulter the section called “Regroupement d'applications”.

- Sélection de la version de Node.js — La plateforme Node.js sur Amazon Linux 2 permet de spécifier une version Node.js dans le `package.json` fichier de votre application. La plateforme Node.js sur AL2 023 ne prend pas en charge cette fonctionnalité. Vous devez utiliser la version par défaut de Node.js fournie par la branche de plateforme. Pour plus d'informations sur la gestion des versions de Node.js, consultezthe section called “Configuration des dépendances”.
- Version du serveur Ruby Puma — La plateforme Ruby sur Amazon Linux 2 ignore la version Puma spécifiée dans le `Gemfile.lock` fichier de votre application et utilise la version Puma par défaut de la plateforme. La plateforme Ruby du AL2 023 respecte la version Puma spécifiée dans le `Gemfile.lock` cas où elle est présente. Si aucune version n'est spécifiée, la plateforme installe la version Puma par défaut de la plateforme.
- Disponibilité des packages PHP — Certains packages disponibles sur les plateformes PHP Amazon Linux 2 ne sont pas disponibles sur les plateformes PHP AL2 023 :
  - Packages clients MySQL — Les packages clients de ligne de `mysql-devel` commande `mysql` et de ligne de commande ne sont pas installés sur les plateformes PHP AL2 023. Si votre application nécessite une connectivité à une base de données MySQL, utilisez le PHP `mysqli` ou des `pdo_mysql` extensions, disponibles sur les deux plateformes.
  - Outils Compass et Ruby — Les `rubygems` packages `ruby-devel` et pour le support du framework CSS Compass ne sont pas installés sur les plateformes PHP AL2 023. Compass est obsolète. Envisagez d'utiliser des outils de prétraitement CSS modernes comme alternative.
- Outils de contrôle de version Go — Le système de contrôle de version Bazaar (`bzr`) n'est pas disponible sur les plateformes AL2 023 Go. Bazaar est obsolète et n'est pas inclus dans le référentiel de packages AL2 023. Utilisez plutôt Git, Mercurial ou Subversion pour le contrôle de version, qui sont tous disponibles sur les plateformes AL2 023 Go.

Liste des plateformes Linux Elastic Beanstalk

La liste suivante fournit les plateformes Linux prises en charge par Elastic Beanstalk pour les différents langages de programmation et les conteneurs Docker. Elastic Beanstalk propose des plateformes basées sur Amazon Linux 2 et Amazon Linux 2023 pour toutes ces plateformes. Pour en savoir plus sur une plateforme, sélectionnez le lien correspondant.

- Docker (et Docker ECS)
- Go
- Tomcat (exécutant Java SE)
- Java SE

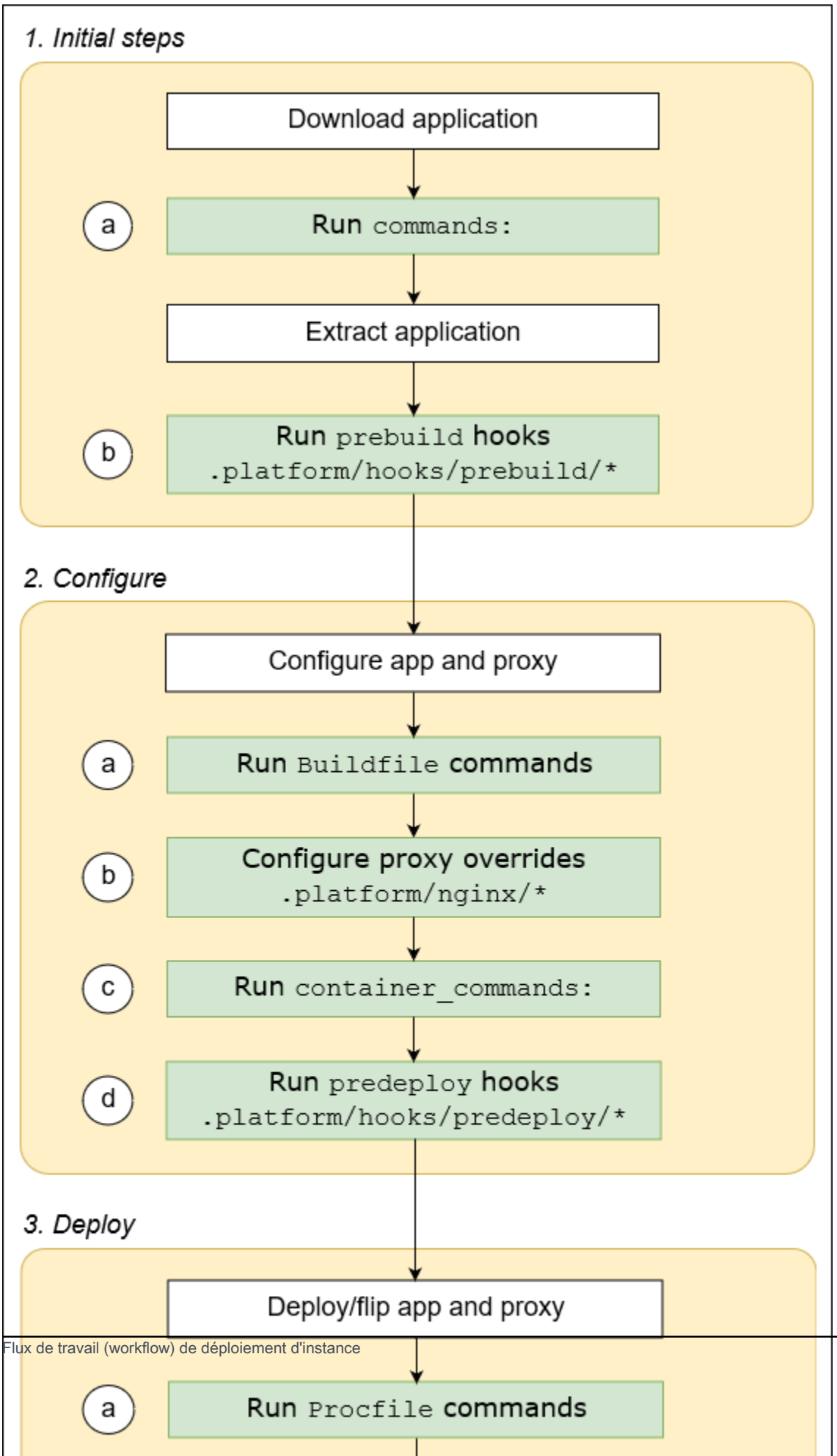
- .NET Core sous Linux
- Node.js
- PHP
- Python
- Ruby

Flux de travail (workflow) de déploiement d'instance

> **Note**
>
> Les informations de cette section ne s'appliquent pas aux branches de plateforme ECS s'exécutant sur Amazon Linux 2 et Amazon Linux 2023. Pour plus d'informations, voir la section suivante Flux de déploiement d'instance pour ECS s'exécutant sur Amazon Linux 2 et versions ultérieures.

Avec de nombreuses façons d'étendre la plateforme de votre environnement, il est utile de savoir ce qui se passe chaque fois qu'Elastic Beanstalk alloue une instance ou exécute un déploiement sur une instance. Le diagramme suivant illustre l'ensemble du workflow de déploiement. Elles décrivent les différentes phases d'un déploiement et les étapes suivies par Elastic Beanstalk au cours de chaque phase.

> **Remarques**
>
> - Le diagramme ne représente pas l'ensemble complet des étapes suivies par Elastic Beanstalk sur les instances d'environnement au cours du déploiement. Nous fournissons ce diagramme à titre d'illustration, pour vous indiquer l'ordre et le contexte de l'exécution de vos personnalisations.
> - Par souci de simplicité, le diagramme ne mentionne que les sous-répertoires hook `.platform/hooks/*` (pour les déploiements d'applications), et non les sous-répertoires hook `.platform/confighooks/*` (pour les déploiements de configuration). Les hooks dans ces derniers sous-répertoires s'exécutent exactement au cours des mêmes étapes que les hooks dans les sous-répertoires correspondants indiqués dans le diagramme.


1. Initial steps
Download application
a
Run commands:
Extract application
b
Run prebuild hooks
.platform/hooks/prebuild/*
2. Configure
Configure app and proxy
a
Run Buildfile commands
b
Configure proxy overrides
.platform/nginx/*
c
Run container_commands:
d
Run predeploy hooks
.platform/hooks/predeploy/*
3. Deploy
Deploy/flip app and proxy
a
Run Procfile commands

La liste suivante détaille les phases et les étapes de déploiement.

1. Étapes initiales

   Elastic Beanstalk télécharge et extrait votre application. Après chacune de ces étapes, Elastic Beanstalk exécute l'une des étapes d'extensibilité.

   a. Exécute les commandes présentes dans la section commands: de tout fichier de configuration.

   b. Exécute tous les fichiers exécutables trouvés dans le répertoire `.platform/hooks/prebuild` de votre bundle source (`.platform/confighooks/prebuild` pour un déploiement de configuration).

2. Configuration

   Elastic Beanstalk configure votre application et le serveur proxy.

   a. Exécute les commandes trouvées dans le bundle de fichiers source `Buildfile`.

   b. Copie vos fichiers de configuration proxy personnalisés, le cas échéant, du répertoire `.platform/nginx` de votre bundle de fichiers source vers leur emplacement d'exécution.

   c. Exécute les commandes de la section container_commands: de tout fichier de configuration.

   d. Exécute tous les fichiers exécutables trouvés dans le répertoire `.platform/hooks/predeploy` de votre bundle source (`.platform/confighooks/predeploy` pour un déploiement de configuration).

3. Déploiement

   Elastic Beanstalk déploie et exécute votre application et le serveur proxy.

   a. Exécute la commande trouvée dans le fichier `Procfile` de votre bundle de fichiers source.

   b. Exécute ou réexécute le serveur proxy avec vos fichiers de configuration proxy personnalisés, le cas échéant.

   c. Exécute tous les fichiers exécutables trouvés dans le répertoire `.platform/hooks/postdeploy` de votre bundle source (`.platform/confighooks/postdeploy` pour un déploiement de configuration).

Flux de déploiement d'instance pour ECS s'exécutant sur Amazon Linux 2 et versions ultérieures

La section précédente décrit les fonctions d'extensibilité prises en charge pendant les phases du flux de déploiement d'applications. Il existe certaines différences pour les branches de la plateforme

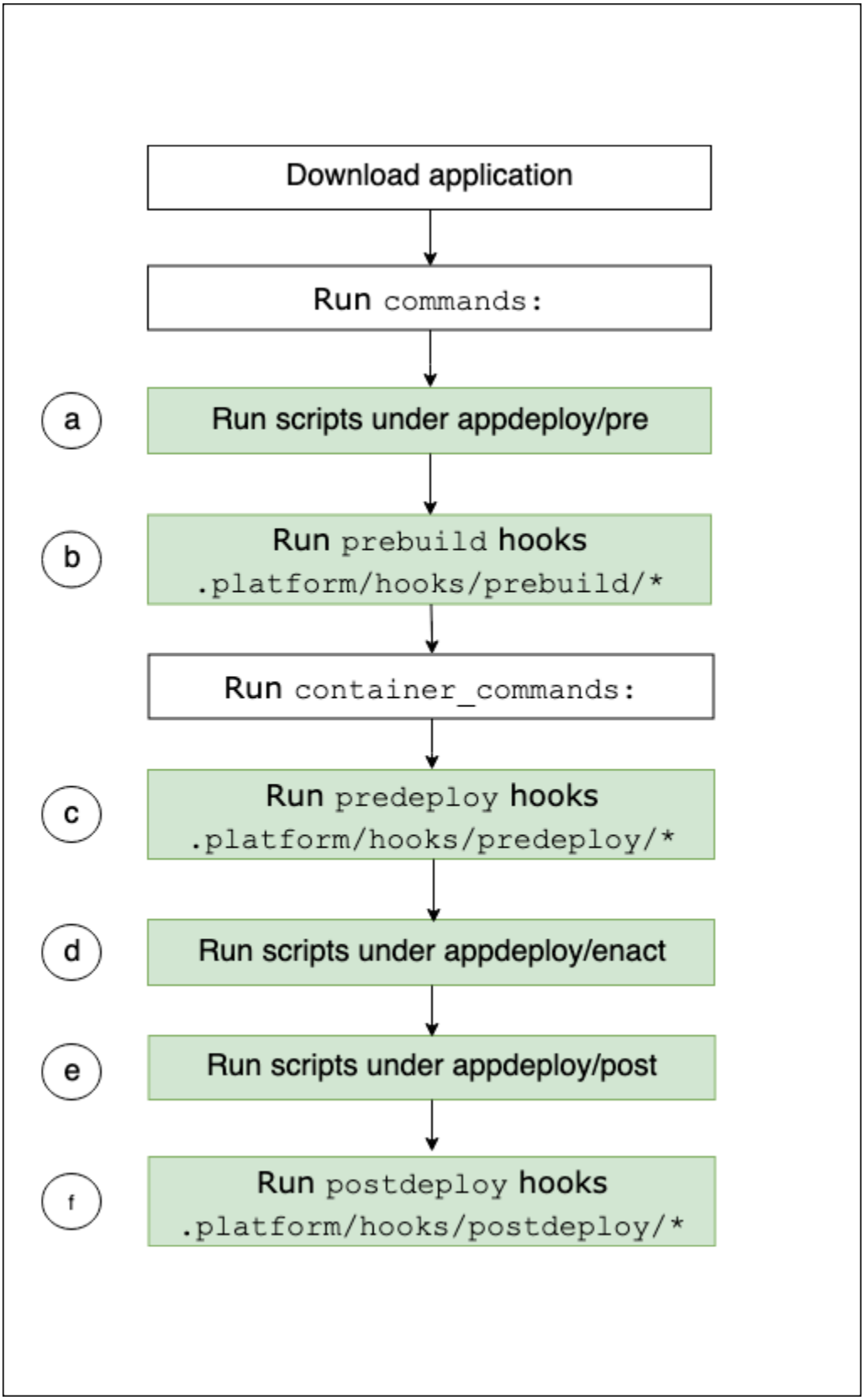
Docker ECS s'exécutant sur Amazon Linux 2 et versions ultérieures. Cette section explique comment ces concepts s'appliquent à cette branche de plateforme spécifique.

Avec de nombreuses façons d'étendre la plateforme de votre environnement, il est utile de savoir ce qui se passe chaque fois qu'Elastic Beanstalk alloue une instance ou exécute un déploiement sur une instance. Le diagramme suivant illustre l'ensemble du flux de déploiement pour un environnement basé sur les branches de plateforme ECS s'exécutant sur Amazon Linux 2 et ECS s'exécutant sur Amazon Linux 2023. Elles décrivent les différentes phases d'un déploiement et les étapes suivies par Elastic Beanstalk au cours de chaque phase.

Contrairement au flux décrit dans la section précédente, la phase de configuration du déploiement ne prend pas en charge les fonctions d'extensibilité suivantes : commandes `Buildfile`, commandes `Procfile`, configuration de proxy inverse.

Remarques

- Le diagramme ne représente pas l'ensemble complet des étapes suivies par Elastic Beanstalk sur les instances d'environnement au cours du déploiement. Nous fournissons ce diagramme à titre d'illustration, pour vous indiquer l'ordre et le contexte de l'exécution de vos personnalisations.
- Par souci de simplicité, le diagramme ne mentionne que les sous-répertoires hook `.platform/hooks/*` (pour les déploiements d'applications), et non les sous-répertoires hook `.platform/confighooks/*` (pour les déploiements de configuration). Les hooks dans ces derniers sous-répertoires s'exécutent exactement au cours des mêmes étapes que les hooks dans les sous-répertoires correspondants indiqués dans le diagramme.



La liste suivante détaille les étapes du flux de déploiement.

a. Exécute tous les fichiers exécutables trouvés dans le répertoire `appdeploy/pre` sous `EBhooksDir`.
b. Exécute tous les fichiers exécutables trouvés dans le répertoire `.platform/hooks/prebuild` de votre bundle source (`.platform/confighooks/prebuild` pour un déploiement de configuration).
c. Exécute tous les fichiers exécutables trouvés dans le répertoire `.platform/hooks/predeploy` de votre bundle source (`.platform/confighooks/predeploy` pour un déploiement de configuration).
d. Exécute tous les fichiers exécutables trouvés dans le répertoire `appdeploy/enact` sous `EBhooksDir`.
e. Exécute tous les fichiers exécutables trouvés dans le répertoire `appdeploy/post` sous `EBhooksDir`.
f. Exécute tous les fichiers exécutables trouvés dans le répertoire `.platform/hooks/postdeploy` de votre bundle source (`.platform/confighooks/postdeploy` pour un déploiement de configuration).

La référence à `EBhooksDir` représente le chemin d'accès au répertoire des hooks de la plateforme. Pour récupérer le nom du chemin d'accès au répertoire, utilisez l'outil de script get-config sur la ligne de commande de votre instance d'environnement, comme illustré :

```
$ /opt/elasticbeanstalk/bin/get-config platformconfig -k EBhooksDir
```

Outils de script de plateforme pour vos environnements Elastic Beanstalk

Cette rubrique décrit les outils AWS Elastic Beanstalk destinés aux environnements utilisant les plateformes Amazon Linux. Les outils se trouvent sur les EC2 instances Amazon des environnements Elastic Beanstalk.

get-config

Utilisez l'`get-config`outil pour récupérer les valeurs des variables d'environnement en texte brut et d'autres informations relatives à la plate-forme et à l'instance. Cet outil est disponible dans `/opt/elasticbeanstalk/bin/get-config`.

Commandes get-config

Chaque commande d'outil `get-config` renvoie un type spécifique d'informations. Utilisez la syntaxe suivante pour exécuter les commandes de n'importe quel outil.

```
$ /opt/elasticbeanstalk/bin/get-config command [ options ]
```

L'exemple suivant exécute uniquement la commande `environment`.

```
$ /opt/elasticbeanstalk/bin/get-config environment -k PORT
```

Selon la commande et les options que vous choisissez, l'outil renvoie un objet (JSON ou YAML) avec des paires clé-valeur, ou une seule valeur.

Vous pouvez effectuer un test `get-config` en utilisant SSH pour vous connecter à une EC2 instance de votre environnement Elastic Beanstalk.

> **Note**
>
> Lorsque vous exécutez `get-config` pour des tests, certaines commandes peuvent nécessiter des privilèges utilisateur root pour accéder aux informations sous-jacentes. Si vous obtenez une erreur d'autorisation d'accès, exécutez à nouveau la commande sous `sudo`. Vous n'avez pas besoin d'ajouter `sudo` lorsque vous utilisez l'outil dans les scripts que vous déployez dans votre environnement. Elastic Beanstalk exécute tous vos scripts en tant qu'utilisateur racine.

Les sections suivantes décrivent les commandes des outils.

optionsettings – Options de configuration

La commande `get-config optionsettings` renvoie un objet répertoriant les options de configuration définies sur l'environnement et utilisées par la plateforme sur les instances d'environnement. Elles sont organisées par espace de noms.

```
$ /opt/elasticbeanstalk/bin/get-config optionsettings
{"aws:elasticbeanstalk:application:environment":
{"JDBC_CONNECTION_STRING":""},"aws:elasticbeanstalk:container:tomcat:jvmoptions":{"JVM
 Options":"","Xms":"256m","Xmx":"256m"},"aws:elasticbeanstalk:environment:proxy":
{"ProxyServer":"nginx","StaticFiles":
[""]},"aws:elasticbeanstalk:healthreporting:system":
```

```
{"SystemType":"enhanced"},"aws:elasticbeanstalk:hostmanager":
{"LogPublicationControl":"false"}}
```

Pour renvoyer une option de configuration spécifique, utilisez l'option `--namespace` (`-n`) pour spécifier un espace de noms, et l'option `--option-name` (`-o`) pour spécifier un nom d'option.

```
$ /opt/elasticbeanstalk/bin/get-config optionsettings -
n aws:elasticbeanstalk:container:php:phpini -o memory_limit
256M
```

environment – Propriétés de l'environnement

La `get-config environment` commande renvoie un objet contenant une liste de propriétés d'environnement, y compris celles configurées par l'utilisateur et fournies par Elastic Beanstalk. Les propriétés configurées par l'utilisateur sont définies dans la console sous forme de texte brut ou avec l'espace de noms aws:elasticbeanstalk:application:environment des options de configuration.

```
$ /opt/elasticbeanstalk/bin/get-config environment
{"JDBC_CONNECTION_STRING":"","RDS_PORT":"3306","RDS_HOSTNAME":"anj9aw1b0tbj6b.cijbpanmxz5u.us-
west-2.rds.amazonaws.com","RDS_USERNAME":"testusername","RDS_DB_NAME":"ebdb","RDS_PASSWORD":"te
```

Par exemple, Elastic Beanstalk fournit des propriétés d'environnement pour la connexion à une instance de base de données Amazon RDS intégrée (par exemple, `RDS_HOSTNAME`). Ces propriétés de connexion RDS apparaissent dans la sortie de `get-config environment`. Cependant, ils n'apparaissent pas dans le rendu final de `get-config optionsettings`. Cela est dû au fait qu'ils n'étaient pas définis dans les options de configuration.

Pour renvoyer une propriété d'environnement spécifique, utilisez l'option `--key` (`-k`) pour spécifier une propriété clé.

```
$ /opt/elasticbeanstalk/bin/get-config environment -k TESTPROPERTY
testvalue
```

> **Note**
>
> L'`get-config`outil ne peut pas récupérer les variables d'environnement qui stockent des secrets. Pour plus d'informations sur la façon de récupérer par programmation des valeurs à partir de magasins secrets ou de paramètres, consultez Utilisation de Secrets Manager ou. Utilisation du Systems Manager Parameter Store

container – Valeurs de configuration sur instance

La commande `get-config container` renvoie un objet qui répertorie les valeurs de configuration de la plateforme et de l'environnement pour les instances d'environnement.

L'exemple suivant illustre la sortie de la commande dans un environnement Amazon Linux 2 Tomcat.

```
$ /opt/elasticbeanstalk/bin/get-config container
{"common_log_list":["/var/log/eb-engine.log","/var/log/eb-
hooks.log"],"default_log_list":["/var/log/nginx/access.log","/var/log/nginx/
error.log"],"environment_name":"myenv-1da84946","instance_port":"80","log_group_name_prefix":"/
aws/elasticbeanstalk","proxy_server":"nginx","static_files":
[""],"xray_enabled":"false"}
```

Pour renvoyer la valeur d'une clé spécifique, utilisez l'option `--key` (`-k`) pour spécifier la clé.

```
$ /opt/elasticbeanstalk/bin/get-config container -k environment_name
myenv-1da84946
```

addons – Valeurs de configuration du module complémentaire

La commande `get-config addons` renvoie un objet qui contient des informations de configuration des modules complémentaires d'environnement. Utilisez-la pour récupérer la configuration d'une base de données Amazon RDS qui est associée à l'environnement.

```
$ /opt/elasticbeanstalk/bin/get-config addons
{"rds":{"Description":"RDS Environment variables","env":
{"RDS_DB_NAME":"ebdb","RDS_HOSTNAME":"ea13k2wimu1dh8i.c18mnpu5rwvg.us-
east-2.rds.amazonaws.com","RDS_PASSWORD":"password","RDS_PORT":"3306","RDS_USERNAME":"user"}}}
```

Vous pouvez restreindre le résultat de deux façons. Pour récupérer les valeurs d'un module complémentaire spécifique, utilisez l'option `--add-on` (`-a`) pour spécifier le nom du module complémentaire.

```
$ /opt/elasticbeanstalk/bin/get-config addons -a rds
{"Description":"RDS Environment variables","env":
{"RDS_DB_NAME":"ebdb","RDS_HOSTNAME":"ea13k2wimu1dh8i.c18mnpu5rwvg.us-
east-2.rds.amazonaws.com","RDS_PASSWORD":"password","RDS_PORT":"3306","RDS_USERNAME":"user"}}
```

Pour renvoyer la valeur d'une clé spécifique dans un module complémentaire, ajoutez l'option `--key` (`-k`) pour spécifier la clé.

```
$ /opt/elasticbeanstalk/bin/get-config addons -a rds -k RDS_DB_NAME
ebdb
```

platformconfig – Valeurs de configuration constantes

La commande `get-config platformconfig` renvoie un objet qui contient des informations de configuration de la plateforme qui sont constantes pour la version de la plateforme. La sortie est la même dans tous les environnements qui s'exécutent la même version de plateforme. L'objet de sortie de la commande comporte deux objets incorporés :

- `GeneralConfig` : contient des informations qui sont constantes sur les dernières versions de toutes les branches de plateforme Amazon Linux 2 et Amazon Linux 2023.
- `PlatformSpecificConfig` – Contient des informations qui sont constantes pour la version de la plateforme et qui lui sont spécifiques.

L'exemple suivant montre la sortie de la commande sur un environnement qui utilise Tomcat 8.5 exécutant la branche plateforme Corretto 11.

```
$ /opt/elasticbeanstalk/bin/get-config platformconfig
{"GeneralConfig":{"AppUser":"webapp","AppDeployDir":"/var/app/
current/","AppStagingDir":"/var/app/
staging/","ProxyServer":"nginx","DefaultInstancePort":"80"},"PlatformSpecificConfig":
{"ApplicationPort":"8080","JavaVersion":"11","TomcatVersion":"8.5"}}
```

Pour renvoyer la valeur d'une clé spécifique, utilisez l'option `--key` (`-k`) pour spécifier la clé. Ces clés sont uniques sur les deux objets incorporés. Vous n'avez pas besoin de spécifier l'objet qui contient la clé.

```
$ /opt/elasticbeanstalk/bin/get-config platformconfig -k AppStagingDir
/var/app/staging/
```

Options de sortie get-config

Utilisez l'option `--output` pour spécifier le format de l'objet en sortie. Les valeurs valides sont `JSON` (par défaut) et `YAML`. Il s'agit d'une option globale. Vous devez le spécifier avant le nom de la commande.

L'exemple suivant renvoie des valeurs d'option de configuration au format YAML.

```
$ /opt/elasticbeanstalk/bin/get-config --output YAML optionsettings
aws:elasticbeanstalk:application:environment:
  JDBC_CONNECTION_STRING: ""
aws:elasticbeanstalk:container:tomcat:jvmoptions:
  JVM Options: ""
  Xms: 256m
  Xmx: 256m
aws:elasticbeanstalk:environment:proxy:
  ProxyServer: nginx
  StaticFiles:
        - ""
aws:elasticbeanstalk:healthreporting:system:
  SystemType: enhanced
aws:elasticbeanstalk:hostmanager:
  LogPublicationControl: "false"
```

pkg-repo

> **Note**
>
> L'outil `pkg-repo` n'est pas disponible pour les environnements basés sur les plateformes Amazon Linux 2023. Toutefois, vous pouvez appliquer manuellement les mises à jour du package et du système d'exploitation à une instance AL2 023. Pour plus d'informations, consultez Gestion des packages et des mises à jour du système d'exploitation (français non garanti) dans le Guide de l'utilisateur Amazon Linux 2023

Dans certaines circonstances urgentes, vous devrez peut-être mettre à jour vos EC2 instances Amazon avec un correctif de sécurité Amazon Linux 2 qui n'a pas encore été publié avec les versions requises de la plateforme Elastic Beanstalk. Vous ne pouvez pas effectuer de mise à jour manuelle sur vos environnements Elastic Beanstalk par défaut. En effet, les versions de la plateforme sont verrouillées sur une version spécifique du référentiel Amazon Linux 2. Ce verrou garantit que les instances exécutent des versions logicielles compatibles et prises en charge. Pour les cas urgents, l'outil `pkg-repo` permet une solution de contournement pour mettre à jour manuellement les packages yum sur Amazon Linux 2 si vous devez les installer sur un environnement avant qu'ils ne soient publiés dans une nouvelle version de la plateforme Elastic Beanstalk.

L'outil `pkg-repo` sur les plateformes Amazon Linux 2 permet de déverrouiller le référentiels de paquets `yum`. Vous pouvez ensuite effectuer manuellement une commande yum update pour un

correctif de sécurité. Inversement, vous pouvez suivre la mise à jour à l'aide de l'outil pour verrouiller les référentiels de paquets yum afin d'éviter d'autres mises à jour. L'`pkg-repo`outil est disponible dans le `/opt/elasticbeanstalk/bin/pkg-repo` répertoire de toutes les EC2 instances de vos environnements Elastic Beanstalk.

Les modifications effectuées à l'aide de l'`pkg-repo`outil sont effectuées uniquement sur l' EC2 instance sur laquelle l'outil est utilisé. Elles n'affectent pas d'autres instances et n'empêchent pas les futures mises à jour de l'environnement. Les exemples décrits plus loin dans cette rubrique expliquent comment appliquer les changements à toutes les instances en appelant les commandes `pkg-repo` à partir de scripts et de fichiers de configuration.

> **Warning**
>
> Nous vous déconseillons cet outil pour la plupart des utilisateurs. Toute modification manuelle appliquée à une version de plateforme déverrouillée est considérée hors bande. Cette option n'est viable que pour les utilisateurs dans des circonstances urgentes qui peuvent accepter les risques suivants :
>
> - Il n'est pas possible de garantir la cohérence des versions des packages dans l'ensemble des instances de vos environnements.
> - Le bon fonctionnement des environnements modifiés à l'aide de l'outil `pkg-repo` n'est pas garanti. Ceux-ci n'ont pas été testés et vérifiés sur les plateformes prises en charge par Elastic Beanstalk.
>
> Nous recommandons vivement d'appliquer les meilleures pratiques, notamment des plans de test et de désinstallation. Pour faciliter les meilleures pratiques, vous pouvez utiliser la console Elastic Beanstalk et l'EB CLI pour cloner un environnement et échanger un environnement. URLs Pour en savoir plus sur l'utilisation de ces opérations, veuillez consulter la section Déploiements bleu/vert du chapitre Gestion des environnements de ce guide.

Si vous prévoyez de modifier manuellement les fichiers de configuration du référentiel yum, exécutez d'abord l'outil `pkg-repo`. L'outil `pkg-repo` peut ne pas fonctionner comme prévu dans un environnement Amazon Linux 2 avec des fichiers de configuration du référentiel yum modifiés manuellement. Cela est dû au fait que l'outil peut ne pas reconnaître les modifications de configuration.

Pour plus d'informations sur le référentiel de packages Amazon Linux, consultez la rubrique relative au référentiel de packages dans le guide de EC2 l'utilisateur Amazon.

Commandes pkg-repo

Utilisez la syntaxe suivante pour exécuter les commandes de l'outil `pkg-repo`.

```
$ /opt/elasticbeanstalk/bin/pkg-repo command [options]
```

Les commandes `pkg-repo` sont les suivantes :

- lock – verrouille les référentiels de paquets `yum` vers une version spécifique
- unlock – déverrouille les référentiels de paquets `yum` à partir d'une version spécifique
- status – répertorie tous les référentiels de packages `yum` et leur état actuel de verrouillage
- help – affiche une aide générale ou une aide pour une commande en particulier

Les options s'appliquent aux commandes comme suit :

- `lock`, `unlock` et `status` – options : `-h`, `--help` ou aucune (par défaut).
- `help` – options : `lock`, `unlock`, `status` ou aucune (par défaut).

L'exemple suivant exécute uniquement la commande unlock.

```
$ sudo /opt/elasticbeanstalk/bin/pkg-repo unlock
Amazon Linux 2 core package repo successfully unlocked
Amazon Linux 2 extras package repo successfully unlocked
```

L'exemple suivant exécute uniquement la commande lock.

```
$ sudo /opt/elasticbeanstalk/bin/pkg-repo lock
Amazon Linux 2 core package repo successfully locked
Amazon Linux 2 extras package repo successfully locked
```

L'exemple suivant exécute uniquement la commande status.

```
$ sudo /opt/elasticbeanstalk/bin/pkg-repo status
Amazon Linux 2 core package repo is currently UNLOCKED
```

```
Amazon Linux 2 extras package repo is currently UNLOCKED
```

L'exemple suivant exécute uniquement la commande help pour la commande lock.

```
$ sudo /opt/elasticbeanstalk/bin/pkg-repo help lock
```

L'exemple suivant exécute uniquement la commande help pour l'outil `pkg-repo`.

```
$ sudo /opt/elasticbeanstalk/bin/pkg-repo help
```

Vous pouvez tester `pkg-repo` en utilisant SSH pour vous connecter à une instance dans votre environnement Elastic Beanstalk. Une option SSH est la commande eb ssh de l'interface de ligne de commande (CLI) EB.

> **Note**
>
> L'outil `pkg-repo` nécessite des privilèges d'utilisateur racine pour s'exécuter. Si vous obtenez une erreur d'autorisation d'accès, exécutez à nouveau la commande sous `sudo`. Vous n'avez pas besoin d'ajouter `sudo` lorsque vous utilisez l'outil dans les scripts ou les fichiers de configuration que vous déployez dans votre environnement. Elastic Beanstalk exécute tous vos scripts en tant qu'utilisateur racine.

Exemples pkg-repo

La section précédente fournit des exemples de ligne de commande pour les tests sur une EC2 instance individuelle d'un environnement Elastic Beanstalk. Cette approche peut être utile pour les tests. Cependant, elle ne met à jour qu'une seule instance à la fois, ce qui n'est pas pratique pour appliquer des changements à toutes les instances d'un environnement.

Une approche plus pragmatique consiste à utiliser des scripts de hook de plateforme scripts ou un fichier de configuration `.ebextensions` pour appliquer les modifications à toutes les instances de manière cohérente.

L'exemple suivant appelle `pkg-repo` à partir d'un fichier de configuration dans le dossier `.ebextensions`. Elastic Beanstalk exécute les commandes du fichier `update_package.config` lorsque vous déployez votre groupe source d'application.

```
.ebextensions
```

```
### update_package.config
```

Afin de recevoir la dernière version du package docker, cette configuration spécifie le package docker dans la commande yum update.

```
### update_package.config ###

commands:
  update_package:
    command: |
      /opt/elasticbeanstalk/bin/pkg-repo unlock
      yum update docker -y
      /opt/elasticbeanstalk/bin/pkg-repo lock
      yum clean all -y
      rm -rf /var/cache/yum
```

Cette configuration ne spécifie aucun package dans la commande yum update. Toutes les mises à jour disponibles sont appliquées en conséquence.

```
### update_package.config ###

commands:
  update_package:
    command: |
      /opt/elasticbeanstalk/bin/pkg-repo unlock
      yum update -y
      /opt/elasticbeanstalk/bin/pkg-repo lock
      yum clean all -y
      rm -rf /var/cache/yum
```

L'exemple suivant appelle `pkg-repo` à partir d'un script bash en tant que hook de plateforme. Elastic Beanstalk exécute le fichier de script `update_package.sh` situé dans le sous-répertoire `prebuild`.

```
.platform
### hooks
    ### prebuild
        ### update_package.sh
```

Afin de recevoir la dernière version du package docker, ce script spécifie le package docker dans la commande yum update. Si le nom du package est omis, toutes les mises à jour disponibles sont appliquées. L'exemple précédent du fichier de configuration le démontre.

```
### update_package.sh ###

#!/bin/bash

/opt/elasticbeanstalk/bin/pkg-repo unlock
yum update docker -y
/opt/elasticbeanstalk/bin/pkg-repo lock
yum clean all -y
rm -rf /var/cache/yum
```

download-source-bundle (AMI Amazon Linux uniquement)

Sur les branches de plateforme de l'AMI Amazon Linux (précédemment Amazon Linux 2), Elastic Beanstalk fournit un outil supplémentaire, qui est `download-source-bundle`. Utilisez cet outil pour télécharger le code source de votre application lors du déploiement de votre plateforme. Cet outil est disponible dans `/opt/elasticbeanstalk/bin/download-source-bundle`.

L'exemple de script `00-unzip.sh` se trouve dans le dossier `appdeploy/pre` des instances d'environnement. Il montre la façon d'utiliser `download-source-bundle` pour télécharger le code source de l'application dans le dossier `/opt/elasticbeanstalk/deploy/appsource` pendant le déploiement.

Extension des plateformes Linux Elastic Beanstalk

Cette rubrique explique comment étendre vos plateformes Linux avec vos propres commandes, scripts, logiciels et configurations. Vous devrez peut-être étendre votre plateforme pour modifier le serveur proxy par défaut et sa configuration. Vous pouvez également avoir besoin de personnaliser la façon dont la plateforme crée ou démarre votre application.

Rubriques

- Buildfile et Procfile
- Hooks de plateforme
- Fichiers de configuration
- Configuration du proxy inverse
- Exemple d'application avec extensions

Buildfile et Procfile

Certaines plateformes vous permettent de personnaliser la façon dont vous créez ou préparez votre application, mais aussi de spécifier les processus qui exécutent votre application. Chaque sujet de plate-forme individuel mentionne spécifiquement Buildfile and/or Procfile si la plate-forme les prend en charge. Recherchez votre plateforme spécifique sous Plateformes.

Pour toutes les plateformes de prise en charge, la syntaxe et la sémantique sont identiques et sont décrites dans cette page. Chaque rubrique de plateforme mentionne l'utilisation spécifique de ces fichiers pour la création et l'exécution d'applications dans leurs langages respectifs.

BuildFile

Pour spécifier une commande de génération et de configuration personnalisée pour votre application, placez un fichier nommé `Buildfile` dans le répertoire racine de votre source d'application. Le nom de fichier est sensible à la casse. Utilisez la syntaxe suivante pour votre `Buildfile`.

```
<process_name>: <command>
```

La commande dans votre fichier `Buildfile` doit correspondre à l'expression régulière suivante : `^[A-Za-z0-9_-]+:\s*[^\s].*$`.

Elastic Beanstalk ne surveille pas l'application exécutée avec un fichier `Buildfile`. Utilisez un `Buildfile` pour les commandes qui s'exécutent pendant de courtes durées et s'arrêtent après avoir terminé leurs tâches. Pour les processus d'applications de longue durée qui ne doivent pas se fermer, utilisez un Procfile.

Tous les chemins d'accès dans le `Buildfile` sont par rapport à la racine du groupe source. Dans l'exemple suivant de fichier `Buildfile`, `build.sh` est un script shell qui se trouve à la racine du bundle de fichiers source.

Example BuildFile

```
make: ./build.sh
```

Si vous souhaitez fournir des étapes de construction personnalisées, nous vous recommandons d'utiliser des hooks de plateforme `predeploy` pour tout sauf les commandes les plus simples, plutôt qu'un `Buildfile`. Les hooks de plateforme permettent des scripts plus riches et une meilleure gestion des erreurs. Les hooks de plateforme sont décrits dans la section suivante.

Procfile

Pour spécifier des commandes personnalisées pour démarrer et exécuter votre application, placez un fichier nommé `Procfile` dans le répertoire racine de votre source d'application. Le nom de fichier est sensible à la casse. Utilisez la syntaxe suivante pour votre `Procfile`. Vous pouvez spécifier une ou plusieurs commandes.

```
<process_name1>: <command1>
<process_name2>: <command2>
...
```

Chaque ligne de votre fichier `Procfile` doit correspondre à l'expression régulière suivante : `^[A-Za-z0-9_-]+:\s*[^\s].*$`.

Utilisez un fichier `Procfile` pour les processus d'application de longue durée qui ne doivent pas se fermer. Elastic Beanstalk s'attend à ce que les processus s'exécutant à partir du fichier `Procfile` le fassent en continu. Elastic Beanstalk surveille ces processus et redémarre tout processus qui s'arrête. Pour les processus de courte durée, utilisez un Buildfile.

Tous les chemins d'accès dans le `Procfile` sont par rapport à la racine du groupe source. L'exemple `Procfile` suivant définit trois processus. Le premier, appelé `web` dans l'exemple, est l'application Web principale.

Example Procfile

```
web: bin/myserver
cache: bin/mycache
foo: bin/fooapp
```

Elastic Beanstalk configure le serveur proxy de sorte à transférer les demandes vers votre application web principale sur le port 5000. Vous pouvez configurer ce numéro de port. Une utilisation courante pour un `Procfile` est de transmettre ce numéro de port à votre application en tant qu'argument de commande. Pour plus de détails sur la configuration du proxy, consultezConfiguration du proxy inverse.

Elastic Beanstalk capture les flux d'erreurs et de sortie standard à partir des processus `Procfile` dans les fichiers journaux. Elastic Beanstalk donne le nom du processus aux fichiers journaux et les stocke dans `/var/log`. Par exemple, le processus `web` dans l'exemple précédent génère des journaux nommés `web-1.log` et `web-1.error.log` pour `stdout` et `stderr`, respectivement.

Hooks de plateforme

Les hooks de plateforme sont spécifiquement conçus pour étendre la plateforme de votre environnement. Il s'agit de scripts personnalisés et autres fichiers exécutables que vous déployez dans le cadre du code source de votre application et qui sont exécutés par Elastic Beanstalk au cours de différentes étapes de provisionnement d'instance.

> **Note**
>
> Les hooks de plateforme ne sont pas pris en charge sur les versions de plateforme de l'AMI Amazon Linux (précédemment Amazon Linux 2).

Hooks de plateforme de déploiement d'applications

Un déploiement d'application se produit lorsque vous fournissez un nouveau bundle source pour le déploiement ou lorsque vous apportez une modification de configuration qui nécessite la résiliation et la récréation de toutes les instances d'environnement.

Pour fournir des hooks de plateforme qui s'exécutent pendant un déploiement d'application, placez les fichiers sous le répertoire `.platform/hooks` de votre bundle source, dans l'un des sous-répertoires suivants.

- `prebuild` – Les fichiers s'exécutent après que le moteur de plateforme Elastic Beanstalk a téléchargé et extrait le bundle de fichiers source de l'application, et avant qu'il installe et configure l'application et le serveur web.

  Les fichiers `prebuild` s'exécutent après l'exécution des commandes trouvées dans la section commands de tout fichier de configuration et avant l'exécution des commandes `Buildfile`.
- `predeploy` – Les fichiers s'exécutent après que le moteur de plateforme Elastic Beanstalk a installé et configuré l'application et le serveur web, et avant qu'il les déploie dans leur emplacement d'exécution final.

  Les fichiers `predeploy` s'exécutent après l'exécution des commandes trouvées dans la section container_commands de tout fichier de configuration et avant l'exécution des commandes `Procfile`.
- `postdeploy` – Les fichiers s'exécutent après que le moteur de plateforme Elastic Beanstalk a déployé l'application et le serveur proxy.

Il s'agit de la dernière étape du workflow de déploiement.

Hooks de plateforme de déploiement de configuration

Un déploiement de configuration se produit lorsque vous apportez des modifications de configuration qui mettent uniquement à jour les instances d'environnement sans les recréer. Les mises à jour des options suivantes provoquent une mise à jour de la configuration.

- Propriétés de l'environnement et paramètres spécifiques à la plateforme
- Fichiers statiques
- AWS X-Ray démon
- Stockage des journaux et streaming
- Port de l'application (pour plus de détails, voirConfiguration du proxy inverse)

Pour fournir des hooks qui s'exécutent lors d'un déploiement de configuration, placez-les sous le répertoire `.platform/confighooks` de votre bundle source. Les trois mêmes sous-répertoires que pour les hooks de déploiement d'applications s'appliquent.

En savoir plus sur les hooks de plateforme

Les fichiers hooks peuvent être des fichiers binaires ou des fichiers script commençant par une ligne `#!` et contenant leur chemin d'interpréteur, par exemple `#!/bin/bash`. Tous les fichiers doivent disposer d'une autorisation d'exécution. Utilisez la commande `chmod +x` pour définir l'autorisation d'exécution sur vos fichiers hook. Pour toutes les versions de plateforme basées sur Amazon Linux 2023 et Amazon Linux 2 publiées à partir du 29 avril 2022, Elastic Beanstalk accorde automatiquement des autorisations d'exécution à tous les scripts de hook de plateforme. Dans ce cas, vous n'avez pas besoin d'accorder manuellement les autorisations d'exécution. Pour obtenir la liste de ces versions de plateforme, consultez les notes de mise à jour de Linux du 29 avril 2022 dans le AWS Elastic Beanstalk Release Notes Guide (Guide de notes de mise à jour).

Elastic Beanstalk exécute les fichiers dans chacun de ces répertoires et dans l'ordre lexicographique des noms de fichier. Tous les fichiers sont exécutés en tant qu'utilisateur `root`. Le répertoire de travail en cours (cwd) pour les hooks de plateforme est le répertoire racine de l'application. Pour les fichiers `prebuild` et `predeploy`, il s'agit du répertoire intermédiaire de l'application ; pour les fichiers `postdeploy`, il s'agit du répertoire en cours de l'application. Si un des fichiers échoue (fin d'exécution avec un code de sortie différent de zéro), le déploiement échoue.

Un script de texte accroche une plate-forme peut échouer s'il contient des caractères de saut de ligne Windows Carriage Retur/Line Feed (CRLF). Si un fichier a été enregistré sur un hôte Windows, puis transféré vers un serveur Linux, il peut contenir des sauts de ligne Windows CRLF. Pour les plateformes publiées le 29 décembre 2022 ou après cette date, Elastic Beanstalk convertit automatiquement les caractères Windows CRLF en caractères de saut de ligne Linux Line Feed (LF) dans les fichiers texte des hooks de plateforme. Si votre application s'exécute sur des plateformes Amazon Linux 2 publiées avant cette date, vous devez convertir les caractères Windows CRLF en caractères Linux LF. Pour cela, vous pouvez créer et enregistrer le fichier script sur un hôte Linux. Des outils permettant de convertir ces caractères sont également disponibles sur Internet.

Les fichiers hook ont accès à toutes les propriétés d'environnement que vous avez définies dans les options d'application, ainsi qu'aux variables d'environnement système `HOME`, `PATH` et `PORT`.

Pour obtenir des valeurs de variables d'environnement et d'autres options de configuration dans vos scripts de hook de plateforme, vous pouvez utiliser l'utilitaire `get-config` fourni par Elastic Beanstalk sur les instances d'environnement. Pour en savoir plus, consultez the section called “Outils de script de plateforme”.

Fichiers de configuration

Vous pouvez ajouter des fichiers de configuration au répertoire `.ebextensions` du code source de votre application afin de configurer différents aspects de votre environnement Elastic Beanstalk. Entre autres choses, les fichiers de configuration vous permettent de personnaliser le logiciel et d'autres fichiers sur les instances de votre environnement, mais aussi d'exécuter des commandes d'initialisation sur les instances. Pour en savoir plus, consultez the section called “Serveur Linux”.

Vous pouvez également définir des options de configuration à l'aide de fichiers de configuration. De nombreuses options permettent de contrôler le comportement de la plateforme, et certaines d'entre elles sont spécifiques à la plateforme.

Pour les plateformes basées sur Amazon Linux 2 et Amazon Linux 2023, nous vous recommandons d'utiliser les Buildfile, Procfile et les hooks de plateforme pour configurer et exécuter du code personnalisé sur vos instances d'environnement pendant le provisionnement d'instance. Ces mécanismes sont décrits dans les sections précédentes de cette page. Vous pouvez toujours utiliser des commandes et des commandes de conteneur dans les fichiers de configuration `.ebextensions`, mais elles ne sont pas aussi simples à utiliser. Par exemple, écrire des scripts de commande dans un fichier YAML peut être difficile d'un point de vue syntaxique. Vous devez toujours utiliser des fichiers de `.ebextensions` configuration pour tout script nécessitant une référence à une AWS CloudFormation ressource.

Configuration du proxy inverse

Toutes les versions de plateforme Amazon Linux 2 et Amazon Linux 2023 utilisent nginx comme serveur proxy inverse par défaut. Les plateformes Tomcat, Node.js, PHP et Python prennent également en charge Apache HTTPD comme alternative. Pour sélectionner Apache sur ces plateformes, définissez l'option `ProxyServer` dans l'espace de noms `aws:elasticbeanstalk:environment:proxy` sur `apache`. Toutes les plateformes permettent la configuration du serveur proxy de manière uniforme, comme décrit dans cette section.

> **Note**
>
> Sur les versions de la plateforme d'AMI Amazon Linux (précédemment Amazon Linux 2), vous devrez peut-être configurer les serveurs proxy différemment. Vous trouverez ces détails hérités dans les rubriques respectives de la plateforme dans ce guide.

Elastic Beanstalk configure le serveur proxy sur les instances de votre environnement pour transférer le trafic web vers l'application web principale sur l'URL racine de l'environnement ; par exemple, `http://my-env.elasticbeanstalk.com`.

Par défaut, Elastic Beanstalk configure le proxy pour transférer les demandes entrantes sur le port 80 vers votre application web principale sur le port 5000. Vous pouvez configurer ce numéro de port en définissant la propriété d'environnement `PORT` à l'aide de l'espace de noms aws:elasticbeanstalk:application:environment dans un fichier de configuration, comme illustré dans l'exemple suivant.

```
option_settings:
  - namespace:  aws:elasticbeanstalk:application:environment
    option_name:  PORT
    value:  <main_port_number>
```

Pour plus d'informations sur la définition des variables d'environnement pour votre application, consultez the section called “Paramètres d'option”.

Votre application doit écouter sur le port configuré pour elle dans le proxy. Si vous modifiez le port par défaut à l'aide de la propriété d'environnement `PORT`, votre code peut y accéder en lisant la valeur de la variable d'environnement `PORT`. Par exemple, appelez `os.Getenv("PORT")` dans Go, ou `System.getenv("PORT")` dans Java. Si vous configurez votre proxy pour envoyer du trafic vers

plusieurs processus d'application, vous pouvez configurer plusieurs propriétés d'environnement et utiliser leurs valeurs à la fois dans la configuration proxy et dans votre code d'application. Une autre option consiste à transmettre la valeur de port au processus en tant qu'argument de commande dans le `Procfile`. Pour de plus amples informations, veuillez consulter Buildfile et Procfile.

Configuration de nginx

Elastic Beanstalk utilise nginx comme proxy inverse par défaut pour mapper votre application à votre équilibreur de charge Elastic Load Balancing. Elastic Beanstalk fournit une configuration nginx par défaut que vous pouvez étendre ou remplacer totalement par votre propre configuration.

> **Note**
>
> Lorsque vous ajoutez ou modifiez un fichier de configuration `.conf` nginx, veillez à l'encoder en UTF-8.

Pour étendre la configuration nginx par défaut d'Elastic Beanstalk, ajoutez les fichiers de configuration `.conf` à un dossier nommé `.platform/nginx/conf.d/` dans le bundle de fichiers source de votre application. La configuration nginx d'Elastic Beanstalk inclut automatiquement les fichiers `.conf` dans ce dossier.

```
~/workspace/my-app/
|-- .platform
|   `-- nginx
|       `-- conf.d
|           `-- myconf.conf
`-- other source files
```

Les fichiers de configuration `.platform/nginx/conf.d/` sont inclus dans le `http` bloc de configuration nginx. Utilisez cet emplacement pour les configurations qui s'appliquent à l'échelle mondiale.

Pour étendre la configuration par défaut des blocs `server` nginx, `.conf` ajoutez des fichiers de configuration dans un dossier `.platform/nginx/conf.d/elasticbeanstalk/` nommé dans le bundle de sources de votre application. La configuration nginx d'Elastic Beanstalk inclut les fichiers de ce dossier au sein du bloc`.conf`. `server`

```
~/workspace/my-app/
```

```
|-- .platform
|   `-- nginx
|       `-- conf.d
|           `-- elasticbeanstalk
|               `-- server.conf
`-- other source files
```

Utilisez cet emplacement pour ajouter des configurations spécifiques au serveur, telles que des blocs d'emplacement supplémentaires, des pages d'erreur personnalisées ou des directives au niveau du serveur. L'exemple suivant ajoute un bloc de localisation personnalisé.

Example. platform/nginx/conf.d/elasticbeanstalk/server.conf

```
location /test {
    return 200 "Hello World!";
    add_header Content-Type text/plain;
}
```

Pour remplacer complètement la configuration nginx par défaut d'Elastic Beanstalk, incluez une configuration dans votre bundle de fichiers source à l'emplacement `.platform/nginx/nginx.conf` :

```
~/workspace/my-app/
|-- .platform
|   `-- nginx
|       `-- nginx.conf
`-- other source files
```

Si vous remplacez la configuration nginx d'Elastic Beanstalk, ajoutez la ligne suivante à votre fichier `nginx.conf` afin d'extraire les configurations d'Elastic Beanstalk pour la Rapports et surveillance de l'état de santé améliorés dans Elastic Beanstalk, les mappages d'application automatiques et les fichiers statiques.

```
include conf.d/elasticbeanstalk/*.conf;
```

Configuration d'Apache HTTPD

Les plateformes Tomcat, Node.js, PHP et Python vous permettent de choisir le serveur proxy HTTPD Apache comme alternative à nginx. Ce n'est pas la valeur par défaut. L'exemple suivant montre comment configurer Elastic Beanstalk pour utiliser Apache HTTPD.

Example.ebextensions/httpd-proxy.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache
```

Vous pouvez étendre la configuration Apache Elastic Beanstalk par défaut avec vos fichiers de configuration supplémentaires. Sinon, vous pouvez remplacer complètement la configuration Apache Elastic Beanstalk par défaut.

Pour étendre la configuration Apache Elastic Beanstalk par défaut, ajoutez les fichiers de configuration `.conf` à un dossier nommé `.platform/httpd/conf.d` dans le bundle de fichiers source de votre application. La configuration Apache Elastic Beanstalk par défaut inclut automatiquement les fichiers `.conf` dans ce dossier.

```
~/workspace/my-app/
|-- .ebextensions
|   -- httpd-proxy.config
|-- .platform
|   -- httpd
|      -- conf.d
|         -- port5000.conf
|         -- ssl.conf
-- index.jsp
```

Par exemple, la configuration Apache 2.4 suivante ajoute un écouteur sur le port 5000 :

Example. platform/httpd/conf.d/port5000.conf

```
listen 5000
<VirtualHost *:5000>
  <Proxy *>
    Require all granted
  </Proxy>
  ProxyPass / http://localhost:8080/ retry=0
  ProxyPassReverse / http://localhost:8080/
  ProxyPreserveHost on

  ErrorLog /var/log/httpd/elasticbeanstalk-error_log
</VirtualHost>
```

Pour remplacer complètement la configuration Apache Elastic Beanstalk par défaut, incluez une configuration dans votre bundle de fichiers source sur `.platform/httpd/conf/httpd.conf`.

```
~/workspace/my-app/
|-- .ebextensions
|   -- httpd-proxy.config
|-- .platform
|   `-- httpd
|       `-- conf
|           `-- httpd.conf
`-- index.jsp
```

> **Note**
>
> Si vous remplacez la configuration Apache Elastic Beanstalk, ajoutez les lignes suivantes à votre fichier `httpd.conf` afin d'extraire les configurations Elastic Beanstalk pour la Rapports et surveillance de l'état de santé améliorés dans Elastic Beanstalk, les mappages d'application automatiques et les fichiers statiques.
>
> ```
> IncludeOptional conf.d/elasticbeanstalk/*.conf
> ```

Exemple d'application avec extensions

L'exemple suivant présente un bundle de fichiers source d'application avec plusieurs fonctionnalités d'extensibilité prises en charge par les plateformes Elastic Beanstalk Amazon Linux 2 et Amazon Linux 2023 : un fichier `Procfile`, des fichiers de configuration `.ebextensions`, des hooks personnalisés et des fichiers de configuration de proxy.

```
~/my-app/
|-- web.jar
|-- Procfile
|-- readme.md
|-- .ebextensions/
|   |-- options.config        # Option settings
|   `-- cloudwatch.config     # Other .ebextensions sections, for example files and
 container commands
`-- .platform/
    |-- nginx/                # Proxy configuration
    |   |-- nginx.conf
```

```
|   `-- conf.d/
|       |-- custom.conf
|       `-- elasticbeanstalk/
|           `-- server.conf
|-- hooks/                # Application deployment hooks
|   |-- prebuild/
|   |   |-- 01_set_secrets.sh
|   |   `-- 12_update_permissions.sh
|   |-- predeploy/
|   |   `-- 01_some_service_stop.sh
|   `-- postdeploy/
|       |-- 01_set_tmp_file_permissions.sh
|       |-- 50_run_something_after_app_deployment.sh
|       `-- 99_some_service_start.sh
`-- confighooks/          # Configuration deployment hooks
    |-- prebuild/
    |   `-- 01_set_secrets.sh
    |-- predeploy/
    |   `-- 01_some_service_stop.sh
    `-- postdeploy/
        |-- 01_run_something_after_config_deployment.sh
        `-- 99_some_service_start.sh
```

> **Note**
>
> Certaines de ces extensions ne sont pas prises en charge sur les versions de plateforme de l'AMI Amazon Linux (précédemment Amazon Linux 2).

Déploiement d'applications .NET Windows avec Elastic Beanstalk

> **Consultez le .NET on AWS Developer Center**
>
> Vous êtes-vous déjà rendu dans notre centre de développement .Net ? C'est notre guichet unique pour tout ce qui concerne .NET AWS.
> Pour plus d'informations, consultez le .NET on AWS Developer Center.

Ce chapitre fournit des instructions pour configurer et déployer vos applications Web ASP.NET et .NET Core pour AWS Elastic Beanstalk Windows sur. Elastic Beanstalk facilite le déploiement, la gestion et le dimensionnement de vos applications Web .NET (Windows) à l'aide d'Amazon Web Services.

Vous pouvez déployer votre application en quelques minutes à l'aide de l'interface de ligne de commande Elastic Beanstalk (EB CLI) ou de la console Elastic Beanstalk. Après avoir déployé votre application Elastic Beanstalk, vous pouvez continuer à utiliser l'EB CLI pour gérer votre application et votre environnement, ou vous pouvez utiliser la console Elastic Beanstalk, ou le. AWS CLI APIs

Ce chapitre propose les didacticiels suivants :

- QuickStart pour .NET Core sous Windows— Step-by-step instructions pour créer et déployer une application Windows Hello World .NET Core à l'aide de l'interface de ligne de commande EB.
- QuickStart pour ASP.NET— Step-by-step instructions pour créer et déployer une application Hello World ASP.NET à l'aide du AWS Toolkit for Visual Studio.

Si vous avez besoin d'aide pour le développement d'applications Windows .NET Core, vous pouvez vous rendre à plusieurs endroits :

- Forum de développement .NET : postez vos questions et recevez des commentaires.
- .NET Developer Center : guichet unique pour des exemples de code, de la documentation, des outils et des ressources supplémentaires.
- AWS Documentation du SDK pour .NET : découvrez comment configurer le SDK et exécuter des exemples de code, les fonctionnalités du SDK et des informations détaillées sur les opérations d'API du SDK.

> **Note**
>
> Cette plate-forme ne prend pas en charge les environnements de travail. Pour en savoir plus, consultez Environnements de travail Elastic Beanstalk.

Rubriques

- QuickStart: Déployer une application .NET Core sous Windows sur Elastic Beanstalk
- QuickStart: Déployer une application ASP.NET sur Elastic Beanstalk
- Configuration de votre environnement de développement .NET
- Utilisation de la plateforme Windows Elastic Beanstalk .NET
- Ajout d'une instance de base de données Amazon RDS à votre environnement d'application .NET
- Le AWS Toolkit for Visual Studio
- Migration de votre application .NET sur site vers Elastic Beanstalk
- Recommandations pour les composants retirés de Windows Server sur Elastic Beanstalk

QuickStart: Déployer une application .NET Core sous Windows sur Elastic Beanstalk

Ce QuickStart didacticiel explique le processus de création d'une application .NET Core sous Windows et son déploiement dans un AWS Elastic Beanstalk environnement.

> **Non destiné à une utilisation en production**
>
> Les exemples sont fournis à titre de démonstration uniquement. N'utilisez pas d'exemples d'applications en production.

Sections

- Votre AWS compte
- Conditions préalables
- Étape 1 : Création d'une application .NET Core sous Windows
- Étape 2 : Exécutez votre application localement

- Étape 3 : Déployez votre application .NET Core sous Windows avec l'EB CLI
- Étape 4 : Exécutez votre application sur Elastic Beanstalk
- Étape 5 : nettoyer
- AWS ressources pour votre application
- Étapes suivantes
- Déployez avec la console Elastic Beanstalk

Votre AWS compte

Si vous n'êtes pas encore AWS client, vous devez créer un AWS compte. L'inscription vous permet d'accéder à Elastic Beanstalk AWS et aux autres services dont vous avez besoin.

Si vous avez déjà un AWS compte, vous pouvez passer àConditions préalables.

Créez un AWS compte

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez l'https://portal.aws.amazon.com/billing/inscription.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d’inscription, vous recevrez un appel téléphonique ou un SMS et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

   Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l’utilisateur racine a accès à l’ensemble des Services AWS et des ressources de ce compte. La meilleure pratique de sécurité consiste à attribuer un accès administratif à un utilisateur, et à utiliser uniquement l’utilisateur racine pour effectuer les tâches nécessitant un accès utilisateur racine.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. À tout moment, vous pouvez consulter l'activité actuelle de votre compte et gérer votre compte en accédant à https://aws.amazon.com/et en choisissant Mon compte.

Création d’un utilisateur doté d’un accès administratif

Une fois que vous vous êtes inscrit à un utilisateur administratif Compte AWS, que vous Utilisateur racine d'un compte AWS l'avez sécurisé AWS IAM Identity Center, que vous l'avez activé et que vous en avez créé un, afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l’aide pour vous connecter en utilisant l’utilisateur racine, consultez Connexion en tant qu’utilisateur racine dans le Guide de l’utilisateur Connexion à AWS .
2. Activez l’authentification multifactorielle (MFA) pour votre utilisateur racine.

   Pour obtenir des instructions, consultez la section Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d’un utilisateur doté d’un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d’ AWS IAM Identity Center dans le Guide de l’utilisateur AWS IAM Identity Center .
2. Dans IAM Identity Center, octroyez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connexion en tant qu’utilisateur doté d’un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l’URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l’utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribution d'un accès à d'autres utilisateurs

1. Dans IAM Identity Center, créez un ensemble d'autorisations qui respecte la bonne pratique consistant à appliquer les autorisations de moindre privilège.

   Pour obtenir des instructions, consultez Création d'un ensemble d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .

2. Attribuez des utilisateurs à un groupe, puis attribuez un accès par authentification unique au groupe.

   Pour obtenir des instructions, consultez Ajout de groupes dans le Guide de l'utilisateur AWS IAM Identity Center .

Conditions préalables

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Les commandes sont affichées dans des listes précédées d'un symbole d'invite (>) et du nom du répertoire en cours, le cas échéant.

```
C:\eb-project> this is a command
this is output
```

INTERFACE DE LIGNE DE COMMANDE (CLI) EB

Ce tutoriel utilise également l'interface de ligne de commande Elastic Beanstalk (CLI EB). Pour de plus amples informations sur l'installation et la configuration de la CLI EB, veuillez consulter Installation de la CLI EB avec un script de configuration (recommandé) et Configuration de l'interface de ligne de commande EB.

.NET Core sous Windows

Si le SDK .NET n'est pas installé sur votre ordinateur local, vous pouvez l'installer en suivant le lien Télécharger .NET sur le site Web de documentation .NET.

Vérifiez votre installation du SDK .NET en exécutant la commande suivante.

```
C:\> dotnet --info
```

Étape 1 : Création d'une application .NET Core sous Windows

Créez un répertoire de projet.

```
C:\> mkdir eb-dotnetcore
C:\> cd eb-dotnetcore
```

Créez ensuite un exemple d'application de service RESTful Web Hello World en exécutant les commandes suivantes.

```
C:\eb-dotnetcore> dotnet new web --name HelloElasticBeanstalk
C:\eb-dotnetcore> cd HelloElasticBeanstalk
```

Étape 2 : Exécutez votre application localement

Exécutez la commande suivante pour exécuter votre application localement.

```
C:\eb-dotnetcore\HelloElasticBeasntalk> dotnet run
```

Le résultat doit ressembler au texte suivant.

```
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: https://localhost:7222
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5228
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\Users\Administrator\eb-dotnetcore\HelloElasticBeanstalk
```

> **Note**
>
> La `dotnet` commande sélectionne un port au hasard lors de l'exécution locale de l'application. Dans cet exemple, le port est 5228. Lorsque vous déployez l'application dans votre environnement Elastic Beanstalk, elle s'exécute sur le port 5000.

Entrez l'adresse URL `http://localhost:port` dans votre navigateur Web. Pour cet exemple spécifique, la commande est`http://localhost:5228`. Le navigateur Web doit afficher « Hello World ! ».

Étape 3 : Déployez votre application .NET Core sous Windows avec l'EB CLI

Exécutez les commandes suivantes pour créer un environnement Elastic Beanstalk pour cette application.

Pour créer un environnement et déployer votre application .NET Core sous Windows

1. Exécutez les commandes suivantes dans le `HelloElasticBeanstalk` répertoire pour publier et compresser votre application.

```
C:\eb-dotnetcore\HelloElasticBeasntalk> dotnet publish -o site
C:\eb-dotnetcore\HelloElasticBeasntalk> cd site
C:\eb-dotnetcore\HelloElasticBeasntalk\site> Compress-Archive -Path * -
DestinationPath ../site.zip
C:\eb-dotnetcore\HelloElasticBeasntalk\site> cd ..
```

2. Créez un nouveau fichier dans l'`HelloElasticBeanstalk`appel `aws-windows-deployment-manifest.json` avec le contenu suivant :

```
{
    "manifestVersion": 1,
    "deployments": {
        "aspNetCoreWeb": [
        {
            "name": "test-dotnet-core",
            "parameters": {
                "appBundle": "site.zip",
                "iisPath": "/",
                "iisWebSite": "Default Web Site"
            }
        }
        ]
    }
}
```

3. Initialisez votre référentiel de la CLI EB avec la commande eb init :

```
C:\eb-dotnetcore\HelloElasticBeasntalk> eb init -p iis dotnet-windows-server-
tutorial --region us-east-2
```

Cette commande crée une application nommée `dotnet-windows-server-tutorial` et configure votre référentiel local pour créer des environnements dotés de la dernière version de la plate-forme Windows Server.

4. Créez un environnement et déployez-y votre application avec eb create. Elastic Beanstalk crée automatiquement un fichier zip pour votre application et le démarre sur le port 5000.

```
C:\eb-dotnetcore\HelloElasticBeasntalk> eb create dotnet-windows-server-env
```

Il faut environ cinq minutes à Elastic Beanstalk pour créer votre environnement.

Étape 4 : Exécutez votre application sur Elastic Beanstalk

Lorsque le processus de création de votre environnement est terminé, ouvrez votre site Web aveceb open.

```
C:\eb-dotnetcore\HelloElasticBeasntalk> eb open
```

Félicitations ! Vous avez déployé une application .NET Core sur Windows avec Elastic Beanstalk ! Celle-ci ouvre une fenêtre de navigation en utilisant le nom de domaine créé pour votre application.

Étape 5 : nettoyer

Vous pouvez mettre fin à votre environnement lorsque vous avez fini d'utiliser votre application. Elastic Beanstalk AWS met fin à toutes les ressources associées à votre environnement.

Pour terminer votre environnement Elastic Beanstalk avec l'EB CLI, exécutez la commande suivante.

```
C:\eb-dotnetcore\HelloElasticBeasntalk> eb terminate
```

AWS ressources pour votre application

Vous venez de créer une application à instance unique. Il s'agit d'un exemple d'application simple avec une seule instance EC2. Il ne nécessite donc pas d'équilibrage de charge ni de

dimensionnement automatique. Pour les applications à instance unique, Elastic Beanstalk crée les ressources suivantes : AWS

- Instance EC2 – Une machine virtuelle Amazon EC2 configurée pour exécuter des applications web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble distinct de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse qui traite le trafic web devant votre application web, lui transmet les demandes, traite les ressources statiques et génère des journaux d'accès et d'erreurs.
- Groupe de sécurité de l'instance – Un groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui mène à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

Elastic Beanstalk gère toutes ces ressources. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

Étapes suivantes

Dès que vous disposez d'un environnement exécutant une application, vous pouvez à tout moment déployer une nouvelle version de l'application ou une application totalement différente. Le déploiement d'une nouvelle version d'application est très rapide, car il n'est pas nécessaire de mettre en service ni de redémarrer les instances EC2. Vous pouvez également explorer votre

nouvel environnement à l'aide de la console Elastic Beanstalk. Pour connaître les étapes détaillées, consultez la section Explorez votre environnement dans le chapitre Mise en route de ce guide.

> **Essayez d'autres didacticiels**
>
> Si vous souhaitez essayer d'autres didacticiels avec différents exemples d'applications, consultezQuickStart pour ASP.NET.

Une fois que vous avez déployé un ou deux exemples d'applications et que vous êtes prêt à commencer à développer et à exécuter des applications .NET Core sur Windows localement, voir Configuration de votre environnement de développement .NET

Déployez avec la console Elastic Beanstalk

Vous pouvez également utiliser la console Elastic Beanstalk pour lancer l'exemple d'application. Pour connaître les étapes détaillées, voir Création d'un exemple d'application dans le chapitre Mise en route de ce guide.

QuickStart: Déployer une application ASP.NET sur Elastic Beanstalk

Ce QuickStart didacticiel explique le processus de création d'une application ASP.NET et de son déploiement dans un AWS Elastic Beanstalk environnement.

> **Non destiné à une utilisation en production**
>
> Les exemples sont destinés uniquement à des fins de démonstration. N'utilisez pas d'exemples d'applications en production.

Sections

- Votre AWS compte
- Conditions préalables
- Étape 1 : créer une application ASP.NET
- Étape 2 : Exécutez votre application localement

- Étape 3 : Déployez votre application ASP.NET avec AWS Toolkit for Visual Studio
- Étape 4 : Exécutez votre application sur Elastic Beanstalk
- Étape 5 : nettoyer
- AWS ressources pour votre application
- Étapes suivantes
- Déployez avec la console Elastic Beanstalk

Votre AWS compte

Si vous n'êtes pas encore AWS client, vous devez créer un AWS compte. L'inscription vous permet d'accéder à Elastic Beanstalk AWS et aux autres services dont vous avez besoin.

Si vous avez déjà un AWS compte, vous pouvez passer àConditions préalables.

Créez un AWS compte

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez l'https://portal.aws.amazon.com/billing/inscription.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d’inscription, vous recevrez un appel téléphonique ou un SMS et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

   Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l’utilisateur racine a accès à l’ensemble des Services AWS et des ressources de ce compte. La meilleure pratique de sécurité consiste à attribuer un accès administratif à un utilisateur, et à utiliser uniquement l’utilisateur racine pour effectuer les tâches nécessitant un accès utilisateur racine.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. À tout moment, vous pouvez consulter l'activité actuelle de votre compte et gérer votre compte en accédant à https://aws.amazon.com/et en choisissant Mon compte.

Création d’un utilisateur doté d’un accès administratif

Une fois que vous vous êtes inscrit à un utilisateur administratif Compte AWS, que vous Utilisateur racine d'un compte AWS l'avez sécurisé AWS IAM Identity Center, que vous l'avez activé et que vous en avez créé un, afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l’aide pour vous connecter en utilisant l’utilisateur racine, consultez Connexion en tant qu’utilisateur racine dans le Guide de l’utilisateur Connexion à AWS .
2. Activez l’authentification multifactorielle (MFA) pour votre utilisateur racine.

   Pour obtenir des instructions, consultez la section Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d’un utilisateur doté d’un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d’ AWS IAM Identity Center dans le Guide de l’utilisateur AWS IAM Identity Center .
2. Dans IAM Identity Center, octroyez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connexion en tant qu’utilisateur doté d’un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l’URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l’utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribution d’un accès à d’autres utilisateurs

1. Dans IAM Identity Center, créez un ensemble d’autorisations qui respecte la bonne pratique consistant à appliquer les autorisations de moindre privilège.

   Pour obtenir des instructions, consultez Création d’un ensemble d’autorisations dans le Guide de l’utilisateur AWS IAM Identity Center .
2. Attribuez des utilisateurs à un groupe, puis attribuez un accès par authentification unique au groupe.

   Pour obtenir des instructions, consultez Ajout de groupes dans le Guide de l’utilisateur AWS IAM Identity Center .

Conditions préalables

Ce QuickStart didacticiel explique comment créer une application « Hello World » et comment la déployer dans un environnement Elastic Beanstalk avec Visual Studio et le. AWS Toolkit for Visual Studio

Visual Studio

Pour télécharger et installer Visual Studio, suivez les instructions de la page de téléchargement de Visual Studio. Cet exemple utilise Visual Studio 2022. Lors de l'installation de Visual Studio, sélectionnez les éléments spécifiques suivants :

- Dans l'onglet Charges de travail, sélectionnez ASP.NET et développement Web.
- Dans l'onglet Composants individuels, sélectionnez les outils de développement .NET Framework 4.8 et les modèles de projet et d'élément .NET Framework.

AWS Toolkit for Visual Studio

Pour télécharger et configurer, AWS Toolkit for Visual Studio suivez les instructions de la rubrique Mise en route du Guide de l' AWS Toolkit for Visual Studio utilisateur.

Étape 1 : créer une application ASP.NET

Créez ensuite une application que vous déploierez dans un environnement Elastic Beanstalk. Nous allons créer une application Web ASP.NET « Hello World ».

Pour créer une application ASP.NET

1. Lancez Visual Studio. Dans le menu Fichier, sélectionnez Nouveau, puis Projet.
2. La boîte de dialogue Créer un nouveau projet s'affiche. Sélectionnez l'application Web ASP.NET (.NET Framework), puis sélectionnez Suivant.
3. Dans la boîte de dialogue Configurer votre nouveau projet, entrez `eb-aspnet` le nom de votre projet. Dans le menu déroulant Framework, sélectionnez .NET Framework 4.8, puis sélectionnez Créer.

   Notez le répertoire du projet. Dans cet exemple, le répertoire du projet est`C:\Users\Administrator\source\repos\eb-aspnet\eb-aspnet`.
4. La boîte de dialogue Créer une nouvelle application Web ASP.NET s'affiche. Sélectionnez le modèle vide. Sélectionnez ensuite Créer.

   À ce stade, vous avez créé un projet d'application Web ASP.NET vide à l'aide de Visual Studio. Nous allons maintenant créer un formulaire Web qui servira de point d'entrée à l'application Web ASP.NET.
5. Dans le menu Projet, sélectionnez Ajouter un nouvel élément. Sur la page Ajouter un nouvel élément, sélectionnez Formulaire Web et nommez-le`Default.aspx`. Sélectionnez ensuite Ajouter.
6. Ajoutez ce qui suit à `Default.aspx`:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
 Inherits="eb_aspnet.Default" %>

<!DOCTYPE html>

<html xmlns="https://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Hello Elastic Beanstalk!</title>
</head>
<body>
    <form id="body" runat="server">
        <div>
            Hello Elastic Beanstalk! This is an ASP.NET on Windows Server
 application.
        </div>
    </form>
</body>
```

```
</html>
```

Étape 2 : Exécutez votre application localement

Dans Visual Studio, dans le menu Debug, sélectionnez Démarrer le débogage pour exécuter votre application localement. La page doit afficher « Hello Elastic Beanstalk ! Il s'agit d'une application ASP.NET sur Windows Server. »

Étape 3 : Déployez votre application ASP.NET avec AWS Toolkit for Visual Studio

Suivez ces étapes pour créer un environnement Elastic Beanstalk et y déployer votre nouvelle application.

Pour créer un environnement et déployer votre application ASP.NET

1. Dans l'Explorateur de solutions, cliquez avec le bouton droit sur votre application, puis sélectionnez Publier sur AWS Elastic Beanstalk.
2. Choisissez un nom pour votre nouvelle application et votre nouvel environnement Elastic Beanstalk.
3. Au-delà de ce point, vous pouvez continuer avec les paramètres par défaut fournis par Elastic Beanstalk ou modifier les options et les paramètres à votre guise.
4. Sur la page de révision, sélectionnez Déployer. Cela va empaqueter votre application Web ASP.NET et la déployer sur Elastic Beanstalk.

   Il faut environ cinq minutes à Elastic Beanstalk pour créer votre environnement. La fonctionnalité de déploiement Elastic Beanstalk surveille votre environnement jusqu'à ce qu'il devienne disponible avec le code nouvellement déployé. Dans l'onglet Env : **< environment name >**, vous verrez l'état de votre environnement.

Étape 4 : Exécutez votre application sur Elastic Beanstalk

Lorsque le processus de création de votre environnement est terminé, l'onglet Env : **< environment name >** affiche des informations sur votre environnement et votre application, notamment l'URL du domaine permettant de lancer votre application. Sélectionnez cette URL dans cet onglet ou copiez-la et collez-la dans votre navigateur Web.

Félicitations ! Vous avez déployé une application ASP.NET avec Elastic Beanstalk !

Étape 5 : nettoyer

Lorsque vous avez fini d'utiliser votre application, vous pouvez arrêter votre environnement dans le AWS Toolkit for Visual Studio.

Pour résilier votre environnement

1. Développez le nœud Elastic Beanstalk et le nœud d'application dans AWS Explorer (Explorateur AWS). Cliquez avec le bouton droit la souris sur votre environnement d'application et sélectionnez Terminate Environment (Résilier l'environnement).
2. Lorsque vous y êtes invité, sélectionnez Oui (Yes) afin de confirmer que vous souhaitez résilier l'environnement. Il faudra quelques minutes à Elastic Beanstalk pour arrêter les ressources en cours d'exécution dans l' AWS environnement.

AWS ressources pour votre application

Vous venez de créer une application à instance unique. Il s'agit d'un exemple d'application simple avec une seule instance EC2, de sorte qu'il ne nécessite pas d'équilibrage de charge ni de dimensionnement automatique. Pour les applications à instance unique, Elastic Beanstalk crée les ressources suivantes : AWS

- Instance EC2 – Une machine virtuelle Amazon EC2 configurée pour exécuter des applications web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble distinct de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse qui traite le trafic web devant votre application web, lui transmet les demandes, traite les ressources statiques et génère des journaux d'accès et d'erreurs.
- Groupe de sécurité de l'instance – Un groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.

- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack : Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

Elastic Beanstalk gère toutes ces ressources. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

Étapes suivantes

Dès que vous disposez d'un environnement exécutant une application, vous pouvez à tout moment déployer une nouvelle version de l'application ou une application totalement différente. Le déploiement d'une nouvelle version d'application est très rapide, car il n'est pas nécessaire de mettre en service ni de redémarrer les instances EC2. Vous pouvez également explorer votre nouvel environnement à l'aide de la console Elastic Beanstalk. Pour connaître les étapes détaillées, consultez la section Explorez votre environnement dans le chapitre Mise en route de ce guide.

> **Essayez d'autres didacticiels**
>
> Si vous souhaitez essayer d'autres didacticiels avec différents exemples d'applications, consultezQuickStart pour .NET Core sous Windows.

Une fois que vous avez déployé un ou deux exemples d'applications et que vous êtes prêt à commencer à développer et à exécuter des applications ASP.NET localement, voir Configuration de votre environnement de développement .NET

Déployez avec la console Elastic Beanstalk

Vous pouvez également utiliser la console Elastic Beanstalk pour lancer l'exemple d'application. Pour connaître les étapes détaillées, voir Création d'un exemple d'application dans le chapitre Mise en route de ce guide.

Configuration de votre environnement de développement .NET

Cette rubrique fournit des instructions pour configurer un environnement de développement .NET Windows afin de tester votre application localement avant de la déployer AWS Elastic Beanstalk. Il fait également référence à des sites Web qui fournissent des instructions d'installation pour des outils utiles.

Sections

- Installation d'un IDE
- Installation du AWS Toolkit for Visual Studio

Si vous devez gérer AWS des ressources depuis votre application, installez le AWS SDK pour .NET. Par exemple, vous pouvez utiliser Amazon S3 pour stocker et récupérer des données.

Avec le AWS SDK pour .NET, vous pouvez démarrer en quelques minutes avec un seul package téléchargeable comprenant des modèles de projet Visual Studio, AWS la bibliothèque .NET, des exemples de code C# et de la documentation. Des exemples pratiques sont fournis dans C# sur la façon d'utiliser les bibliothèques pour créer des applications. Nous vous proposons des didacticiels vidéo en ligne et des documents de référence pour vous aider à utiliser les bibliothèques et les exemples de code.

Consultez la page d'accueil AWS SDK pour .NET pour plus d'informations et des instructions d'installation.

Installation d'un IDE

Les environnements de développement intégrés (IDEs) fournissent un large éventail de fonctionnalités qui facilitent le développement d'applications. Si vous n'avez pas utilisé un IDE pour le développement .NET, essayez Visual Studio Community pour démarrer.

Visitez la page Visual Studio Community pour télécharger et installer Visual Studio Community.

Installation du AWS Toolkit for Visual Studio

Il s'AWS Toolkit for Visual Studioagit d'un plug-in open source pour l'IDE Visual Studio qui permet aux développeurs de développer, de déboguer et de déployer plus facilement des applications .NET à l'aide AWS de. Consultez la page d'accueil Toolkit for Visual Studio pour obtenir des instructions d'installation.

Utilisation de la plateforme Windows Elastic Beanstalk .NET

Cette rubrique explique comment configurer, créer et exécuter vos applications Web ASP.NET et .NET Core pour Windows sur Elastic Beanstalk.

AWS Elastic Beanstalk prend en charge un certain nombre de plateformes pour différentes versions du framework de programmation .NET et de Windows Server. Consultez .NET sur Windows Server avec IIS dans le document Plateformes AWS Elastic Beanstalk pour obtenir une liste complète.

Elastic Beanstalk propose des options de configuration que vous pouvez utiliser pour personnaliser le logiciel qui s'exécute EC2 sur les instances de votre environnement Elastic Beanstalk. Vous pouvez configurer des variables d'environnement nécessaires pour votre application, activer la rotation des journaux sur Amazon S3 et définir les paramètres de .NET framework.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour modifier la configuration d'un environnement en cours d'exécution. Pour éviter de perdre la configuration de votre environnement en le résiliant, vous pouvez utiliser des configurations enregistrées pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des fichiers de configuration. Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

Les paramètres appliqués dans la console Elastic Beanstalk remplacent les mêmes paramètres des fichiers de configuration, s'ils existent. Cela vous permet d'utiliser les paramètres par défaut dans les fichiers de configuration et de les remplacer par des paramètres spécifiques à l'environnement dans la console. Pour de plus amples informations sur la priorité et les autres méthodes de modification des paramètres, veuillez consulter Options de configuration.

Configuration de votre environnement .NET dans la console Elastic Beanstalk

Vous pouvez utiliser la console Elastic Beanstalk pour activer la rotation des journaux sur Amazon S3, configurer des variables que votre application peut lire depuis l'environnement et modifier les paramètres de .NET Framework.

Pour configurer votre environnement .NET dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.

Options de conteneur

- Cibler l'exécution .NET – Réglez sur `2.0` pour exécuter CLR v2.
- Activer les applications 32 bits – Réglez sur `True` pour exécuter des applications 32 bits.

Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Activer la rotation des fichiers journaux vers Amazon S3 : indique si les fichiers journaux des EC2 instances Amazon de votre application sont copiés dans le compartiment Amazon S3 associé à votre application.

Propriétés de l'environnement

La section Propriétés de l'environnement vous permet de définir les paramètres de configuration de l'environnement sur EC2 les instances Amazon qui exécutent votre application. Ces paramètres sont passés en tant que paires clé-valeur à l'application. Utilisez `System.GetEnvironmentVariable` pour les lire. Des clés identiques peuvent exister dans `web.config` et en tant que propriétés de l'environnement. Utilisez l'espace de noms `System.Configuration` pour lire les valeurs de `web.config`.

```
NameValueCollection appConfig = ConfigurationManager.AppSettings;
string endpoint = appConfig["API_ENDPOINT"];
```

Pour plus d'informations, consultez Variables d'environnement et autres paramètres du logiciel.

Espace de noms aws:elasticbeanstalk:container:dotnet:apppool

Vous pouvez utiliser un fichier de configuration pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être spécifiques à la plate-forme ou s'appliquer à toutes les plateformes du service Elastic Beanstalk dans son ensemble. Les options de configuration sont organisées en espaces de noms.

La plateforme .NET définit les options dans l'espace de noms `aws:elasticbeanstalk:container:dotnet:apppool` que vous pouvez utiliser pour configurer l'environnement d'exécution .NET.

L'exemple de fichier de configuration suivant affiche des paramètres pour chacune des options disponibles dans cet espace de noms :

Example.ebextensions/dotnet-settings.config

```
option_settings:
  aws:elasticbeanstalk:container:dotnet:apppool:
    Target Runtime: 2.0
    Enable 32-bit Applications: True
```

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d' AWS CLI. Pour plus d'informations, consultez Options de configuration.

Migration entre les principales versions de la plateforme Windows Server pour Elastic Beanstalk

AWS Elastic Beanstalk dispose de plusieurs versions majeures de sa plateforme Windows Server. Cette page couvre les améliorations principales de chaque version majeure et les éléments que vous devez prendre en compte avant de migrer vers une version ultérieure.

La plateforme Windows Server est actuellement à la version 2 (v2). Si votre application utilise une version de plateforme Windows Server antérieure à v2, nous vous recommandons de migrer vers v2.

Nouveautés des versions majeures de la plateforme Windows Server

Plateforme Windows Server V2

La version 2 (v2) de la plateforme Windows Server pour Elastic Beanstalk a été publiée en février 2019. La V2 aligne le comportement de la plateforme Windows Server sur celui des plateformes Linux Elastic Beanstalk pour plusieurs aspects importants. Le V2 est entièrement rétrocompatible avec la V1, ce qui facilite la migration à partir de la V1.

La plateforme Windows Server prend désormais en charge les fonctionnalités suivantes :

- Gestion des versions : chaque version reçoit un nouveau numéro et vous pouvez consulter les versions précédentes (qui sont toujours disponibles) lors de la création et de la gestion des environnements.
- État amélioré – Pour plus de détails, veuillez consulter Rapports et surveillance de l'état de santé améliorés dans Elastic Beanstalk.
- Déploiements immuables et propagés avec un lot supplémentaire – Pour de plus amples informations sur les stratégies de déploiement, veuillez consulter Déploiement d'applications dans des environnements Elastic Beanstalk.
- Mises à jour immuables – Pour de plus amples informations sur les types de mises à jour, veuillez consulter Configuration changes.
- Mises à jour gérées de la plateforme – Pour de plus amples informations, veuillez consulter Mises à jour gérées de la plateforme.

> **Note**
>
> Les nouvelles fonctionnalités de déploiement et de mise à jour dépendent des rapports améliorés sur l'état de santé. Activez la fonctionnalité correspondante pour les utiliser. Pour plus d'informations, consultez Activation des rapports améliorés sur l'état Elastic Beanstalk.

Plateforme Windows Server V1

La version 1.0.0 (v1) de la plateforme Windows Server pour Elastic Beanstalk a été publiée en octobre 2015. Cette version modifie l'ordre dans lequel Elastic Beanstalk traite les commandes dans les fichiers de configuration lors de la création et des mises à jour d'un environnement.

Les versions antérieures de la plateforme ne comportent pas de numéro dans le nom de pile de solutions :

- Windows Server 64 bits 2012 R2 exécutant IIS 8.5
- Windows Server Core 64 bits 2012 R2 exécutant IIS 8.5
- Windows Server 64 bits 2012 exécutant IIS 8
- Windows Server 64 bits 2008 R2 exécutant IIS 7.5

Dans les versions antérieures, l'ordre de traitement des fichiers de configuration n'est pas cohérent. Pendant la création de l'environnement, les commandes `Container Commands` sont exécutées une fois que la source de l'application est déployée dans IIS. Lors d'un déploiement dans un environnement en cours d'exécution, les commandes de conteneur sont exécutées avant le déploiement de la nouvelle version. Lors d'un ajustement à la hausse, les fichiers de configuration ne sont pas traités du tout.

En outre, IIS démarre avant l'exécution des commandes de conteneur. Ce comportement a conduit certains clients à mettre en place des solutions de contournement dans les commandes de conteneur. Elles consistent à mettre en veille le serveur IIS avant l'exécution des commandes et à le redémarrer une fois l'exécution terminée.

La version 1 corrige le problème d'incohérence et aligne le comportement de la plateforme Windows Server sur celui des plateformes Linux Elastic Beanstalk. Sur la plateforme v1, Elastic Beanstalk exécute toujours les commandes de conteneur avant de démarrer le serveur IIS.

Les piles de solutions de la plateforme v1 comportent la mention `v1` après la version de Windows Server :

- Windows Server 64 bits 2012 R2 v1.1.0 exécutant IIS 8.5
- Windows Server Core 64 bits 2012 R2 v1.1.0 exécutant IIS 8.5
- Windows Server 64 bits 2012 v1.1.0 exécutant IIS 8
- Windows Server 64 bits 2008 R2 v1.1.0 exécutant IIS 7.5

En outre, la plateforme v1 extrait le contenu du bundle de fichiers source de votre application dans `C:\staging\` avant d'exécuter les commandes de conteneur. Une fois que l'exécution des commandes de conteneur est terminée, le contenu de ce dossier est compressé en fichier .zip et déployé dans IIS. Ce processus vous permet de modifier le contenu du bundle source de votre application à l'aide des commandes ou d'un script avant le déploiement.

Migration à partir de versions majeures antérieures de la plateforme Windows Server

Consultez cette section pour prendre connaissance des considérations relatives à la migration avant de mettre à jour votre environnement. Pour mettre à jour votre plateforme d'environnement vers une version plus récente, consultez Mise à jour de la version de la plateforme de votre environnement Elastic Beanstalk.

De V1 vers V2

La plateforme Windows Server v2 ne prend pas en charge .NET Core 1.x et 2.0. Si vous migrez votre application à partir de Windows Server v1 vers v2 et que votre application utilise l'une de ces versions .NET Core, mettez à jour votre application vers une version .NET Core prise en charge par la v2. Pour obtenir la liste des versions prises en charge, consultez .NET sur Windows Server avec IIS dans les Plateformes AWS Elastic Beanstalk .

Si votre application utilise une Amazon Machine Image (AMI) personnalisée, créez une AMI personnalisée basée sur une AMI de plate-forme Windows Server v2. Pour en savoir plus, consultez la section Utilisation d'une image machine Amazon (AMI) personnalisée dans votre environnement Elastic Beanstalk.

> **Note**
>
> Les fonctionnalités de déploiement et de mise à jour qui sont nouvelles dans Windows Server v2 dépendent des rapports améliorés sur l'état de santé. Lorsque vous migrez un environnement vers v2, les rapports améliorés sur l'état de santé sont désactivés. Activez-les pour utiliser ces fonctionnalités. Pour plus d'informations, consultez Activation des rapports améliorés sur l'état Elastic Beanstalk.

À partir de versions antérieures à la V1

Outre les considérations concernant la migration à partir de la v1, si vous migrez votre application à partir d'une pile de solutions Windows Server antérieure à la v1 et que vous utilisez actuellement des commandes de conteneur, supprimez toutes les commandes que vous avez ajoutées pour contourner les incohérences de traitement lors de la migration vers une version plus récente. Depuis la v1, l'exécution complète des commandes de conteneur est garantie avant l'application source qui est déployée et avant le démarrage d'IIS. Vous pouvez ainsi apporter des modifications à la source dans `C:\staging` et modifier sans problème les fichiers de configuration IIS au cours de cette étape.

Par exemple, vous pouvez utiliser le AWS CLI pour télécharger un fichier DLL vers la source de votre application depuis Amazon S3 :

`.ebextensions\copy-dll.config`

```
container_commands:
  copy-dll:
    command: aws s3 cp s3://amzn-s3-demo-bucket/dlls/large-dll.dll .\lib\
```

Pour de plus amples informations sur l'utilisation des fichiers de configuration, veuillez consulter Personnalisation d'environnement avancée avec fichiers de configuration (`.ebextensions`).

Exécution de plusieurs applications et d'applications ASP.NET Core avec un manifeste de déploiement

Vous pouvez utiliser un manifeste de déploiement pour indiquer à Elastic Beanstalk comment déployer votre application. Avec cette méthode, vous n'avez pas besoin d'utiliser `MSDeploy` pour générer une solution groupée source pour une seule application ASP.NET qui s'exécute sur le chemin d'accès racine de votre site Web. À la place, vous pouvez utiliser un fichier manifeste afin d'exécuter plusieurs applications sur différents chemins. Vous pouvez également demander à Elastic Beanstalk de déployer et d'exécuter l'application avec ASP.NET Core. Vous pouvez également utiliser un manifeste de déploiement pour configurer un groupe d'applications dans lequel exécuter vos applications.

Les manifestes de déploiement ajoutent la prise en charge des applications .NET Core à Elastic Beanstalk. Vous pouvez déployer une application .NET Framework sans manifeste de déploiement. Toutefois, les applications .NET Core nécessitent un manifeste de déploiement pour s'exécuter sur Elastic Beanstalk. Lors de l'utilisation d'un manifeste de déploiement, vous créez une archive de site pour chaque application, puis regroupez les archives de site dans un second fichier ZIP contenant le manifeste de déploiement.

Les manifestes de déploiement ajoutent également la possibilité d'exécuter plusieurs applications sur des chemins d'accès différents. Un manifeste de déploiement définit un ensemble de cibles de déploiement, dotées chacune d'une archive de site et d'un chemin d'accès sur lequel elle doit être exécutée par IIS. Par exemple, vous pouvez exécuter une API web sur le chemin d'accès `/api` pour répondre aux demandes asynchrones, et une application web sur le chemin d'accès racine qui utilise l'API.

Vous pouvez utiliser un manifeste de déploiement pour configurer les sites Web IIS avec des liaisons personnalisées et des chemins physiques. Cela vous permet de configurer des sites Web qui écoutent sur des ports ou des noms d'hôtes spécifiques avant de déployer vos applications.

Vous pouvez également utiliser un manifeste de déploiement pour exécuter plusieurs applications à l'aide de groupes d'applications dans IIS ou Kestrel. Vous pouvez configurer un pool d'applications pour redémarrer périodiquement vos applications, exécuter des applications 32 bits ou utiliser une version spécifique du runtime .NET Framework.

Pour une personnalisation complète, vous pouvez écrire vos propres scripts de déploiement sous Windows PowerShell et indiquer à Elastic Beanstalk les scripts à exécuter pour installer, désinstaller et redémarrer votre application.

Les manifestes de déploiement et les fonctionnalités associées requièrent une plateforme Windows Server version 1.2.0 ou ultérieure.

Pour obtenir des informations détaillées sur toutes les options de configuration, les propriétés et les fonctionnalités avancées disponibles, telles que le fait d'ignorer les réinitialisations IIS, consultez la référence du schéma du manifeste de déploiement.

Sections

- Applications .NET core
- Exécution de plusieurs applications
- Configuration des sites Web IIS
- Utilisation du routage des demandes d'application (ARR)
- Configuration de groupes d'applications
- Définition de déploiements personnalisés
- Référence du schéma du manifeste de déploiement

Applications .NET core

Vous pouvez utiliser un manifeste de déploiement pour exécuter des applications .NET Core sur Elastic Beanstalk. .NET Core est une version multiplateforme de .NET qui inclut un outil de ligne de commande (`dotnet`). Vous pouvez l'utiliser pour générer une application, l'exécuter localement et la préparer en vue de sa publication.

Pour exécuter une application .NET Core sur Elastic Beanstalk, vous pouvez exécuter `dotnet publish` et placer la sortie dans une archive ZIP, en excluant les répertoires contenant des fichiers.

Placez l'archive de site dans un bundle de fichiers source avec un manifeste de déploiement, avec une cible de déploiement de type `aspNetCoreWeb`.

Le manifeste de déploiement suivant exécute une application .NET Core à partir d'une archive de site nommée `dotnet-core-app.zip` sur le chemin d'accès racine.

Example aws-windows-deployment-manifest.json - .NET core

```
{
  "manifestVersion": 1,
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "my-dotnet-core-app",
        "parameters": {
          "archive": "dotnet-core-app.zip",
          "iisPath": "/"
        }
      }
    ]
  }
}
```

Regroupez le manifeste et l'archive de site dans un fichier ZIP pour créer un bundle de fichiers source.

Example dotnet-core-bundle.zip

```
.
|-- aws-windows-deployment-manifest.json
`-- dotnet-core-app.zip
```

L'archive de site contient le code d'application compilé, les dépendances et le fichier `web.config`.

Example dotnet-core-app.zip

```
.
|-- Microsoft.AspNetCore.Hosting.Abstractions.dll
|-- Microsoft.AspNetCore.Hosting.Server.Abstractions.dll
|-- Microsoft.AspNetCore.Hosting.dll
|-- Microsoft.AspNetCore.Http.Abstractions.dll
|-- Microsoft.AspNetCore.Http.Extensions.dll
```

```
|-- Microsoft.AspNetCore.Http.Features.dll
|-- Microsoft.AspNetCore.Http.dll
|-- Microsoft.AspNetCore.HttpOverrides.dll
|-- Microsoft.AspNetCore.Server.IISIntegration.dll
|-- Microsoft.AspNetCore.Server.Kestrel.dll
|-- Microsoft.AspNetCore.WebUtilities.dll
|-- Microsoft.Extensions.Configuration.Abstractions.dll
|-- Microsoft.Extensions.Configuration.EnvironmentVariables.dll
|-- Microsoft.Extensions.Configuration.dll
|-- Microsoft.Extensions.DependencyInjection.Abstractions.dll
|-- Microsoft.Extensions.DependencyInjection.dll
|-- Microsoft.Extensions.FileProviders.Abstractions.dll
|-- Microsoft.Extensions.FileProviders.Physical.dll
|-- Microsoft.Extensions.FileSystemGlobbing.dll
|-- Microsoft.Extensions.Logging.Abstractions.dll
|-- Microsoft.Extensions.Logging.dll
|-- Microsoft.Extensions.ObjectPool.dll
|-- Microsoft.Extensions.Options.dll
|-- Microsoft.Extensions.PlatformAbstractions.dll
|-- Microsoft.Extensions.Primitives.dll
|-- Microsoft.Net.Http.Headers.dll
|-- System.Diagnostics.Contracts.dll
|-- System.Net.WebSockets.dll
|-- System.Text.Encodings.Web.dll
|-- dotnet-core-app.deps.json
|-- dotnet-core-app.dll
|-- dotnet-core-app.pdb
|-- dotnet-core-app.runtimeconfig.json
`-- web.config
```

Exécution de plusieurs applications

Vous pouvez exécuter plusieurs applications à l'aide d'un manifeste de déploiement en définissant plusieurs cibles de déploiement.

Le manifeste de déploiement suivant configure deux applications .NET Core. L'`WebApiSampleApp`application implémente une API Web simple et envoie des requêtes asynchrones sur le `/api` chemin. L'application `DotNetSampleApp` est une application Web qui sert les demandes à la racine du chemin d'accès.

Example aws-windows-deployment-manifest.json - plusieurs applications

```
{
```

```
  "manifestVersion": 1,
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "WebAPISample",
        "parameters": {
          "appBundle": "WebApiSampleApp.zip",
          "iisPath": "/api"
        }
      },
      {
        "name": "DotNetSample",
        "parameters": {
          "appBundle": "DotNetSampleApp.zip",
          "iisPath": "/"
        }
      }
    ]
  }
}
```

Un exemple d'application avec plusieurs applications est disponible ici :

- Ensemble de sources déployable - -v2.zip dotnet-multiapp-sample-bundle
- Code source : dotnet-multiapp-sample-source-v2.zip

Configuration des sites Web IIS

Vous pouvez configurer les sites Web IIS avec des liaisons personnalisées et des chemins physiques à l'aide du manifeste de déploiement. Cela est utile lorsque vous devez configurer des sites Web qui écoutent sur des ports spécifiques, utilisent des noms d'hôtes personnalisés ou diffusent du contenu provenant de répertoires spécifiques.

Le manifeste de déploiement suivant configure un site Web IIS personnalisé qui écoute sur HTTP avec un numéro de port spécifique et un chemin physique personnalisé :

Example aws-windows-deployment-manifest.json - Configuration du site Web IIS

```
{
  "manifestVersion": 1,
  "iisConfig": {
```

```
    "websites": [
      {
        "name": "MyCustomSite",
        "physicalPath": "C:\inetpub\wwwroot\mysite",
        "bindings": [
          {
            "protocol": "http",
            "port": 8080,
            "hostName": "mysite.local"
          }
        ]
      }
    ]
  },
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "my-dotnet-core-app",
        "parameters": {
          "appBundle": "dotnet-core-app.zip",
          "iisWebSite": "MyCustomSite",
          "iisPath": "/"
        }
      }
    ]
  }
}
```

Dans cet exemple :

- Un site Web nommé MyCustomSite « » est créé avec un chemin physique personnalisé
- Le site Web possède une liaison HTTP sur le port 8080 avec un nom d'hôte spécifique
- L'application ASP.NET Core est déployée sur ce site Web personnalisé à l'aide du paramètre `iisWebSite`

Utilisation du routage des demandes d'application (ARR)

Les modules Application Request Routing (ARR) et URL Rewrite sont préinstallés et disponibles dans Elastic Beanstalk Windows. AMIs Ces modules permettent des scénarios de routage avancés et la manipulation d'URL via la configuration IIS à l'aide d'extensions Web ou de la configuration d'applications.

L'exemple suivant montre un manifeste de déploiement simple qui configure un site Web avec un port personnalisé, combiné à une configuration ebextensions qui définit le routage ARR de base :

Example aws-windows-deployment-manifest.json - Configuration ARR simple

```
{
  "manifestVersion": 1,
  "iisConfig": {
    "websites": [
      {
        "name": "ARRSite",
        "physicalPath": "C:\\inetpub\\wwwroot\\arrsite",
        "bindings": [
          {
            "protocol": "http",
            "port": 8080,
            "hostName": "localhost"
          }
        ]
      }
    ]
  },
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "BackendApp",
        "parameters": {
          "appBundle": "backend-app.zip",
          "iisWebSite": "ARRSite",
          "iisPath": "/backend"
        }
      }
    ]
  }
}
```

La configuration ARR se fait par le biais d'ebextensions. La configuration suivante définit les règles de routage ARR de base :

Example.ebextensions/arr-config.config - Configuration ARR de base

```
files:
  "C:\\temp\\configure-arr.ps1":
```

```
    content: |
      # Enable ARR proxy at server level
      Set-WebConfigurationProperty -PSPath 'MACHINE/WEBROOT/APPHOST' -Filter
 'system.webServer/proxy' -Name 'enabled' -Value 'True'

      # Clear any existing global rules to avoid conflicts
      Clear-WebConfiguration -PSPath 'MACHINE/WEBROOT/APPHOST' -Filter
 'system.webServer/rewrite/globalRules'

      # Add global rule to route all requests to backend
      Add-WebConfigurationProperty -PSPath 'MACHINE/WEBROOT/APPHOST' `
        -Filter 'system.webServer/rewrite/globalRules' `
        -Name '.' `
        -Value @{
          name = 'Route_to_Backend'
          stopProcessing = 'True'
          match = @{ url = '^(?!backend/)(.*)' }
          action = @{
            type = 'Rewrite'
            url = 'http://localhost:8080/backend/{R:1}'
          }
        }

container_commands:
  01_configure_arr:
    command: powershell -ExecutionPolicy Bypass -File "C:\\temp\\configure-arr.ps1"
    waitAfterCompletion: 0
```

Cette configuration crée un site Web sur le port 8080 et configure ARR pour acheminer toutes les demandes entrantes vers l'application principale exécutée sur ce site.

Configuration de groupes d'applications

Vous pouvez prendre en charge plusieurs applications dans votre environnement Windows. Vous avez le choix entre deux approches :

- Vous pouvez utiliser le modèle out-of-process d'hébergement avec le serveur Web Kestrel. Avec ce modèle, vous configurez plusieurs applications pour qu'elles s'exécutent dans un groupe d'applications.
- Vous pouvez utiliser le modèle d'hébergement en cours, avec lequel vous utilisez plusieurs groupes d'applications pour exécuter plusieurs applications avec une seule application dans

chaque groupe. Si vous utilisez le serveur IIS et que vous avez besoin d'exécuter plusieurs applications, vous devez utiliser cette approche.

Pour configurer Kestrel de façon à exécuter plusieurs applications dans un groupe d'applications, ajoutez `hostingModel="OutofProcess"` dans le fichier `web.config`. Considérez les exemples suivants.

Example web.config - pour le modèle d'hébergement Kestrel out-of-process

```
<configuration>
<location path="." inheritInChildApplications="false">
<system.webServer>
<handlers>
<add
    name="aspNetCore"
    path="*" verb="*"
    modules="AspNetCoreModuleV2"
    resourceType="Unspecified" />
</handlers>
<aspNetCore
    processPath="dotnet"
    arguments=".\CoreWebApp-5-0.dll"
    stdoutLogEnabled="false"
    stdoutLogFile=".\logs\stdout"
    hostingModel="OutofProcess" />
</system.webServer>
</location>
</configuration>
```

Example aws-windows-deployment-manifest.json - applications multiples

```
{
"manifestVersion": 1,
  "deployments": {"msDeploy": [
      {"name": "Web-app1",
        "parameters": {"archive": "site1.zip",
          "iisPath": "/"
        }
      },
      {"name": "Web-app2",
        "parameters": {"archive": "site2.zip",
          "iisPath": "/app2"
```

```
        }
      }
    ]
  }
}
```

IIS ne prend pas en charge plusieurs applications dans un groupe d'applications car il utilise le modèle d'hébergement en cours. Par conséquent, vous devez configurer plusieurs applications en affectant chaque application à un groupe d'applications. En d'autres termes, n'affectez qu'une seule application à un groupe d'applications.

Vous pouvez configurer IIS pour qu'il utilise différents groupes d'applications dans le fichier `aws-windows-deployment-manifest.json`. Effectuez les mises à jour suivantes lorsque vous vous référez au fichier exemple suivant :

- Ajoutez une section `iisConfig` qui comprend une sous-section appelée `appPools`.
- Dans le bloc `appPools`, répertoriez les groupes d'applications.
- Dans la section `deployments`, définissez une section `parameters` pour chaque application.
- Pour chaque application, la section `parameters` spécifie une archive, un chemin d'accès pour l'exécuter et un `appPool` dans lequel s'exécuter.

Le manifeste de déploiement suivant configure deux groupes d'applications qui redémarrent leur application toutes les 10 minutes. En outre, ils attachent leurs applications à une application Web .NET Framework qui s'exécute au chemin spécifié.

Example aws-windows-deployment-manifest.json : une application par pool d'applications

```
{
"manifestVersion": 1,
  "iisConfig": {"appPools": [
      {"name": "MyFirstPool",
       "recycling": {"regularTimeInterval": 10}
      },
      {"name": "MySecondPool",
       "recycling": {"regularTimeInterval": 10}
      }
     ]
    },
  "deployments": {"msDeploy": [
      {"name": "Web-app1",
```

```
        "parameters": {
           "archive": "site1.zip",
           "iisPath": "/",
           "appPool": "MyFirstPool"
           }
      },
      {"name": "Web-app2",
        "parameters": {
           "archive": "site2.zip",
           "iisPath": "/app2",
           "appPool": "MySecondPool"
         }
      }
     ]
    }
}
```

Définition de déploiements personnalisés

Pour encore davantage de contrôle, vous pouvez personnaliser entièrement un déploiement d'applications en définissant un déploiement personnalisé.

Ce manifeste de déploiement indique à Elastic Beanstalk PowerShell d'exécuter des scripts en mode 32 bits. Il spécifie trois scripts : un `install` script (`siteInstall.ps1`) qui s'exécute lors du lancement et des déploiements de l'instance, un `uninstall` script (`siteUninstall.ps1`) qui s'exécute avant d'installer de nouvelles versions pendant les déploiements et un `restart` script (`siteRestart.ps1`) qui s'exécute lorsque vous sélectionnez Redémarrer le serveur d'applications dans la AWS console de gestion.

Example aws-windows-deployment-manifest.json - déploiement personnalisé

```
{
  "manifestVersion": 1,
  "deployments": {
    "custom": [
      {
        "name": "Custom site",
        "architecture" : 32,
        "scripts": {
          "install": {
            "file": "siteInstall.ps1"
          },
```

```
          "restart": {
            "file": "siteRestart.ps1"
          },
          "uninstall": {
            "file": "siteUninstall.ps1"
          }
        }
      }
    ]
  }
}
```

Incluez les artefacts requis pour exécuter l'application dans le bundle de fichiers source avec le manifeste et les scripts.

Example ustom-site-bundleC.zip

```
.
|-- aws-windows-deployment-manifest.json
|-- siteInstall.ps1
|-- siteRestart.ps1
|-- siteUninstall.ps1
`-- site-contents.zip
```

Référence du schéma du manifeste de déploiement

Le manifeste de déploiement est un fichier JSON qui définit la manière dont Elastic Beanstalk doit déployer et configurer vos applications Windows. Cette section fournit une référence complète pour toutes les propriétés et options de configuration prises en charge dans le schéma du manifeste.

Structure du manifeste

Le manifeste de déploiement suit un schéma JSON spécifique avec la structure de haut niveau suivante :

Example Structure de base du manifeste

```
{
  "manifestVersion": 1,
  "skipIISReset": false,
  "iisConfig": {
```

```
    "websites": [...],
    "appPools": [...]
  },
  "deployments": {
    "msDeploy": [...],
    "aspNetCoreWeb": [...],
    "custom": [...]
  }
}
```

Propriétés de haut niveau

`manifestVersion` (obligatoire)

Type : nombre

Par défaut : 1

Valeurs valides : 1

Spécifie la version du schéma du manifeste. Actuellement, seule la version 1 est prise en charge.

`skipIISReset` (facultatif)

Type : valeur booléenne

Par défaut : faux

Contrôle si IIS est réinitialisé lors des déploiements d'applications. Cet indicateur affecte à la fois les types de `aspNetCoreWeb` déploiement `msDeploy` et les types de déploiement.

Comportement :

- Non spécifié ou *`false`* (par défaut) : les réinitialisations IIS sont effectuées lors des opérations d'installation, de désinstallation et de mise à jour. C'est le comportement traditionnel.
- *`true`*: les réinitialisations IIS sont ignorées lors des opérations de déploiement.

Avantages :

- Réduction des temps d'arrêt : les applications subissent des interruptions de service plus courtes pendant les déploiements.
- Déploiements plus rapides : élimine le temps nécessaire au redémarrage et à la réinitialisation complets d'IIS.

> **Note**
>
> Lors de l'utilisation`skipIISReset`, l'RestartAppServeropération effectue une réinitialisation IIS quel que soit ce paramètre d'indicateur.

Exemple :

```
{
  "manifestVersion": 1,
  "skipIISReset": true,
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "my-dotnet-core-app",
        "parameters": {
          "archive": "dotnet-core-app.zip",
          "iisPath": "/"
        }
      }
    ]
  }
}
```

`deployments` (obligatoire)

Type : objet

Contient les configurations de déploiement pour vos applications. Cet objet peut inclure `msDeployaspNetCoreWeb`, et des types de `custom` déploiement.

`iisConfig` (facultatif)

Type : objet

Définit les paramètres de configuration IIS à appliquer avant de déployer des applications. Prend en charge la configuration du site Web et du pool d'applications.

Configuration d'IIS

Cette `iisConfig` section vous permet de configurer les paramètres IIS avant de déployer vos applications. Cela inclut la configuration de pools d'applications avec des configurations spécifiques et la configuration de sites Web IIS avec des liaisons personnalisées.

Sites Web IIS

Les sites Web IIS vous permettent de configurer des paramètres de site Web personnalisés, notamment des chemins physiques et des liaisons réseau, avant de déployer vos applications.

> Considérations importantes relatives à la création de différents sites Web IIS
>
> - Ordre de configuration des sites Web : les sites Web sont configurés séquentiellement dans l'ordre dans lequel ils apparaissent dans le `websites` tableau. La plateforme traite la configuration de chaque site Web dans l'ordre. Veillez donc à ce que l'ordre soit correct si vous avez des dépendances entre les sites Web.
> - Pare-feu et accès aux ports : seul le port 80 est automatiquement exposé via la configuration par défaut du pare-feu Windows Elastic Beanstalk. Si vous configurez des sites Web pour utiliser des ports non standard, vous devez définir des règles de pare-feu personnalisées par le biais d'ebextensions ou de scripts de déploiement personnalisés pour autoriser l'accès externe à ces ports.

Example Configuration du site Web

```
{
  "iisConfig": {
    "websites": [
      {
        "name": "MyCustomSite",
        "physicalPath": "C:\inetpub\wwwroot\mysite",
        "bindings": [
          {
            "protocol": "http",
            "port": 8080,
            "hostName": "mysite.local"
          },
          {
            "protocol": "https",
```

```
            "port": 8443
          }
        ]
      }
    ]
  }
}
```

Propriétés du site Web

`name` (obligatoire)

Type : chaîne

Nom du site Web IIS. Ce nom est utilisé pour identifier le site Web dans le Gestionnaire des services Internet et doit être unique dans la configuration IIS.

`physicalPath` (obligatoire)

Type : chaîne

Le chemin physique sur le serveur où les fichiers du site Web sont stockés. Ce chemin doit être accessible au processus de travail IIS.

`bindings` (obligatoire)

Type : tableau

Nombre minimum d'articles : 1

Ensemble de configurations de liaison qui définissent la manière dont le site Web répond aux demandes du réseau. Chaque liaison spécifie un protocole, un port et un nom d'hôte facultatif.

Liaisons du site Web

Les liaisons de sites Web définissent les points de terminaison du réseau sur lesquels votre site Web IIS écoutera les demandes entrantes.

`protocol` (obligatoire)

Type : chaîne

Valeurs valides : « http », « https »

Protocole utilisé pour la liaison.

`port` (obligatoire)

Type : entier

Plage valide : 1-65535

Numéro de port sur lequel le site Web écoutera les demandes.

`hostName` (facultatif)

Type : chaîne

Le nom d'hôte (nom de domaine) pour la liaison.

Pools d'applications

Les pools d'applications isolent les applications et vous permettent de configurer les paramètres d'exécution pour des groupes d'applications.

Example Configuration du pool d'applications

```
{
  "iisConfig": {
    "appPools": [
      {
        "name": "MyAppPool",
        "enable32Bit": false,
        "managedPipelineMode": "Integrated",
        "managedRuntimeVersion": "v4.0",
        "queueLength": 1000,
        "cpu": {
          "limitPercentage": 80,
          "limitAction": "Throttle",
          "limitMonitoringInterval": 5
        },
        "recycling": {
          "regularTimeInterval": 1440,
          "requestLimit": 10000,
          "memory": 1048576,
          "privateMemory": 524288
        }
      }
```

```
    ]
  }
}
```

Propriétés du pool d'applications

`name` (obligatoire)

Type : chaîne

Nom du pool d'applications. Ce nom est utilisé pour faire référence au pool dans les configurations de déploiement.

`enable32Bit` (facultatif)

Type : valeur booléenne

Permet à une application 32 bits de s'exécuter sur une version 64 bits de Windows. Définissez cette option `true` pour les applications existantes qui nécessitent une compatibilité 32 bits.

`managedPipelineMode` (facultatif)

Type : chaîne

Valeurs valides : « Intégré », « Classique »

Spécifie le mode de traitement des demandes pour le pool d'applications.

`managedRuntimeVersion` (facultatif)

Type : chaîne

Valeurs valides : « Aucun code géré », « v2.0 », « v4.0 »

Spécifie la version du .NET Framework pour le pool d'applications.

`queueLength` (facultatif)

Type : entier

Nombre maximal de demandes que HTTP.sys met en file d'attente pour le pool d'applications avant de rejeter les demandes supplémentaires.

Configuration du processeur

L'`cpu`objet configure les limites d'utilisation du processeur et la surveillance du pool d'applications.

`limitPercentage` (facultatif)

Type : nombre

Pourcentage maximal de temps processeur que les processus de travail peuvent consommer dans le pool d'applications.

`limitAction` (facultatif)

Type : chaîne

Valeurs valides : "NoAction«, « KillW3WP », « Throttle », "» ThrottleUnderLoad

Action à entreprendre lorsque la limite du processeur est atteinte.

`limitMonitoringInterval` (facultatif)

Type : nombre

Période de réinitialisation (en minutes) pour la surveillance du processeur et les limites de régulation.

Configuration du recyclage

L'`recycling`objet configure quand et comment les processus de travail du pool d'applications sont recyclés.

`regularTimeInterval` (facultatif)

Type : entier

Intervalle de temps (en minutes) après lequel le pool d'applications recycle. Réglez sur 0 pour désactiver le recyclage basé sur le temps.

`requestLimit` (facultatif)

Type : entier

Nombre maximal de demandes traitées par le pool d'applications avant le recyclage.

`memory` (facultatif)

Type : entier

Quantité de mémoire virtuelle (en kilo-octets) qui déclenche le recyclage des processus de travail.

`privateMemory` (facultatif)

Type : entier

Quantité de mémoire privée (en kilo-octets) qui déclenche le recyclage des processus de travail.

Types de déploiement

L'`deployments`objet contient des tableaux de configurations de déploiement pour différents types d'applications. Chaque type de déploiement possède des propriétés et des cas d'utilisation spécifiques.

MSDeploy déploiements

MSDeploy les déploiements sont utilisés pour les applications .NET Framework traditionnelles qui peuvent être déployées à l'aide de Web Deploy (MSDeploy).

Example MSDeploy configuration de déploiement

```
{
  "deployments": {
    "msDeploy": [
      {
        "name": "WebApp",
        "description": "Main web application",
        "parameters": {
          "appBundle": "webapp.zip",
          "iisPath": "/",
          "appPool": "DefaultAppPool"
        }
      }
    ]
  }
}
```

MSDeploy propriétés de déploiement

`name` (obligatoire)

Type : chaîne

Nom unique pour le déploiement. Ce nom doit être unique pour tous les déploiements du manifeste.

`description` (facultatif)

Type : chaîne

Description lisible par l'homme du déploiement.

`parameters` (obligatoire)

Type : objet

Paramètres de configuration de l' MSDeploy opération.

`scripts` (facultatif)

Type : objet

PowerShell scripts à exécuter à différentes étapes du cycle de vie du déploiement.

MSDeploy paramètres

`appBundle` (obligatoire)

Type : chaîne

Chemin d'accès au bundle d'applications (fichier ZIP) relatif au fichier manifeste. Ce bundle contient les fichiers d'application à déployer.

`iisWebSite` (facultatif)

Type : chaîne

Par défaut : « Site Web par défaut »

Le site Web IIS sur lequel déployer l'application. Par défaut, les applications sont déployées sur le « site Web par défaut ». Vous pouvez éventuellement spécifier un autre nom de site Web, tel que celui configuré dans la `iisConfig.websites` section.

`iisPath` (facultatif)

Type : chaîne

Par défaut : «/»

Chemin du répertoire virtuel dans IIS où l'application sera déployée. Utilisez «/» pour le chemin racine ou « /api » pour un sous-répertoire.

`appPool` (facultatif)

Type : chaîne

Nom du pool d'applications pour exécuter cette application.

Déploiements ASP.NET Core

Les déploiements ASP.NET Core sont spécialement conçus pour les applications .NET Core et .NET 5+.

Example Configuration du déploiement d'ASP.NET Core

```
{
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "CoreAPI",
        "description": "ASP.NET Core Web API",
        "parameters": {
          "appBundle": "coreapi.zip",
          "iisPath": "/api",
          "appPool": "CoreAppPool"
        }
      }
    ]
  }
}
```

Les déploiements ASP.NET Core utilisent la même structure de propriétés que MSDeploy les déploiements, la principale différence étant l'environnement d'exécution et le modèle d'hébergement utilisés pour l'application.

Paramètres de déploiement d'ASP.NET Core

`appBundle` (obligatoire)

Type : chaîne

Chemin d'accès au bundle d'applications relatif au fichier manifeste. Il peut s'agir d'une archive ZIP ou d'un chemin de répertoire contenant l'application ASP.NET Core publiée.

`iisWebSite` (facultatif)

Type : chaîne

Par défaut : « Site Web par défaut »

Le site Web IIS sur lequel déployer l'application ASP.NET Core. Par défaut, les applications sont déployées sur le « site Web par défaut ». Vous pouvez éventuellement spécifier un autre nom de site Web, tel que celui configuré dans la `iisConfig.websites` section.

`iisPath` (facultatif)

Type : chaîne

Par défaut : «/»

Chemin du répertoire virtuel dans IIS pour l'application ASP.NET Core.

`appPool` (facultatif)

Type : chaîne

Pool d'applications pour l'application ASP.NET Core. Le pool sera configuré de manière appropriée pour l'hébergement ASP.NET Core.

Déploiements personnalisés

Les déploiements personnalisés permettent de contrôler totalement le processus de déploiement par le biais de PowerShell scripts. Ce type de déploiement est utile pour les scénarios complexes qui nécessitent une installation, une configuration ou une logique de déploiement personnalisées.

Example Configuration de déploiement personnalisée

```
{
  "deployments": {
    "custom": [
      {
        "name": "CustomService",
        "description": "Custom Windows service deployment",
        "architecture": 32,
```

```
        "scripts": {
          "install": {
            "file": "install-service.ps1"
          },
          "restart": {
            "file": "restart-service.ps1"
          },
          "uninstall": {
            "file": "uninstall-service.ps1",
            "ignoreErrors": true
          }
        }
      }
    ]
  }
}
```

Propriétés de déploiement personnalisées

`name` (obligatoire)

Type : chaîne

Nom unique pour le déploiement personnalisé.

`description` (facultatif)

Type : chaîne

Description du déploiement personnalisé.

`architecture` (facultatif)

Type : entier

Par défaut : 32

Valeurs valides : 32, 64

La spécification d'architecture pour le mode d'exécution des scripts PowerShell

`scripts` (obligatoire)

Type : objet

PowerShell des scripts qui définissent le comportement de déploiement. Les déploiements personnalisés prennent en charge des types de scripts supplémentaires par rapport aux autres types de déploiement.

Scripts de déploiement

Les scripts de déploiement sont PowerShell des scripts qui s'exécutent à des moments spécifiques du cycle de vie du déploiement. Les différents types de déploiement prennent en charge différents ensembles d'événements de script.

Événements liés au script

Les événements de script suivants sont disponibles en fonction du type de déploiement :

Scripts de déploiement standard (MSDeploy et aspNetCore Web)

`preInstall`

S'exécute avant l'installation ou la mise à jour de l'application.

`postInstall`

S'exécute après l'installation ou la mise à jour de l'application.

`preRestart`

S'exécute avant le redémarrage de l'application.

`postRestart`

S'exécute après le redémarrage de l'application.

`preUninstall`

S'exécute avant la désinstallation de l'application.

`postUninstall`

S'exécute après la désinstallation de l'application.

Scripts de déploiement personnalisés (déploiements personnalisés uniquement)

`install`

Script d'installation principal pour les déploiements personnalisés. Ce script est responsable de l'installation de l'application ou du service.

`restart`

Script pour redémarrer l'application ou le service. Appelé au redémarrage de l'environnement.

`uninstall`

Script permettant de désinstaller l'application ou le service. Appelé lors de la fermeture de l'environnement ou de la suppression d'une application.

Propriétés du script

Chaque script est défini comme un objet doté des propriétés suivantes :

`file` (obligatoire)

Type : chaîne

Chemin d'accès au fichier de PowerShell script relatif au fichier manifeste. Le script doit avoir une `.ps1` extension.

`ignoreErrors` (facultatif)

Type : valeur booléenne

Par défaut : faux

Lorsque ce paramètre est défini sur`true`, le déploiement se poursuit même si le script échoue. Utilisez-le pour des scripts ou des opérations de nettoyage non critiques.

Example Exemple de configuration de script

```
{
  "scripts": {
    "preInstall": {
      "file": "backup-config.ps1",
      "ignoreErrors": true
    },
    "postInstall": {
      "file": "configure-app.ps1"
    }
  }
}
```

Utilisation de EC2 Fast Launch avec les branches de la plate-forme Windows

La fonctionnalité EC2 Fast Launch réduit les temps de lancement des instances Windows dans vos environnements Elastic Beanstalk. L'objectif de cette rubrique est de vous expliquer comment utiliser cette fonctionnalité dans vos environnements Elastic Beanstalk. À partir de la version 2.16.2 de la plateforme Windows, publiée le 22 janvier 2025, les versions de la plateforme Elastic Beanstalk incluent la version de base avec Fast Launch activé. AMIs EC2

Disponibilité du lancement EC2 rapide par défaut

Les dernières versions de la plateforme Windows Elastic Beanstalk AMIs incluent la version de EC2 base avec Fast Launch automatiquement activé, sans frais supplémentaires. Toutefois, lorsque de nouvelles versions de plate-forme sont publiées, EC2 Fast Launch peut ne pas rester automatiquement activé sur la base AMIs des anciennes versions de plate-forme.

Nous vous recommandons de passer à la dernière version de la plateforme Windows pour utiliser la base AMIs avec EC2 Fast Launch automatiquement activé. Toutefois, si vous devez continuer à utiliser la version de votre plateforme existante, vous pouvez activer manuellement EC2 Fast Launch sur l'AMI de base de votre environnement. Pour obtenir des instructions, veuillez consulter Configuration manuelle de EC2 Fast Launch.

Configuration manuelle de EC2 Fast Launch

> **Note**
>
> L'activation manuelle de EC2 Fast Launch peut entraîner des coûts supplémentaires par rapport à l'utilisation de versions de plateforme avec EC2 Fast Launch automatiquement activé. Pour plus d'informations sur les coûts de EC2 Fast Launch, consultez la page des ressources sous-jacentes de EC2 Fast Launch dans le guide de EC2 l'utilisateur Amazon.

Procédez comme suit pour activer EC2 Fast Launch sur une AMI de base Windows utilisée par votre environnement Elastic Beanstalk :

Pour activer manuellement EC2 Fast Launch pour votre environnement Elastic Beanstalk

1. Identifiez l'AMI de base de votre environnement :

Suivez les étapes décrites dans Création d'une AMI personnalisée pour identifier l'ID d'AMI de base de votre environnement. Notez qu'il n'est pas nécessaire de créer une AMI personnalisée, il vous suffit de suivre les étapes pour localiser votre ID d'AMI de base actuel.

2. Activez le lancement EC2 rapide sur l'AMI :

   Suivez les instructions de la section Activer le lancement EC2 rapide du guide de EC2 l'utilisateur Amazon pour configurer EC2 Fast Launch pour votre AMI.

Ajout d'une instance de base de données Amazon RDS à votre environnement d'application .NET

Cette rubrique fournit des instructions pour créer un Amazon RDS à l'aide de la console Elastic Beanstalk. Vous pouvez utiliser une instance de base de données Amazon Relational Database Service (Amazon RDS) pour stocker les données collectées et modifiées par votre application. La base de données peut être liée à votre environnement et gérée par Elastic Beanstalk, ou elle peut être créée comme déliée et gérée en externe par un autre service. Dans ces instructions, la base de données est couplée à votre environnement et gérée par Elastic Beanstalk. Pour plus d'informations sur l'intégration d'un Amazon RDS avec Elastic Beanstalk, consultez Ajout d'une base de données à votre environnement Elastic Beanstalk.

Sections

- Ajout d'une instance de base de données à votre environnement
- Téléchargement d'un pilote
- Connexion à une base de données

Ajout d'une instance de base de données à votre environnement

Pour ajouter une instance DB à votre environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.

4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

L'ajout d'une instance DB prend environ 10 minutes. Une fois la mise à jour de l'environnement terminée, le nom d'hôte de l'instance DB et les autres informations de connexion sont disponibles dans votre application, via les propriétés d'environnement suivantes :

Nom de la propriété	Description	Valeur de la propriété
`RDS_HOSTNAME`	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
`RDS_PORT`	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
`RDS_DB_NAME`	Nom de la base de données, **`ebdb`**.	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
`RDS_USERNAME`	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
`RDS_PASSWORD`	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

Pour plus d'informations sur la configuration d'une instance de base de données liée à un environnement Elastic Beanstalk, consultez Ajout d'une base de données à votre environnement Elastic Beanstalk.

Téléchargement d'un pilote

Téléchargez et installez le package `EntityFramework` et un pilote de base de données pour votre environnement de développement avec `NuGet`.

Fournisseurs de base de données Common Entity Framework pour .NET

- SQL Server – `Microsoft.EntityFrameworkCore.SqlServer`
- MySQL – `Pomelo.EntityFrameworkCore.MySql`
- PostgreSQL – `Npgsql.EntityFrameworkCore.PostgreSQL`

Connexion à une base de données

Elastic Beanstalk fournit des informations de connexion pour les instances de base de données attachées dans les propriétés de l'environnement. Utilisez `ConfigurationManager.AppSettings` pour lire les propriétés et configurer une connexion de base de données.

Example Helpers.cs - Méthode de chaîne de connexion

```
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Linq;
using System.Web;

namespace MVC5App.Models
{
  public class Helpers
  {
    public static string GetRDSConnectionString()
    {
      var appConfig = ConfigurationManager.AppSettings;

      string dbname = appConfig["RDS_DB_NAME"];

      if (string.IsNullOrEmpty(dbname)) return null;
```

```
      string username = appConfig["RDS_USERNAME"];
      string password = appConfig["RDS_PASSWORD"];
      string hostname = appConfig["RDS_HOSTNAME"];
      string port = appConfig["RDS_PORT"];

      return "Data Source=" + hostname + ";Initial Catalog=" + dbname + ";User ID=" +
 username + ";Password=" + password + ";";
    }
  }
}
```

Utilisez la chaîne de connexion pour initialiser votre contexte de base de données.

Example DBContext.cs

```
using System.Data.Entity;
using System.Security.Claims;
using System.Threading.Tasks;
using Microsoft.AspNet.Identity;
using Microsoft.AspNet.Identity.EntityFramework;

namespace MVC5App.Models
{
  public class RDSContext : DbContext
  {
    public RDSContext()
      : base(GetRDSConnectionString())
    {
    }

    public static RDSContext Create()
    {
      return new RDSContext();
    }
  }
}
```

Le AWS Toolkit for Visual Studio

Visual Studio fournit des modèles pour différents langages de programmation et types d'applications. Vous pouvez commencer avec n'importe lequel de ces modèles. Le AWS Toolkit for Visual Studio

fournit également trois modèles de projet qui démarrent le développement de votre application : AWS Console Project, AWS Web Project et AWS Empty Project. Pour cet exemple, vous allez créer une application web ASP.NET.

Pour créer un projet d'application web ASP.NET

1. Dans Visual Studio, dans le menu Fichier, cliquez sur Nouveau puis cliquez sur Projet.
2. Dans la boîte de dialogue Nouveau projet, cliquez sur Modèles installés, cliquez sur Visual C#, puis cliquez sur Web. Cliquez sur Application Web ASP.NET vide, tapez un nom de projet puis cliquez sur OK.

Pour exécuter un projet

Effectuez l'une des actions suivantes :

1. Appuyez sur F5.
2. Sélectionnez Démarrer le débogage dans le menu Déboguer.

Test local

Visual Studio vous facilite le test de votre application localement. Pour tester ou exécuter des applications web ASP.NET, vous avez besoin d'un serveur web. Visual Studio fournit plusieurs options, telles qu'Internet Information Services (IIS), IIS Express ou le serveur de développement intégré de Visual Studio. Pour en savoir plus sur chacune de ces options et décider laquelle vous convient le mieux, consultez Web Servers in Visual Studio for ASP.NET Web Projects.

Créer un environnement Elastic Beanstalk

Après avoir testé votre application, vous êtes prêt à la déployer dans Elastic Beanstalk.

> **Note**
>
> Un fichier de configuration doit faire partie du projet pour être inclus dans l'archive. Sinon, au lieu d'inclure les fichiers de configuration dans le projet, vous pouvez utiliser Visual Studio pour déployer tous les fichiers dans le dossier de projet. Dans l'Explorateur de solutions, cliquez avec le bouton droit sur le nom du projet, puis cliquez sur Propriétés. Cliquez sur l'onglet Package/Publication Web. Dans la section Items to deploy (Éléments à déployer),

sélectionnez All Files in the Project Folder (Tous les fichiers dans le dossier du projet) dans la liste déroulante.

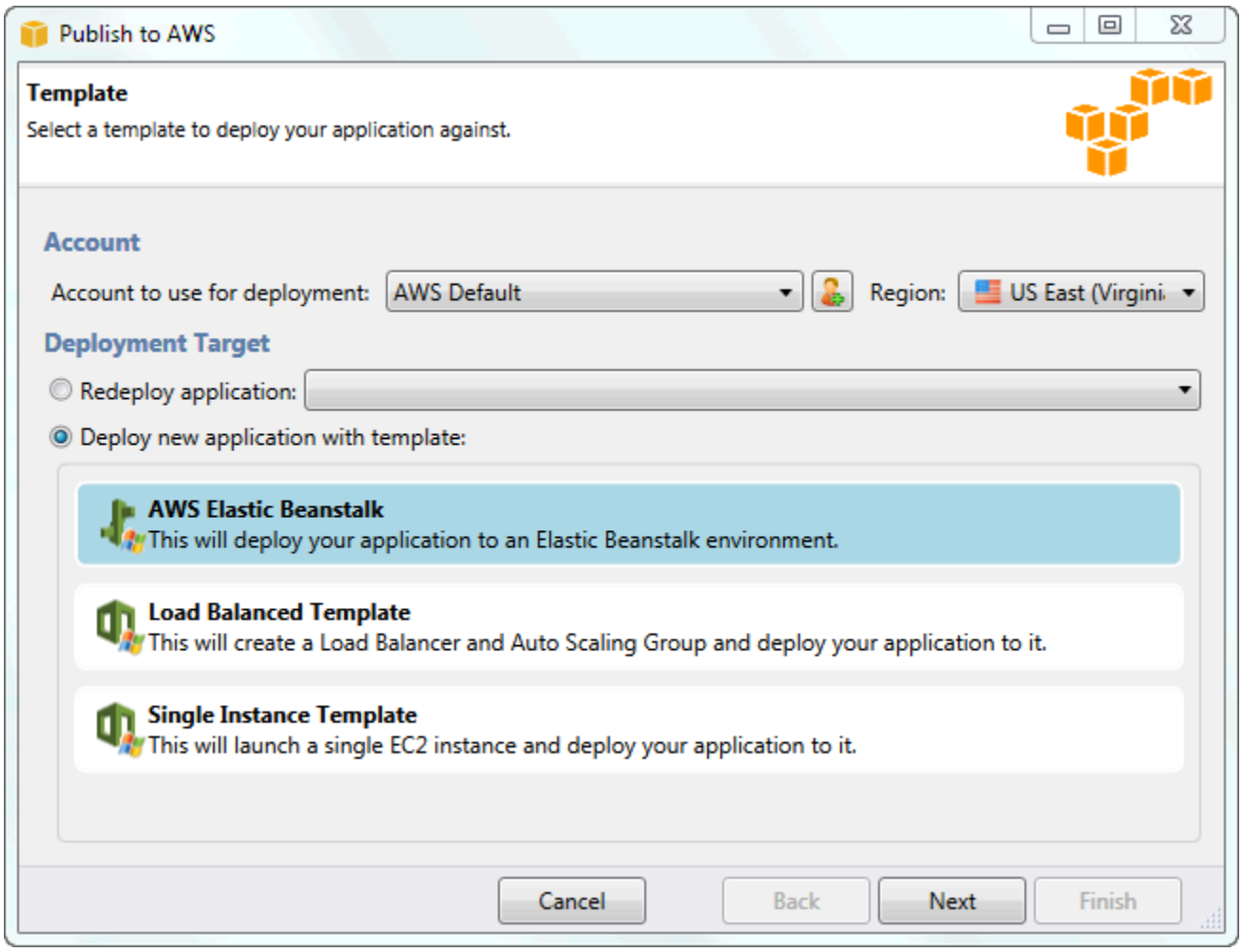
Pour déployer votre application sur Elastic Beanstalk AWS à l'aide du kit d'outils pour Visual Studio

1. Dans l'Explorateur de solutions, cliquez avec le bouton droit sur votre application, puis sélectionnez Publier sur AWS.
2. Dans l'assistant Publish to AWS (Publier sur AWS), entrez les informations de votre compte.

   a. Pour AWS account to use for deployment (Compte AWS à utiliser pour le déploiement), sélectionnez votre compte ou Other (Autre) pour entrer de nouvelles informations de compte.

   b. Pour Region (Région), sélectionnez la région où vous souhaitez déployer votre application. Pour plus d'informations sur AWS les régions disponibles, consultez la section AWS Elastic Beanstalk Points de terminaison et quotas dans le Références générales AWS. Si vous sélectionnez une région qui n'est pas prise en charge par Elastic Beanstalk, l'option de déploiement vers Elastic Beanstalk devient indisponible.

   c. Cliquez sur Deploy new application with template (Déployer la nouvelle application avec le modèle) puis sur Elastic Beanstalk. Ensuite, cliquez sur Suivant.



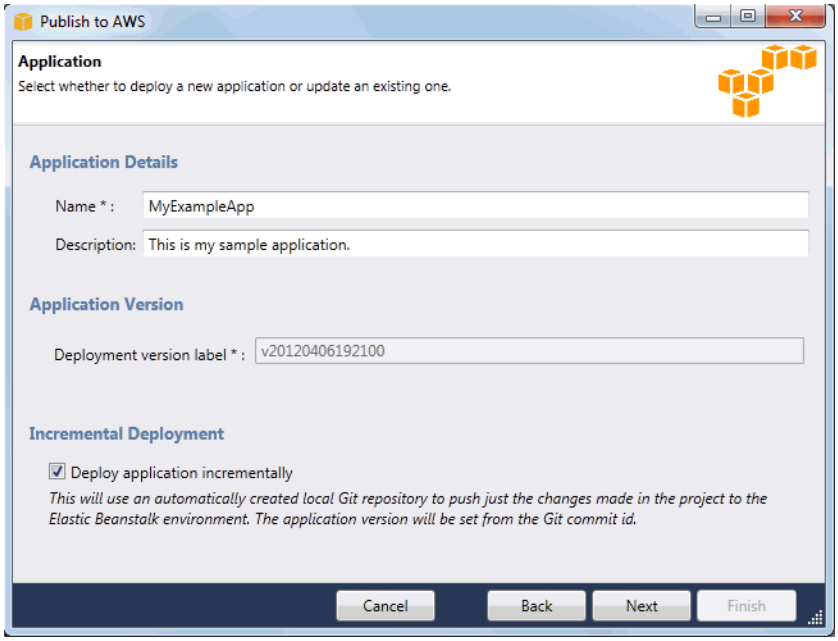
3. Sur la page Application, entrez les détails de votre application.

   a. Pour Name (Nom), tapez le nom de l'application.

   b. Pour Description, entrez une description de l'application. Cette étape est facultative.

   c. L'étiquette de version de l'application s'affiche automatiquement dans l'étiquette de version Deployment.

   d. Sélectionnez Deploy application incrementally (Déployer l'application de manière incrémentielle) pour déployer uniquement les fichiers modifiés. Un déploiement incrémentiel est plus rapide, car vous mettez à jour uniquement les fichiers qui ont été modifiés au lieu de tous les fichiers. Si vous sélectionnez cette option, une version de l'application est définie à partir de l'ID de validation Git. Si vous choisissez de ne pas déployer votre application de manière incrémentielle, vous pouvez alors mettre à jour l'étiquette de version dans la zone Deployment version label (Étiquette de version Déploiement).



e. Cliquez sur Next (Suivant).

4. Sur la page Environment (Environnement), décrivez les détails de votre environnement.

   a. Sélectionnez Create a new environment for this application (Créer un environnement pour cette application).

   b. Pour Name (Nom), tapez un nom unique pour votre environnement.

   c. Pour Description, qualifiez votre environnement. Cette étape est facultative.

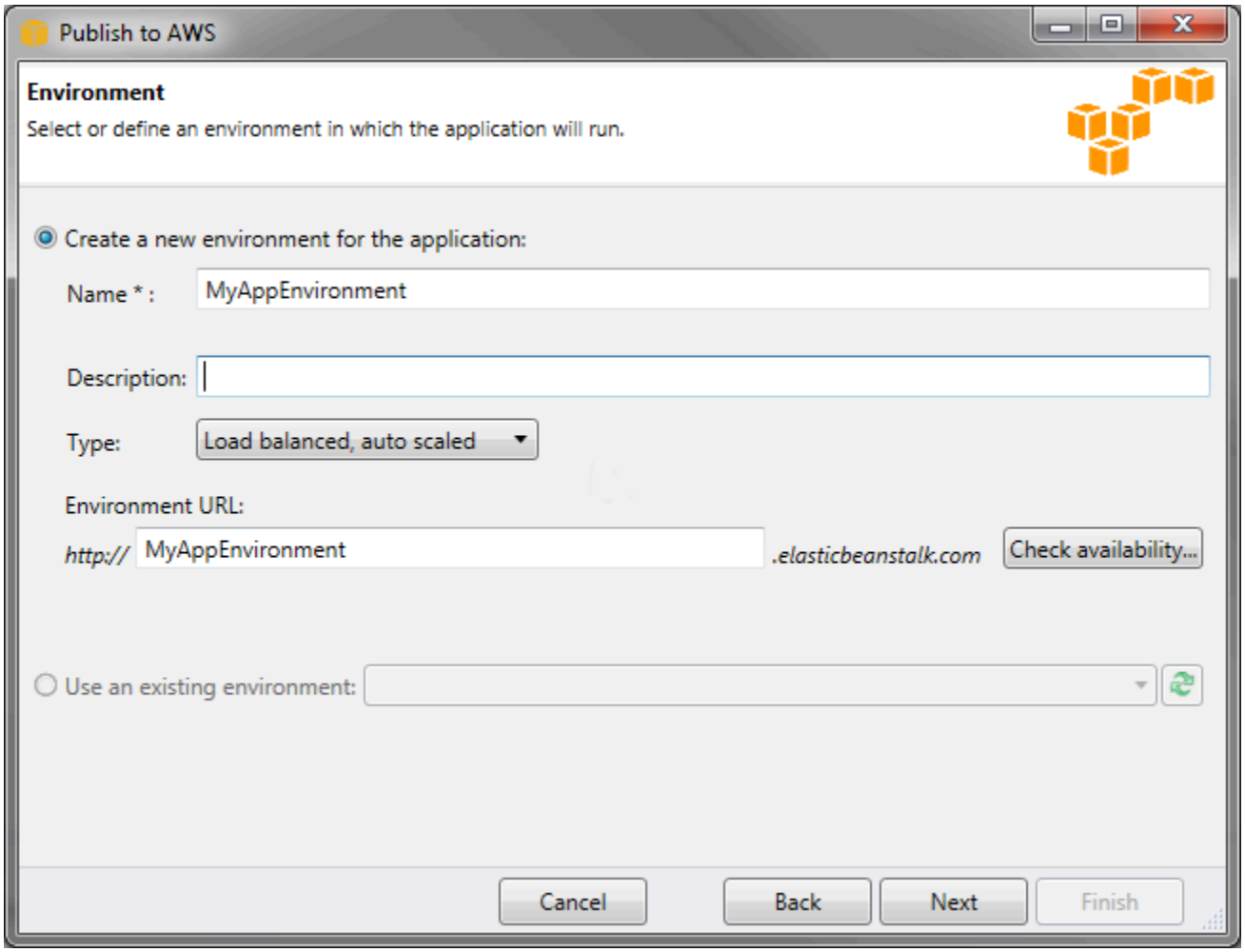
   d. Sélectionnez le Type d'environnement que vous voulez.

      Vous avez le choix entre deux types d'environnement : Load balanced, auto scaled (Équilibrage de charge, scalabilité automatique) ou Single instance (Instance unique). Pour plus d'informations, consultez Types d'environnement.

      > **Note**
      >
      > Pour les environnements instance unique, les paramètres URL d'équilibrage de charge, de mise à l'échelle automatique et de vérification de l'état ne s'appliquent pas.

   e. L'URL d'environnement s'affiche automatiquement dans Environment URL (URL de l'environnement) une fois que vous déplacez votre curseur dans cette zone.

f. Cliquez sur Check availability (Vérifier la disponibilité) pour vous assurer que l'URL d'environnement est disponible.



g. Cliquez sur Suivant.

5. Sur la page AWS Options (Options AWS), configurez des informations de sécurité et des options supplémentaires pour votre déploiement.

   a. Pour Container Type (Type de conteneur), sélectionnez 64bit Windows Server 2012 running IIS 8 (Serveur Windows 2012 64 bits exécutant IIS 8) ou 64bit Windows Server 2008 running IIS 7.5 (Serveur Windows 2008 64 bits exécutant IIS 7.5).

   b. Pour Instance Type (Type d'instance), sélectionnez Micro.

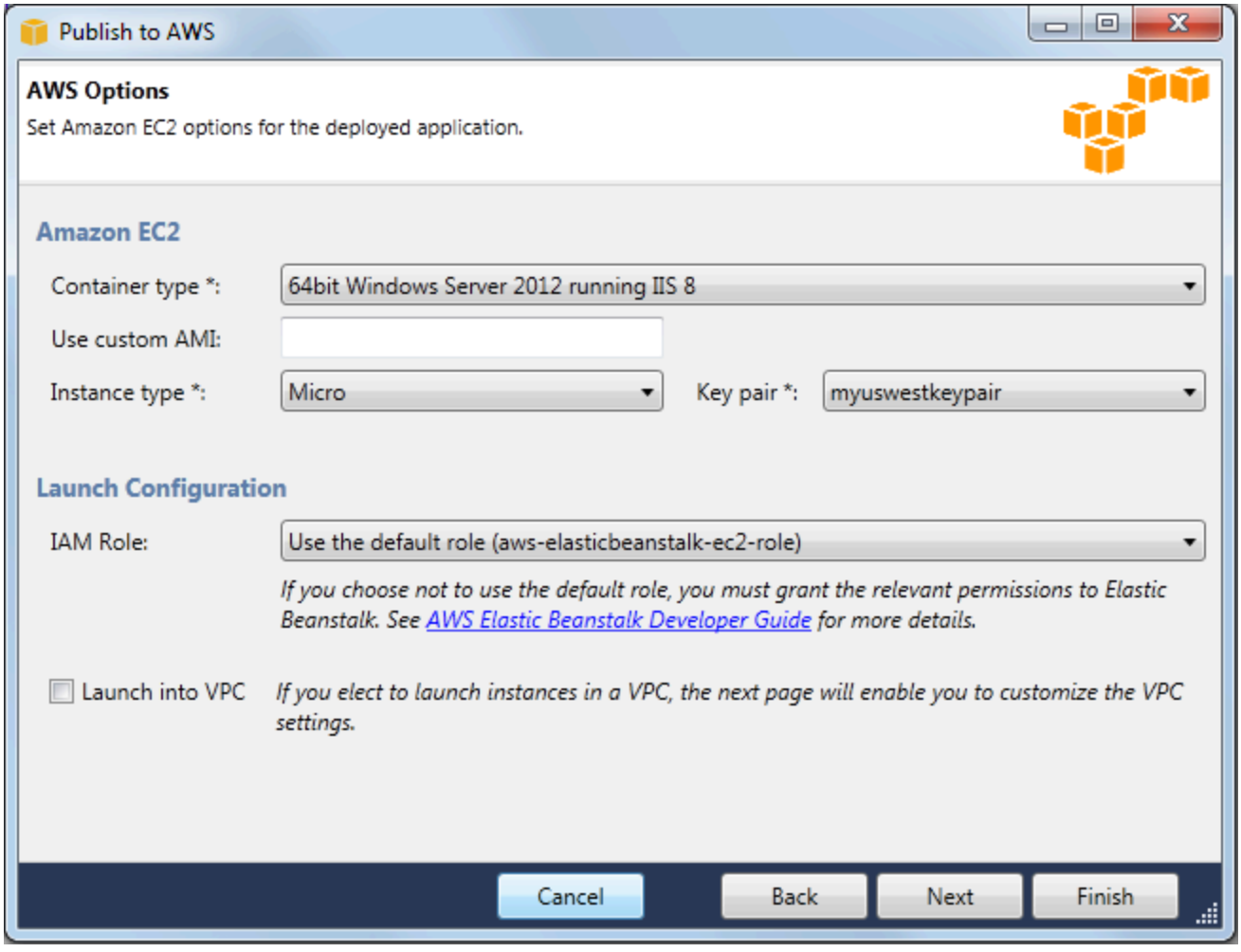
   c. Pour Key pair (Paire de clés), sélectionnez Create new key pair (Créer une paire de clés). Tapez un nom pour la nouvelle paire de clés. Dans cet exemple, nous utiliserons **`myuswestkeypair`**, puis nous cliquerons sur OK. Une paire de clés permet d'accéder à distance à vos instances Amazon EC2 . Pour plus d'informations sur les paires de EC2 clés

Amazon, consultez la section Utilisation des informations d'identification dans le guide de l'utilisateur d'Amazon Elastic Compute Cloud.

d. Sélectionner un profil d'instance.

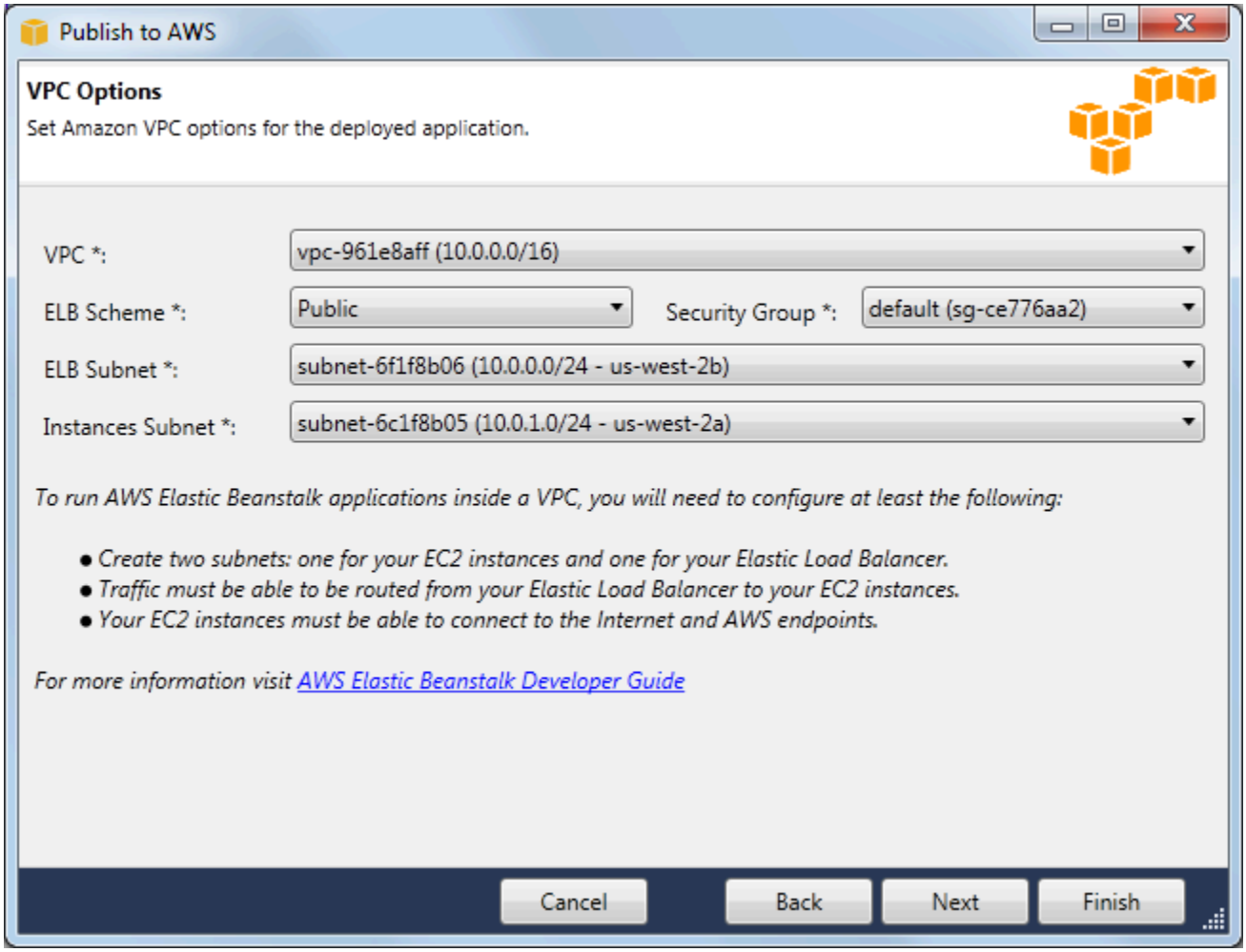
   Si vous n'avez pas de profil d'instance, sélectionnez Create a default instance profile (Créer un profil d'instance par défaut). Pour de plus amples informations sur l'utilisation des profils d'instance avec Elastic Beanstalk, veuillez consulter Gestion des profils d'instance Elastic Beanstalk.

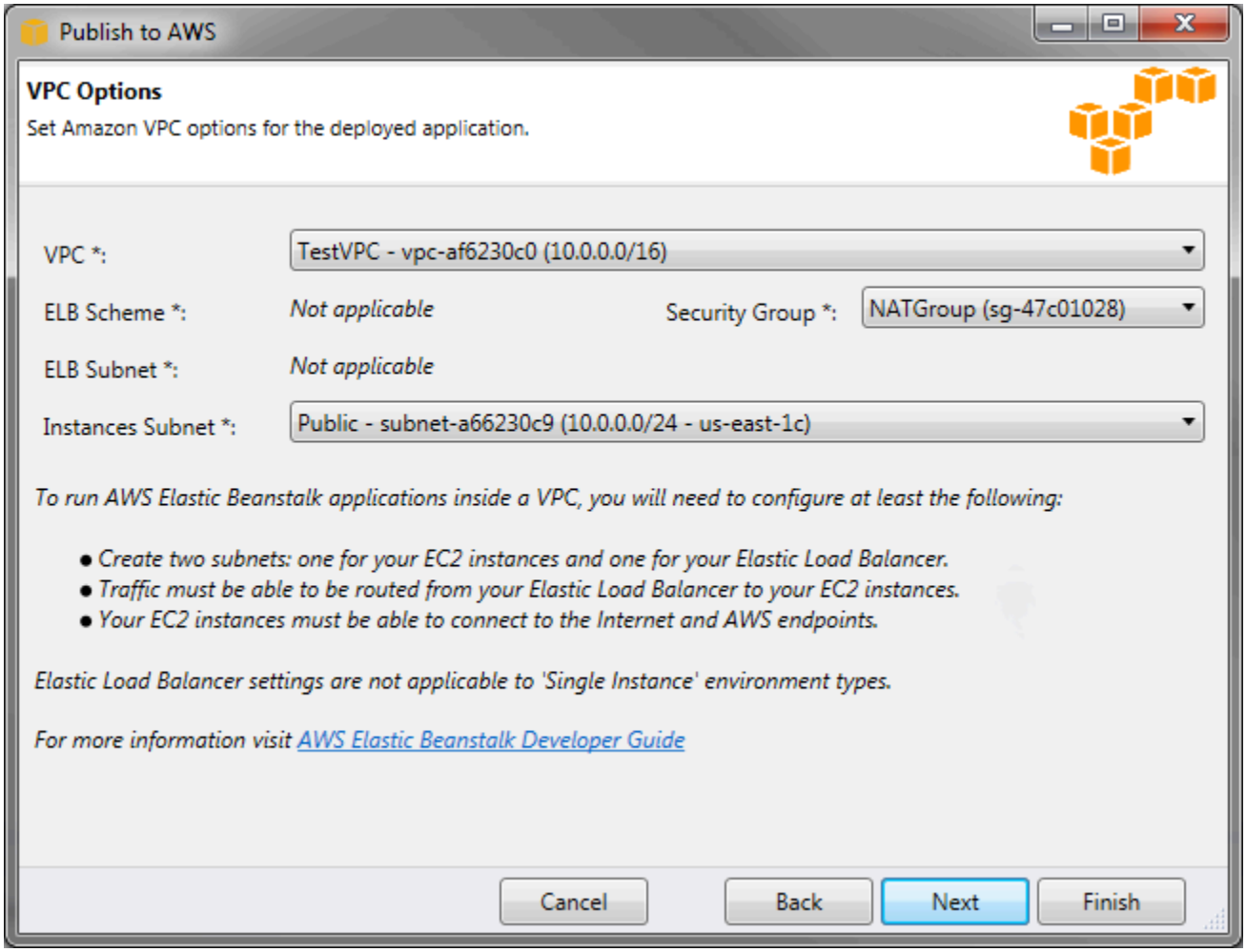
e. Si vous avez un VPC personnalisé que vous souhaitez utiliser avec votre environnement, cliquez sur Launch into VPC (Lancer dans le VPC). Vous pouvez configurer les informations de VPC sur la page suivante. Pour de plus amples informations sur Amazon VPC, veuillez consulter Amazon Virtual Private Cloud (Amazon VPC). Pour afficher la liste des types de conteneurs non hérités pris en charge, consultez the section called “Pourquoi certaines versions de plate-forme sont-elles marquées héritées ?”



f. Cliquez sur Suivant.

6. Si vous avez choisi de lancer votre environnement à l'intérieur d'un VPC, la page VPC Options (Options du VPC) s'affiche. Sinon, la page Additional Options (Options supplémentaires) s'affiche. Ici, vous allez configurer vos options de VPC.





a. Sélectionnez l'ID de VPC du VPC dans lequel vous souhaitez lancer votre environnement.

b. Pour un environnement à charge équilibrée et évolutif, sélectionnez privé pour ELB Scheme si vous ne souhaitez pas que votre Elastic Load Balancer soit disponible sur Internet.

   Pour un environnement instance unique, cette option n'est pas applicable, car l'environnement n'a pas d'équilibreur de charge. Pour de plus amples informations, veuillez consulter Types d'environnement.

c. Pour un environnement évolutif et équilibré en charge, sélectionnez les sous-réseaux pour l'équilibreur de charge élastique et les instances. EC2 Si vous avez créé des sous-réseaux publics et privés, assurez-vous que l'équilibreur de charge élastique et les EC2 instances sont associés au sous-réseau approprié. Par défaut, Amazon VPC crée un sous-réseau public par défaut à l'aide de 10.0.0.0/24 et d'un sous-réseau privé à l'aide de 10.0.1.0/24. Vous pouvez consulter vos sous-réseaux existants dans la console Amazon VPC à l'adresse. https://console.aws.amazon.com/vpc/

Pour un environnement instance unique, votre VPC a besoin uniquement d'un sous-réseau public de l'instance. Sélectionner un sous-réseau pour l'équilibreur de charge n'est pas applicable, car l'environnement n'a pas d'équilibreur de charge. Pour plus d'informations, consultez Types d'environnement.

d. Pour un environnement à charge équilibrée et évolutif, sélectionnez le groupe de sécurité que vous avez créé pour vos instances, le cas échéant.

   Pour un environnement instance unique, vous n'avez pas besoin d'un périphérique NAT. Sélectionnez le groupe de sécurité par défaut. Elastic Beanstalk attribue à l'instance une adresse IP Elastic qui lui permet d'accéder à Internet.

e. Cliquez sur Suivant.

7. Sur la page Application Options (Options de l'application), configurez les options de votre application.

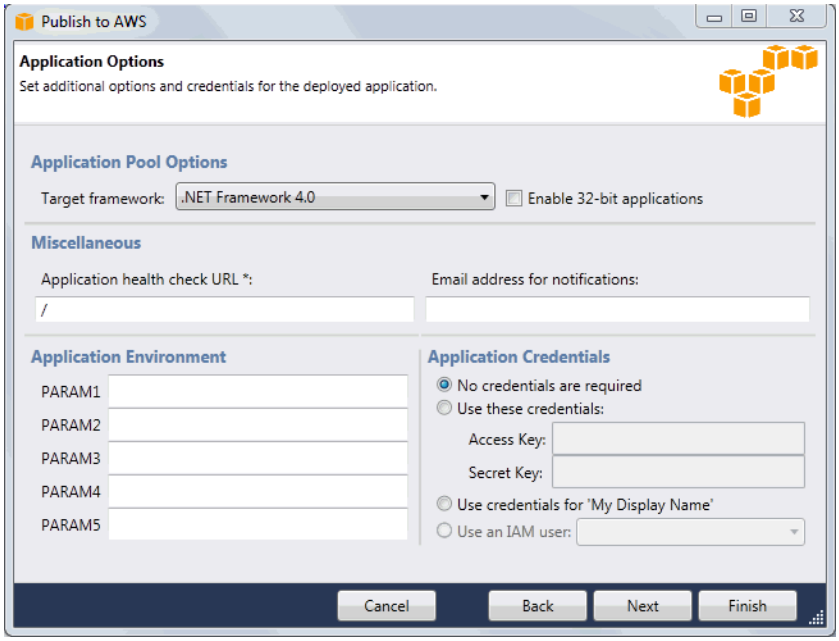
   a. Pour l'infrastructure cible, sélectionnez .NET Framework 4.0.

   b. Elastic Load Balancing utilise un test de santé pour déterminer si les EC2 instances Amazon exécutant votre application sont saines. La vérification de l'état détermine l'état d'intégrité d'une instance en détectant une URL spécifiée à un intervalle défini. Vous pouvez remplacer l'URL par défaut par une URL qui correspond à une ressource existante dans votre application (par exemple, `/myapp/index.aspx`) en entrant celle-ci dans la zone URL de vérification de l'état de l'application. Pour plus d'informations sur les vérifications de l'état de l'application, consultez Surveillance de l'état.

   c. Tapez une adresse e-mail si vous souhaitez recevoir des notifications Amazon Simple Notification Service (Amazon SNS) des événements importants qui affectent votre application.

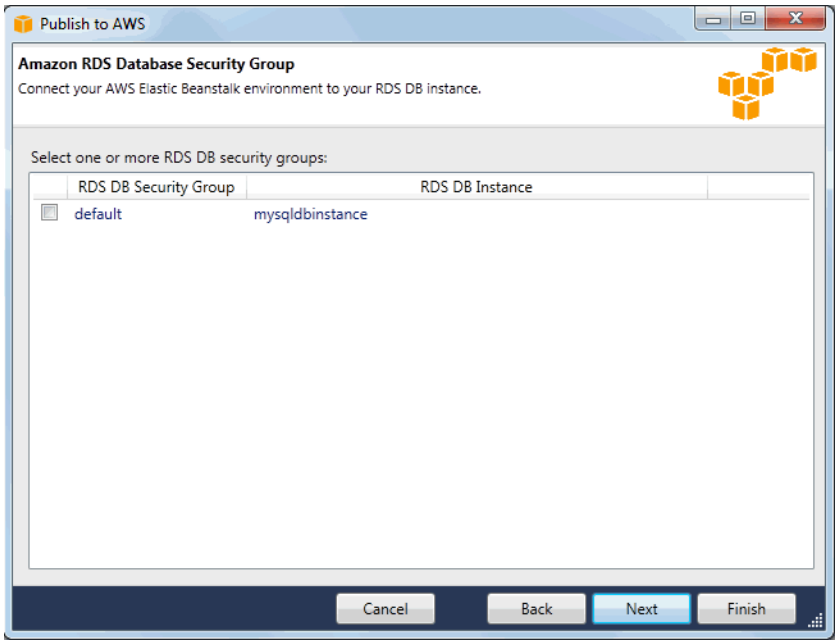
   d. La section Environnement de l'application vous permet de spécifier des variables d'environnement sur les EC2 instances Amazon qui exécutent votre application. Ce paramètre permet une plus grande portabilité en éliminant la nécessité de recompiler votre code source pendant que vous vous déplacez entre les environnements.

   e. Sélectionnez l'option d'informations d'identification des applications que vous voulez utiliser pour déployer votre application.

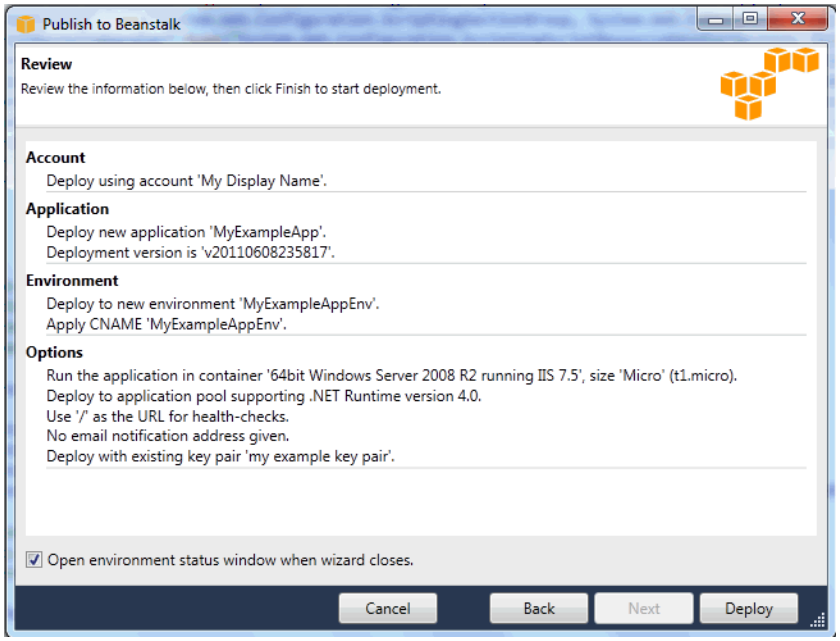


f. Cliquez sur Suivant.

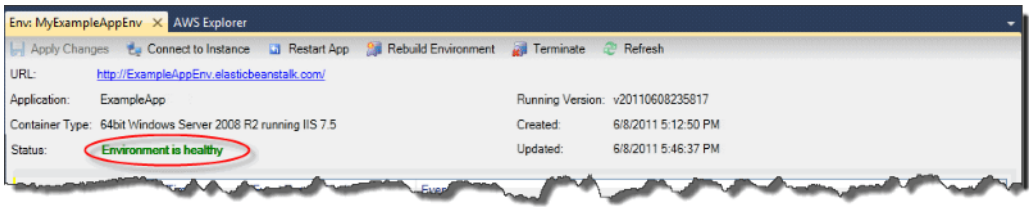
8. Si vous avez déjà mis en place une base de données Amazon RDS, la page Amazon RDS DB Security Group (Groupe de sécurité DB Amazon RDS) s'affiche. Si vous souhaitez connecter votre environnement Elastic Beanstalk à votre Instance DB Amazon RDS, sélectionnez alors un ou plusieurs groupes de sécurité. Sinon, allez à l'étape suivante. Lorsque vous êtes prêt, cliquez sur Next (Suivant).



9. Passez en revue vos options de déploiement. Si tout est tel que vous le souhaitez, cliquez sur Deploy (Déploiement).



Votre projet ASP.NET sera exporté comme un fichier de déploiement web, téléchargé sur Amazon S3 et inscrit en tant que nouvelle version de l'application avec Elastic Beanstalk. La fonctionnalité de déploiement Elastic Beanstalk surveille votre environnement jusqu'à ce qu'il devienne disponible avec le code nouvellement déployé. Sous l'onglet env:<nom de l'environnement>, vous verrez l'état de votre environnement.



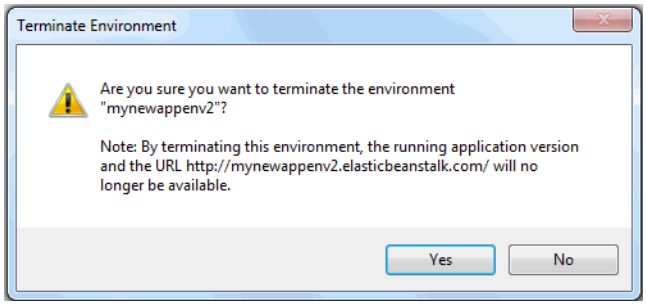
Résiliation d'un environnement

Pour éviter d'encourir des frais pour les AWS ressources non utilisées, vous pouvez mettre fin à un environnement en cours d'exécution à l'aide du AWS Toolkit for Visual Studio.

> **Note**
>
> Vous pouvez toujours lancer un nouvel environnement en utilisant la même version ultérieurement.

Pour résilier un environnement

1. Développez le nœud Elastic Beanstalk et le nœud d'application dans AWS Explorer (Explorateur AWS). Cliquez avec le bouton droit la souris sur votre environnement d'application et sélectionnez Terminate Environment (Résilier l'environnement).
2. Lorsque vous y êtes invité, cliquez sur Yes (Oui) afin de confirmer que vous souhaitez résilier l'environnement. Elastic Beanstalk mettra quelques minutes à arrêter les ressources en cours d'exécution dans l' AWS environnement.



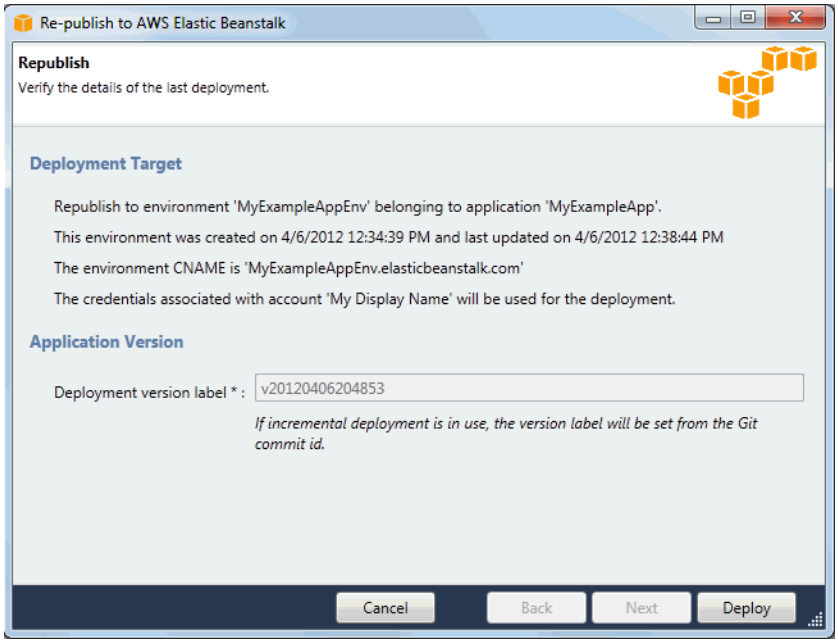
> **Note**
>
> Lorsque vous résiliez votre environnement, le CNAME associé à l'environnement résilié devient disponible pour que tout le monde puisse l'utiliser.

Déploiement dans votre environnement

Maintenant que vous avez testé votre application, il est facile de modifier et de redéployer de votre application et de consulter les résultats en quelques instants.

Pour modifier et redéployer votre application web ASP.NET

1. Dans l'Explorateur de solutions, cliquez avec le bouton droit sur votre application, puis cliquez sur Republier dans l'environnement < ***your environment name*** >. L'assistant Republication sur AWS Elastic Beanstalk s'ouvre.



2. Passez en revue les détails de votre déploiement et cliquez sur Deploy (Déploiement).

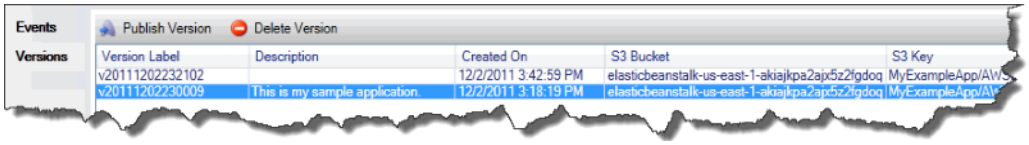
> **Note**
>
> Si vous souhaitez modifier certains de vos paramètres, vous pouvez cliquer sur Cancel (Annuler) et utiliser l'assistant Publish to AWS (Publier dans AWS) à la place. Pour obtenir des instructions, consultez Créer un environnement Elastic Beanstalk.

Votre projet web ASP.NET mis à jour sera exporté comme un fichier Web Deploy avec la nouvelle étiquette de version, téléchargé sur Amazon S3 et inscrit en tant que nouvelle version de l'application avec Elastic Beanstalk. La fonctionnalité de déploiement Elastic Beanstalk surveille votre environnement existant jusqu'à ce qu'il devienne disponible avec le code nouvellement déployé. Dans l'onglet env : < ***environment name*** >, vous verrez l'état de votre environnement.

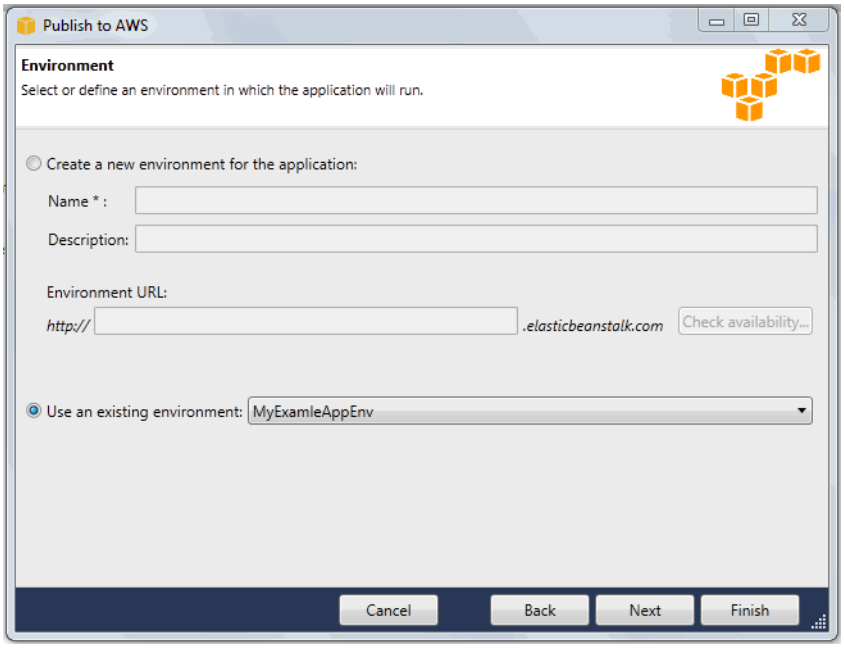
Vous pouvez également déployer une application existante dans un environnement existant si, par exemple, vous devez restaurer une version précédente de l'application.

Pour déployer une version de l'application dans un environnement existant

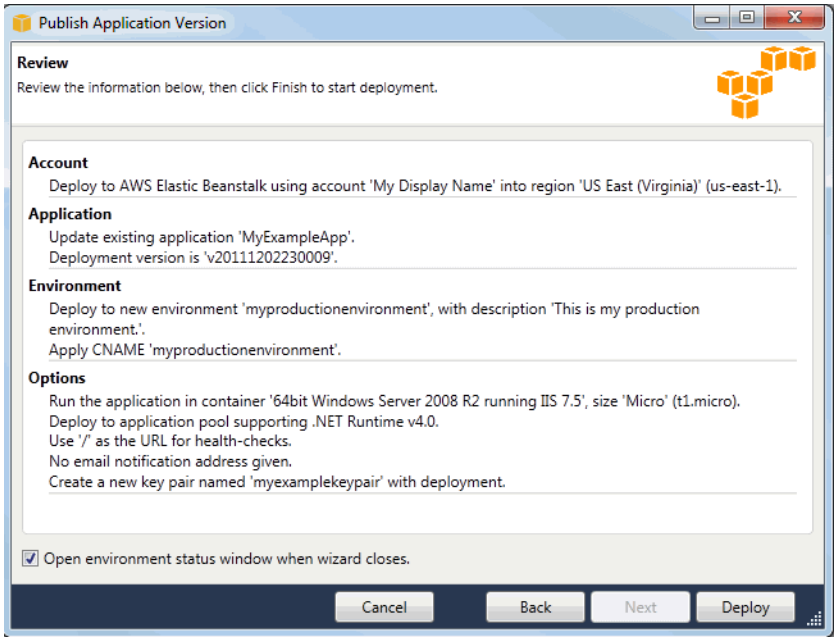
1. Cliquez avec le bouton droit de la souris sur votre application Elastic Beanstalk en développant le nœud Elastic Beanstalk dans AWS Explorer (Explorateur AWS). Sélectionnez View Status (Afficher l'état).
2. Dans l'onglet App : **< *application name* >**, cliquez sur Versions.



3. Cliquez sur la version de l'application que vous souhaitez déployer et cliquez sur Publish Version (Publier la version).
4. Dans l'assistant Publish Application Version (Publier la version de l'application), cliquez sur Next (Suivant).



5. Passez en revue les options de votre déploiement et cliquez sur Deploy (Déploiement).



Votre projet ASP.NET sera exporté comme un fichier Web Deploy et chargé sur Amazon S3. La fonctionnalité de déploiement Elastic Beanstalk surveille votre environnement jusqu'à ce qu'il devienne disponible avec le code nouvellement déployé. Dans l'onglet env : < ***environment name*** >, vous verrez l'état de votre environnement.

Gestion de vos environnements d'application Elastic Beanstalk

Avec le AWS Toolkit for Visual Studio et la console de AWS gestion, vous pouvez modifier le provisionnement et la configuration des AWS ressources utilisées par vos environnements d'applications. Pour plus d'informations sur la gestion de vos environnements d'applications à l'aide de la console AWS de gestion, consultezGestion des environnements Elastic Beanstalk. Cette section décrit les paramètres de service spécifiques que vous pouvez modifier dans AWS Toolkit for Visual Studio dans le cadre de la configuration d'environnement de votre application.

Modification des paramètres de configuration de l'environnement

Lorsque vous déployez votre application, Elastic Beanstalk configure un certain nombre de services de cloud computing. AWS Vous pouvez contrôler la façon dont ces services individuels sont configurés à l'aide du AWS Toolkit for Visual Studio.

Pour modifier les paramètres d'environnement d'une application

- Développez le nœud Elastic Beanstalk et le nœud de votre application. Ensuite, cliquez avec le bouton droit de la souris sur votre environnement Elastic Beanstalk dans AWS Explorer (Explorateur AWS). Sélectionnez View Status (Afficher l'état).

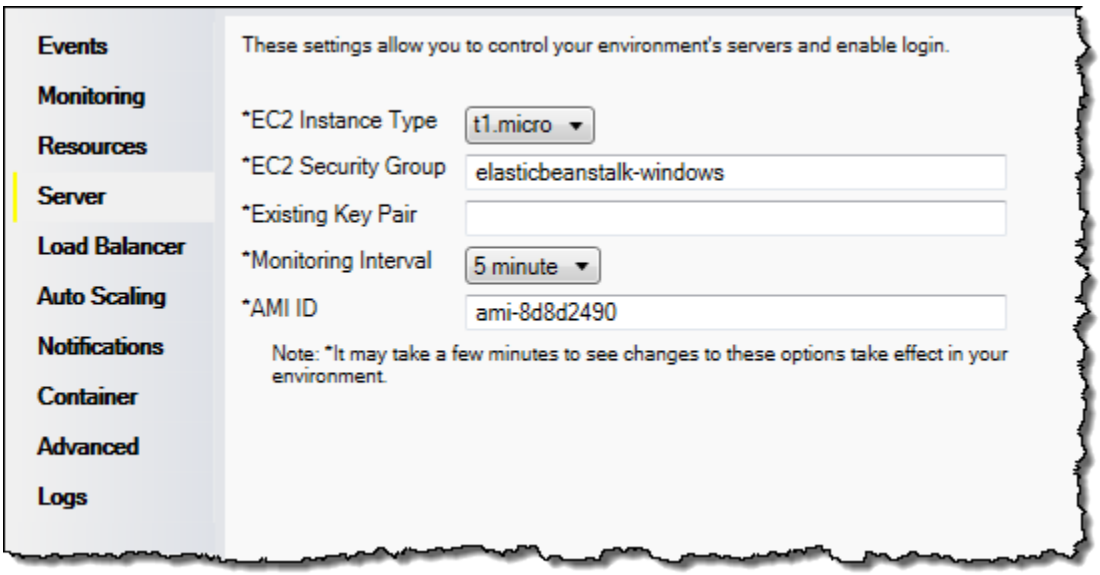
  Vous pouvez à présent configurer des paramètres pour les éléments suivants :

  - de bases de données
  - Equilibrage de charge
  - Auto Scaling
  - Notifications
  - Propriétés de l'environnement

Configuration des instances de EC2 serveur à l'aide du AWS kit d'outils pour Visual Studio

Amazon Elastic Compute Cloud (Amazon EC2) est un service Web que vous utilisez pour lancer et gérer des instances de serveur dans les centres de données d'Amazon. Vous pouvez utiliser les instances de EC2 serveur Amazon à tout moment, aussi longtemps que vous le souhaitez et à toutes fins légales. Les instances sont disponibles dans différentes tailles et configurations. Pour plus d'informations, rendez-vous sur Amazon EC2.

Vous pouvez modifier la configuration de l'instance Amazon de l'environnement Elastic Beanstalk à l'aide EC2 de l'onglet Server situé dans l'onglet de votre environnement d'application dans le Toolkit for Visual AWS Studio.



Types d' EC2 instances Amazon

Instance type (Type d'instance) affiche les types d'instance disponibles pour votre application Elastic Beanstalk. Changez le type d'instance pour sélectionner un serveur dont les caractéristiques (y compris la taille de la mémoire et la puissance d'UC) sont les mieux adaptées à votre application. Par exemple, les applications exécutant des opérations intensives et de longue durée peuvent nécessiter plus de puissance de calcul et de mémoire.

Pour plus d'informations sur les types d' EC2 instances Amazon disponibles pour votre application Elastic Beanstalk, consultez la section Types d'instances dans le guide de l'utilisateur Amazon Elastic Compute Cloud.

Groupes EC2 de sécurité Amazon

Vous pouvez contrôler l'accès à votre application Elastic Beanstalk à l'aide EC2 d'un groupe de sécurité Amazon. Un groupe de sécurité définit les règles de pare-feu de vos instances. Ces règles déterminent le trafic réseau d'entrée (c'est à dire, entrant) doit être acheminé vers votre instance. Tout autre trafic d'entrée sera ignoré. Vous pouvez modifier les règles pour un groupe à la fois. Les nouvelles règles sont appliquées automatiquement pour toutes les instances en cours d'exécution et les instances lancées par la suite.

Vous pouvez configurer vos groupes de EC2 sécurité Amazon à l'aide de la console de AWS gestion ou du AWS Toolkit for Visual Studio. Vous pouvez spécifier quels groupes de EC2 sécurité Amazon contrôlent l'accès à votre application Elastic Beanstalk en saisissant les noms d'un ou de plusieurs groupes de sécurité EC2 Amazon (séparés par des virgules) dans la zone de texte Security Groups. EC2

> **Note**
>
> Assurez-vous que le port 80 (HTTP) est accessible à partir de 0.0.0.0/0 comme source de la plage CIDR si vous souhaitez activer les vérifications de l'état pour votre application. Pour plus d'informations sur les vérifications de l'état, consultez Vérifications de l'état.

Pour créer un groupe de sécurité à l'aide du AWS kit d'outils pour Visual Studio

1. Dans Visual Studio, dans AWS Explorer, développez le EC2 nœud Amazon, puis double-cliquez sur Security Groups.
2. Cliquez sur Create Security Group (Créer un groupe de sécurité) et entrez un nom et une description pour votre groupe de sécurité.
3. Cliquez sur OK.

Pour plus d'informations sur les groupes EC2 de sécurité Amazon, consultez la section Utilisation des groupes de sécurité dans le guide de l'utilisateur d'Amazon Elastic Compute Cloud.

Paires EC2 de clés Amazon

Vous pouvez vous connecter en toute sécurité aux EC2 instances Amazon mises en service pour votre application Elastic Beanstalk à l'aide d'une paire de clés Amazon. EC2

> **Important**
>
> Vous devez créer une paire de EC2 clés Amazon et configurer vos instances Amazon fournies par Elastic Beanstalk pour utiliser la EC2 paire de clés Amazon avant de pouvoir accéder à vos EC2 instances Amazon provisionnées par Elastic Beanstalk. EC2 Vous pouvez créer votre paire de clés à l'aide de l' AWS assistant Publish to du AWS Toolkit for Visual Studio lorsque vous déployez votre application sur Elastic Beanstalk. Si vous souhaitez créer des paires de clés supplémentaires à l'aide de Toolkit, procédez comme suit. Vous pouvez également configurer vos paires de EC2 clés Amazon à l'aide de la console AWS de gestion. Pour obtenir des instructions sur la création d'une paire de clés pour Amazon EC2, consultez le guide de démarrage Amazon Elastic Compute Cloud.

La zone de texte Existing Key Pair vous permet de spécifier le nom d'une paire de EC2 clés Amazon que vous pouvez utiliser pour vous connecter en toute sécurité aux EC2 instances Amazon exécutant votre application Elastic Beanstalk.

Pour spécifier le nom d'une paire de EC2 clés Amazon

1. Développez le EC2 nœud Amazon et double-cliquez sur Key Pairs.
2. Cliquez sur Create Key Pair (Créer une paire de clés) et saisissez le nom de la paire de clés.
3. Cliquez sur OK.

Pour plus d'informations sur les paires de EC2 clés Amazon, consultez la section Utilisation des EC2 informations d'identification Amazon dans le guide de l'utilisateur d'Amazon Elastic Compute Cloud. Pour plus d'informations sur la connexion aux EC2 instances Amazon, consultezAffichage de la liste des instances de serveur et connexion à ces instances.

Intervalle de surveillance

Par défaut, seules les CloudWatch métriques Amazon de base sont activées. Elles renvoient des données toutes les cinq minutes. Vous pouvez activer des CloudWatch mesures plus détaillées d'une minute en sélectionnant 1 minute pour l'intervalle de surveillance dans la section Serveur de l'onglet Configuration de votre environnement dans le. AWS Toolkit for Eclipse

> **Note**
>
> Des frais CloudWatch de service Amazon peuvent s'appliquer pour les mesures d'intervalle d'une minute. Consultez Amazon CloudWatch pour plus d'informations.

ID d'AMI personnalisé

Vous pouvez remplacer l'AMI par défaut utilisée pour vos EC2 instances Amazon par votre propre AMI personnalisée en saisissant l'identifiant de votre AMI personnalisée dans le champ ID d'AMI personnalisé de la section Serveur de l'onglet Configuration de votre environnement dans le AWS Toolkit for Eclipse.

> **Important**
>
> L'utilisation de votre propre image AMI est une tâche avancée qui doit être effectuée avec soin. Si vous avez besoin d'une AMI personnalisée, nous vous recommandons de démarrer

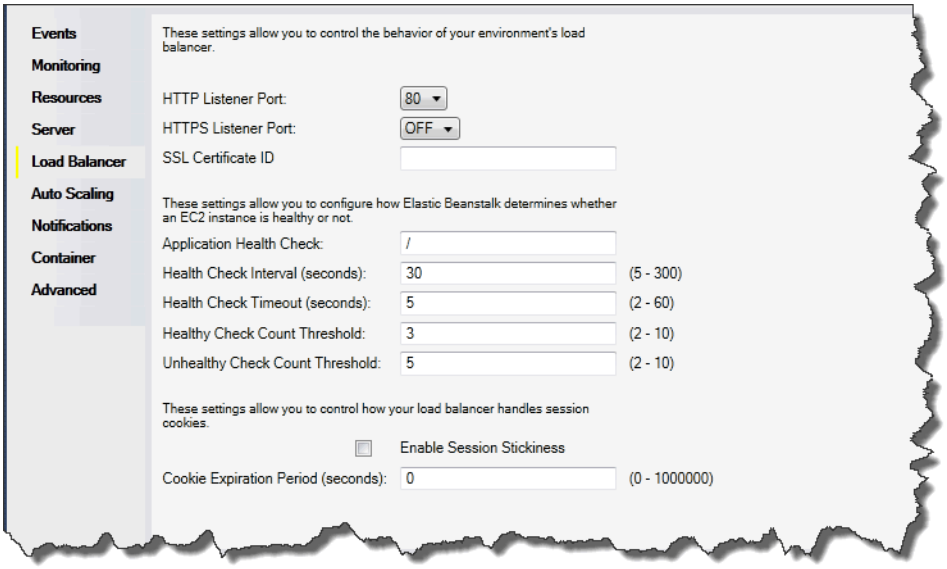
par l'AMI Elastic Beanstalk par défaut, puis de la modifier. Pour être considérées comme saines, Elastic Beanstalk EC2 attend des instances Amazon qu'elles répondent à un ensemble d'exigences, notamment qu'elles soient dotées d'un gestionnaire d'hôte actif. Si ces conditions ne sont pas satisfaites, il se peut que votre environnement ne fonctionne pas correctement.

Configuration d'Elastic Load Balancing à l'aide du AWS kit d'outils pour Visual Studio

Elastic Load Balancing est un service d'Amazon Web Services qui vous aide à améliorer la disponibilité et l'évolutivité de votre application. Ce service vous permet de répartir facilement les charges d'applications entre deux ou plusieurs EC2 instances Amazon. Elastic Load Balancing active la disponibilité par la redondance et prend en charge l'augmentation du trafic pour votre application.

Elastic Load Balancing permet de répartir et d'équilibrer automatiquement le trafic entrant de votre application sur toutes les instances que vous exécutez. Il vous permet également d'ajouter aisément de nouvelles instances lorsque vous avez besoin d'augmenter la capacité de votre application.

Elastic Beanstalk fournit automatiquement Elastic Load Balancing lorsque vous déployez une application. Vous pouvez modifier la configuration de l'instance Amazon de l'environnement Elastic Beanstalk à l'aide EC2 de l'onglet Load Balancer situé dans l'onglet environnement de votre application dans Toolkit AWS for Visual Studio.



Les sections suivantes décrivent les paramètres Elastic Load Balancing que vous pouvez configurer pour votre application.

Ports

L'équilibreur de charge configuré pour traiter les demandes relatives à votre application Elastic Beanstalk envoie des demandes aux instances EC2 Amazon qui exécutent votre application. L'équilibreur de charge provisionné peut écouter les demandes sur les ports HTTP et HTTPS et acheminer les demandes vers les EC2 instances Amazon de votre AWS Elastic Beanstalk application. Par défaut, l'équilibreur de charge gère les demandes sur le port HTTP. Au moins un des ports (HTTP ou HTTPS) doit être activé.

> **Important**
>
> Assurez-vous que le port spécifié n'est pas verrouillé. S'il l'est, les utilisateurs ne pourront pas se connecter à votre application Elastic Beanstalk.

Contrôle du port HTTP

Pour désactiver le port HTTP, sélectionnez OFF (désactivé) pour HTTP Listener Port (Port d'écoute HTTP). Pour activer le port HTTP, vous sélectionnez un port HTTP (par exemple, 80) dans la liste.

> **Note**
>
> Pour accéder à votre environnement à l'aide d'un port autre que le port 80, par exemple le port 8080, vous pouvez ajouter un écouteur à l'équilibreur de charge existant et configurer le nouvel écouteur de sorte qu'il écoute sur ce port.
> Par exemple, en utilisant les équilibreurs de charge AWS CLI pour Classic, tapez la commande suivante, en la *LOAD_BALANCER_NAME* remplaçant par le nom de votre équilibreur de charge pour Elastic Beanstalk.
>
> ```
> aws elb create-load-balancer-listeners --load-balancer-name LOAD_BALANCER_NAME
>  --listeners "Protocol=HTTP, LoadBalancerPort=8080, InstanceProtocol=HTTP,
>  InstancePort=80"
> ```

Par exemple, à l'aide de AWS CLI for Application Load Balancers, tapez la commande suivante, en la *LOAD_BALANCER_ARN* remplaçant par l'ARN de votre équilibreur de charge pour Elastic Beanstalk.

```
aws elbv2 create-listener --load-balancer-arn LOAD_BALANCER_ARN --protocol HTTP
 --port 8080
```

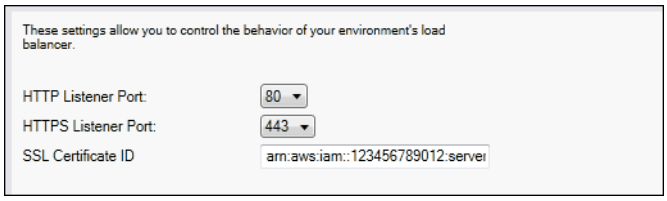
Si vous souhaitez que Elastic Beanstalk surveille votre environnement, ne supprimez pas l'écouteur sur le port 80.

Contrôle du port HTTPS

Elastic Load Balancing prend en charge le HTTPS/TLS protocole permettant de chiffrer le trafic pour les connexions des clients à l'équilibreur de charge. Les connexions entre l'équilibreur de charge et les EC2 instances utilisent le chiffrement en texte brut. Par défaut, le port HTTPS est désactivé.

Pour activer le port HTTPS

1. Créez un nouveau certificat à l'aide de AWS Certificate Manager (ACM) ou téléchargez un certificat et une clé vers Gestion des identités et des accès AWS (IAM). Pour plus d'informations sur une demande de certificat ACM, consultez Request a Certificate (Demande de certificat) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur AWS Certificate Manager). Pour plus d'informations sur l'importation de certificats tiers dans ACM, consultez Importing Certificates (Importation de certificats) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur AWS Certificate Manager). Si ACM n'est pas disponible dans votre région, utilisez Gestion des identités et des accès AWS (IAM) pour télécharger un certificat tiers. Les services ACM et IAM stockeront le certificat et fourniront un Amazon Resource Name (ARN) pour le certificat SSL. Pour de plus amples informations sur la création et le chargement des certificats dans IAM, veuillez consulter Utilisation des certificats de serveur dans le Guide de l'utilisateur IAM.
2. Spécifiez le port HTTPS en sélectionnant un port pour HTTPS Listener Port (Port d'écoute HTTPS).



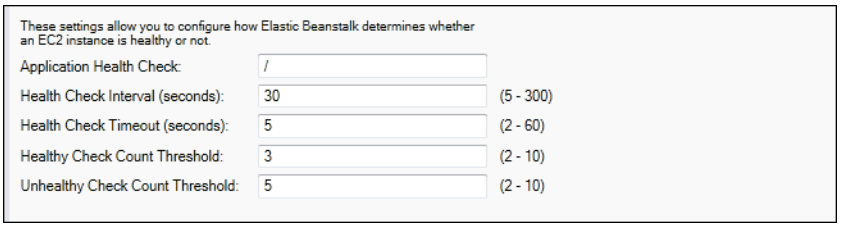
3. Pour SSL Certificate ID (ID du certificat SSL), saisissez l'ARN (Amazon Resources Name) de votre certificat SSL. Par exemple, **`arn:aws:iam::123456789012:server-certificate/abc/certs/build`** ou **`arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678`**. Utilisez le certificat SSL que vous avez créé ou chargé à l'étape 1.

Pour désactiver le port HTTPS, sélectionnez OFF (désactivé) pour HTTPS Listener Port (Port d'écoute HTTPS).

Vérifications de l'état

La définition de la vérification de l'état inclut une URL à interroger pour l'intégrité de l'instance. Par défaut, Elastic Beanstalk utilise TCP:80 pour les conteneurs non hérités et HTTP:80 pour les conteneurs hérités. Vous pouvez remplacer l'URL par défaut par une URL qui correspond à une ressource existante dans votre application (par exemple, `/myapp/default.aspx`) en entrant celle-ci dans la zone URL de vérification de l'état de l'application. Si vous remplacez l'URL par défaut, Elastic Beanstalk utilise HTTP pour interroger la ressource. Pour vérifier si vous utilisez un type de conteneur hérité, consultez the section called “Pourquoi certaines versions de plate-forme sont-elles marquées héritées ?”

Vous pouvez contrôler les paramètres du bilan de santé à l'aide de la section EC2 Instance Health Check du panneau d'équilibrage de charge.



La définition de la vérification de l'état inclut une URL à interroger pour l'intégrité de l'instance. Remplacez l'URL par défaut par une URL qui correspond à une ressource existante dans votre application (par exemple, `/myapp/index.jsp`) en entrant celle-ci dans la zone URL de vérification de l'état de l'application.

La liste suivante décrit les paramètres de vérification de l'état que vous pouvez définir pour votre application.

- Pour Health Check Interval (secondes), entrez le nombre de secondes pendant lesquelles Elastic Load Balancing attend entre les tests de santé pour les EC2 instances Amazon de votre application.

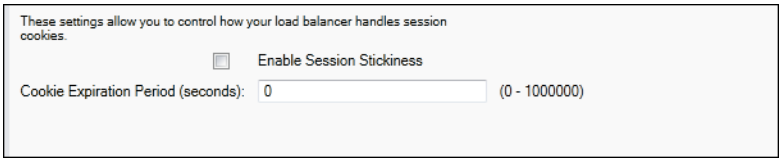
- Pour Délai de vérification de l'état (secondes), spécifiez le nombre de secondes d'attente d'une réponse pour Elastic Load Balancing avant de considérer que l'instance ne répond pas.
- Pour Seuil du nombre de vérifications de l'état saines et Seuil du nombre de vérifications de l'état non saines, spécifiez le nombre d'analyses d'URL consécutives réussies et non réussies avant qu'Elastic Load Balancing ne modifie l'état de l'instance. Par exemple, si vous spécifiez **5** pour Seuil du nombre de vérifications de l'état non saines, l'URL doit renvoyer un message d'erreur ou une expiration du délai cinq fois de suite avant qu'Elastic Load Balancing considère que la vérification de l'état est un échec.

Séances

Par défaut, un équilibreur de charge achemine chaque demande de façon indépendante à l'instance de serveur ayant la plus petite charge. Par comparaison, une session permanente lie la séance d'un utilisateur à une instance de serveur spécifique afin que toutes les demandes provenant de l'utilisateur pendant la séance soient envoyées à la même instance de serveur.

Elastic Beanstalk utilise des cookies HTTP générés par l'équilibreur de charge lorsque des sessions permanentes sont activées pour une application. L'équilibreur de charge utilise un cookie spécial généré par l'équilibreur de charge pour suivre l'instance d'application pour chaque demande. Lorsque l'équilibreur de charge reçoit une demande, il vérifie d'abord si ce cookie est présent dans la demande. Si tel est le cas, la demande est envoyée à l'instance d'application spécifiée dans le cookie. S'il n'y a pas de cookie, l'équilibreur de charge choisit une instance d'application à partir de l'algorithme d'équilibrage de charge existant. Un cookie est inséré dans la réponse pour lier les demandes suivantes provenant du même utilisateur à cette instance d'application. La configuration de la stratégie définit l'expiration d'un cookie, ce qui établit la durée de validité de chaque cookie.

Vous pouvez utiliser la section Sessions dans l'onglet Équilibreur de charge afin d'indiquer si l'équilibreur de charge pour votre application autorise ou non la permanence de session.



Pour de plus amples informations Elastic Load Balancing, veuillez consulter le Guide du développeur Elastic Load Balancing.

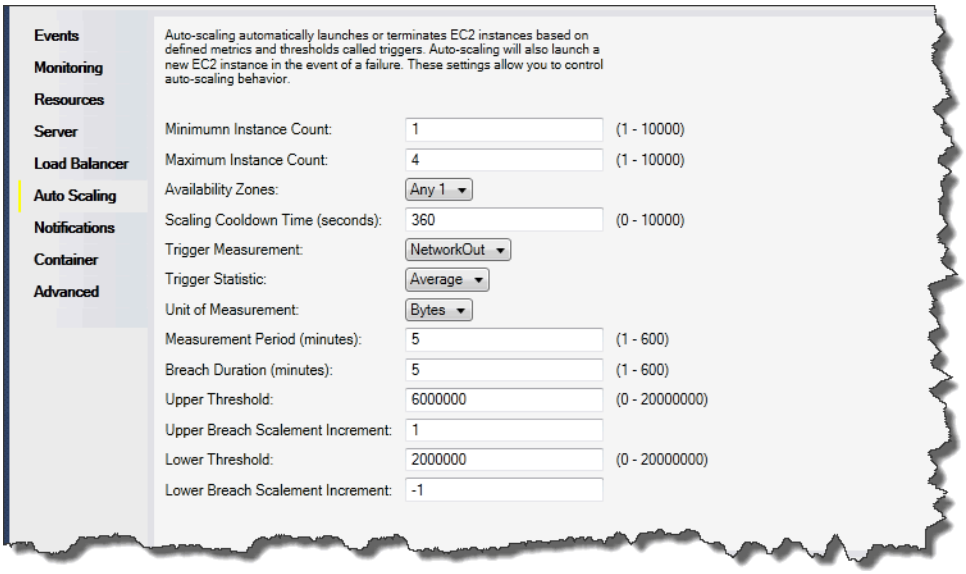
Configuration d'Auto Scaling à l'aide du AWS kit d'outils pour Visual Studio

Amazon EC2 Auto Scaling est un service Web Amazon conçu pour lancer ou mettre fin automatiquement aux EC2 instances Amazon en fonction de déclencheurs définis par l'utilisateur. Les utilisateurs peuvent configurer des groupes Auto Scaling et y associer des déclencheurs afin de mettre à l'échelle automatiquement les ressources de calcul selon des métriques comme l'utilisation de la bande passante ou l'utilisation de l'UC. Amazon EC2 Auto Scaling travaille avec Amazon CloudWatch pour récupérer les métriques des instances de serveur exécutant votre application.

Amazon EC2 Auto Scaling vous permet de prendre un groupe d' EC2 instances Amazon et de définir différents paramètres pour que le nombre de ce groupe augmente ou diminue automatiquement. Amazon EC2 Auto Scaling peut ajouter ou supprimer EC2 des instances Amazon de ce groupe afin de vous aider à gérer facilement les modifications du trafic vers votre application.

Amazon EC2 Auto Scaling surveille également l'état de santé de chaque EC2 instance Amazon lancée. Si une instance se termine de façon inattendue, Amazon EC2 Auto Scaling détecte la résiliation et lance une instance de remplacement. Cette fonctionnalité vous permet de maintenir automatiquement un nombre fixe et souhaité d' EC2 instances Amazon.

Elastic Beanstalk fournit EC2 Amazon Auto Scaling pour votre application. Vous pouvez modifier la configuration de l'instance Amazon de l'environnement Elastic Beanstalk à l'aide EC2 de l'onglet Auto Scaling situé dans l' AWS onglet Environnement de votre application dans le Toolkit for Visual Studio.

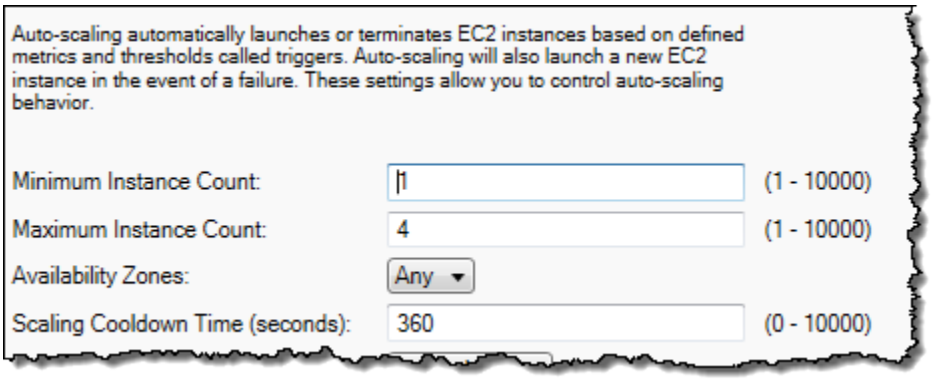


La section suivante explique comment configurer les paramètres Auto Scaling pour votre application.

Lancement de la configuration

Vous pouvez modifier la configuration de lancement pour contrôler la manière dont votre application Elastic Beanstalk approvisionne les ressources EC2 Amazon Auto Scaling.

Les zones Minimum Instance Count (Nombre minimum d'instances) et Maximum Instance Count (Nombre maximum d'instances) vous permettent de spécifier les tailles maximale et minimale du groupe Auto Scaling utilisé par votre application Elastic Beanstalk.



> **Note**
>
> Pour conserver un nombre fixe d' EC2 instances Amazon, définissez le nombre minimum d'instances et le nombre maximal d'instances sur la même valeur.

La case Zones de disponibilité vous permet de spécifier le nombre de zones de disponibilité dans lesquelles vous souhaitez que vos EC2 instances Amazon se trouvent. Il est important de définir ce nombre si vous souhaitez créer des applications à tolérance de panne. Si une zone de disponibilité est défaillante, l'exécution de vos instances se poursuivra dans vos autres zones de disponibilité.

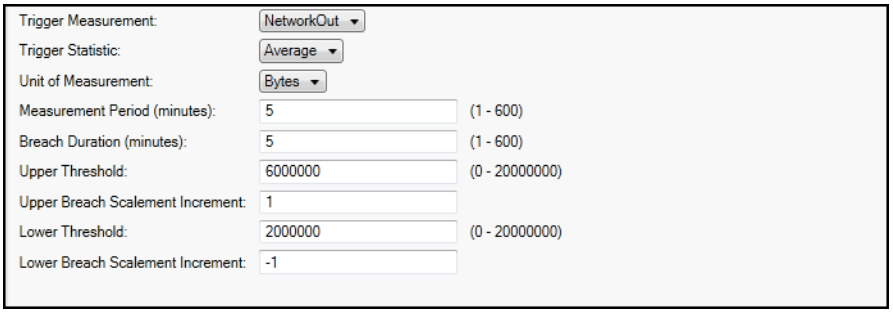
> **Note**
>
> Il est actuellement impossible de spécifier la zone de disponibilité dans laquelle se situera votre instance.

Triggers

Un déclencheur est un mécanisme Amazon EC2 Auto Scaling que vous configurez pour indiquer au système quand vous souhaitez augmenter (augmenter) le nombre d'instances et quand vous souhaitez le diminuer (augmenter) le nombre d'instances. Vous pouvez configurer des déclencheurs

pour qu'ils se déclenchent sur n'importe quelle métrique publiée sur Amazon CloudWatch, telle que l'utilisation du processeur, et déterminer si les conditions que vous avez spécifiées sont remplies. Lorsque les seuils inférieurs ou supérieurs des conditions que vous avez spécifiées pour la métrique ont été dépassés pendant la période spécifiée, le déclencheur lance un processus de longue durée que nous appelons une activité de dimensionnement.

Vous pouvez définir un déclencheur de dimensionnement pour votre application Elastic Beanstalk à l'aide de Toolkit for AWS Visual Studio.



Les déclencheurs Amazon EC2 Auto Scaling fonctionnent en surveillant une CloudWatch métrique Amazon spécifique pour une instance. Les déclencheurs incluent l'utilisation de l'UC, le trafic réseau et l'activité du disque. Utilisez le paramètre Mesure du déclencheur pour sélectionner une métrique associée à votre déclencheur.

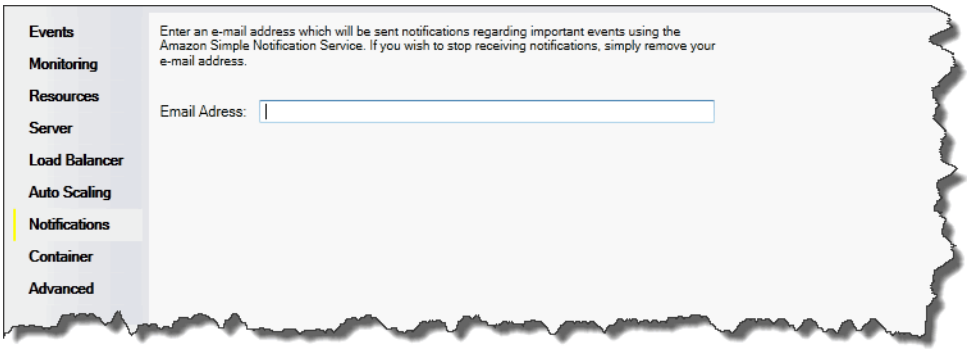
La liste suivante décrit les paramètres de déclenchement que vous pouvez configurer à l'aide de la console AWS de gestion.

- Vous pouvez spécifier les statistiques que le déclencheur devrait utiliser. Vous pouvez sélectionner Minimum, Maximum, Sum (Somme) ou Average (Moyenne) pour Statistique du déclencheur.
- Pour Unité de mesure, spécifiez l'unité de mesure du déclencheur.
- La valeur figurant dans le champ Période de mesure indique à quelle fréquence Amazon CloudWatch mesure les statistiques pour votre déclencheur. La valeur Durée de la faille correspond à la durée pendant laquelle une métrique peut se situer au-delà de sa limite définie (telle que spécifiée dans Seuil supérieur et Seuil inférieur) avant l'activation du déclencheur.
- Pour l'incrément d'échelle de violation supérieur et l'incrément d'échelle de violation inférieur, spécifiez le nombre d' EC2instances Amazon à ajouter ou à supprimer lors de l'exécution d'une activité de dimensionnement.

Pour plus d'informations sur Amazon EC2 Auto Scaling, consultez la section Amazon EC2 Auto Scaling de la documentation Amazon Elastic Compute Cloud.

Configuration des notifications à l'aide du AWS kit d'outils pour Visual Studio

Elastic Beanstalk utilise Amazon Simple Notification Service (Amazon SNS) pour vous informer des événements importants qui concernent votre application. Pour activer les notifications Amazon SNS, il suffit d'entrer votre adresse e-mail dans la zone Adresse e-mail. Pour désactiver ces notifications, supprimez votre adresse e-mail de la zone.



Configuration de conteneurs .NET à l'aide du AWS kit d'outils pour Visual Studio

Le panneau Container/.NET Options vous permet d'affiner le comportement de vos EC2 instances Amazon et d'activer ou de désactiver la rotation des journaux Amazon S3. Vous pouvez utiliser le AWS Toolkit for Visual Studio pour configurer les informations de votre conteneur.

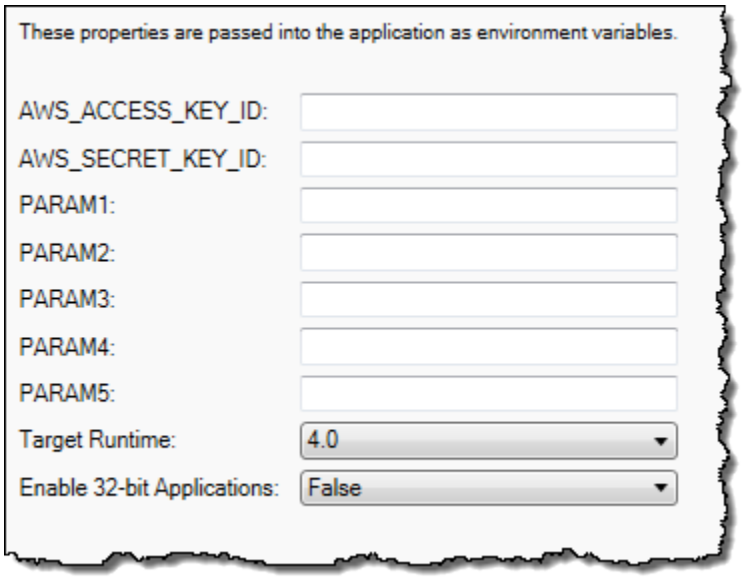
> **Note**
>
> Pour modifier vos paramètres de configuration sans aucune interruption, échangez le CNAME pour vos environnements. Pour plus d'informations, consultez Déploiements bleu/vert avec Elastic Beanstalk.

Si vous le souhaitez, vous pouvez étendre le nombre de paramètres. Pour plus d'informations sur l'extension des paramètres, consultez Paramètres d'option.

Pour accéder au panneau d'options Container/.NET de votre application Elastic Beanstalk

1. Dans AWS Toolkit for Visual Studio, développez le nœud Elastic Beanstalk et le nœud de votre application.
2. Dans AWS Explorer (Explorateur AWS), double-cliquez sur votre environnement Elastic Beanstalk.
3. En bas du volet Présentation, cliquez sur l'onglet Configuration.

4. Sous Container (Conteneur), vous pouvez configurer les options de conteneur.

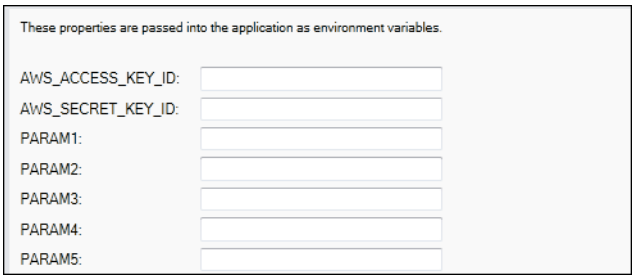


Options du conteneur .NET

Vous pouvez choisir la version de .NET Framework pour votre application. Choisissez 2.0 ou 4.0 pour Runtime cible. Sélectionnez Activer les applications 32 bits si vous souhaitez activer des applications 32 bits.

Paramètres de l'application

La section Paramètres de l'application vous permet de spécifier des variables d'environnement que vous pouvez lire depuis votre code d'application.



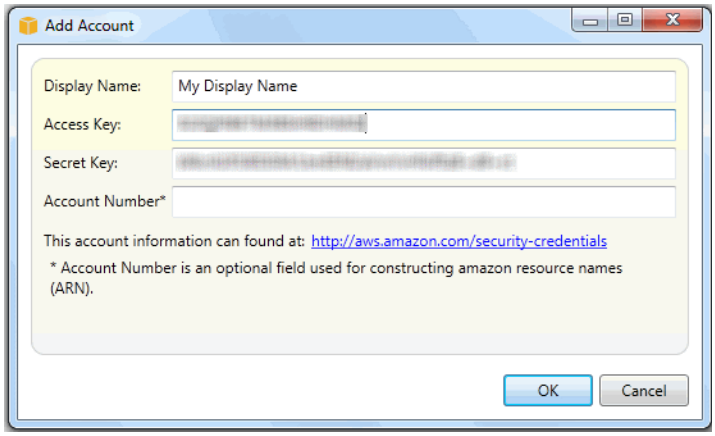
Gestion des comptes

Si vous souhaitez configurer différents AWS comptes pour effectuer différentes tâches, telles que les tests, la mise en scène et la production, vous pouvez ajouter, modifier et supprimer des comptes à l'aide du AWS Toolkit for Visual Studio.

Pour gérer plusieurs comptes

1. Dans Visual Studio, dans le menu Affichage, cliquez sur AWS Explorateur.
2. À côté de la liste Account (Compte), cliquez sur le bouton Add Account (Ajouter un compte).

La boîte de dialogue Add Account (Ajouter un compte) apparaît.



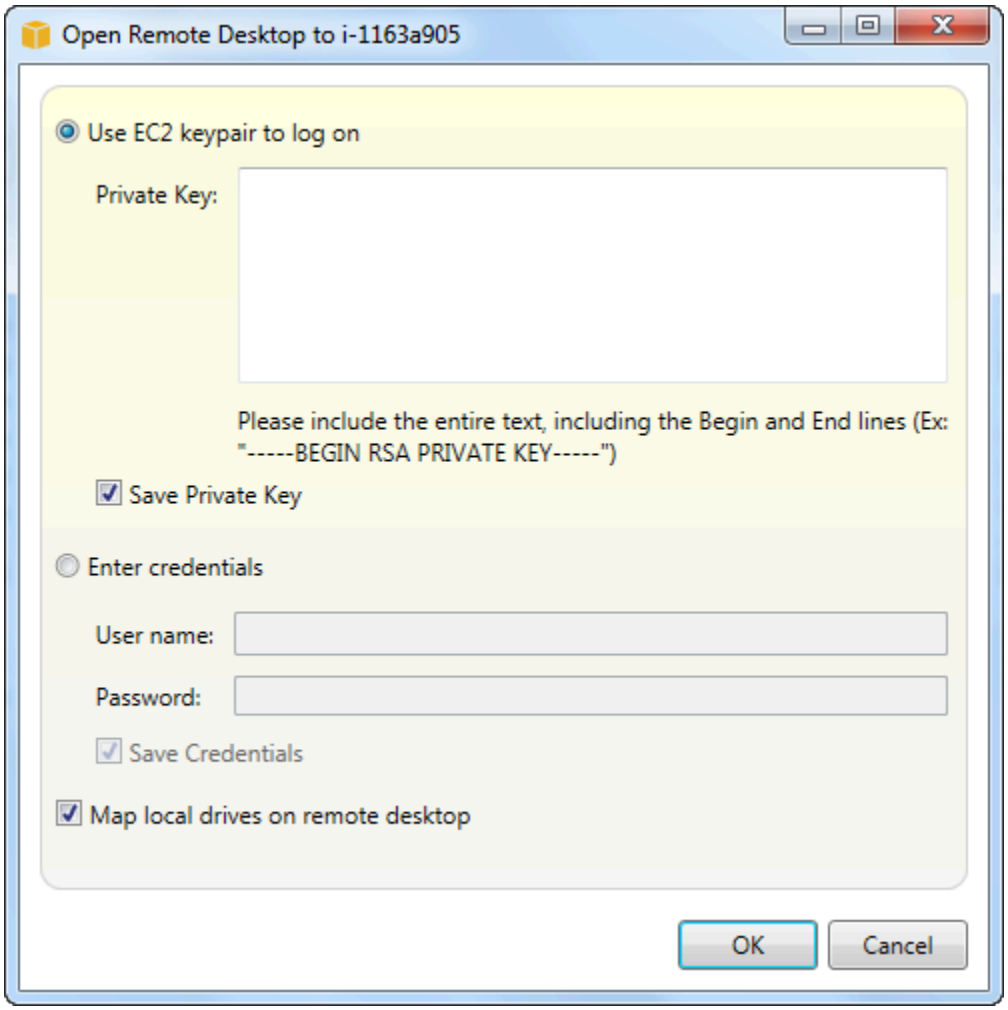
3. Renseignez les informations requises.
4. Les informations de votre compte apparaissent désormais dans l'onglet AWS Explorateur. Lorsque vous publiez sur Elastic Beanstalk, vous pouvez sélectionner le compte que vous souhaitez utiliser.

Affichage de la liste des instances de serveur et connexion à ces instances

Vous pouvez consulter la liste des EC2 instances Amazon exécutant votre environnement applicatif Elastic Beanstalk via le Toolkit for AWS Visual Studio ou depuis la console de gestion. AWS Vous pouvez vous connecter à ces instances à l'aide d'une connexion Bureau à distance. Pour plus d'informations sur le référencement et la connexion à vos instances de serveur à l'aide de la console AWS de gestion, consultezAffichage de la liste des instances de serveur et connexion à ces instances. La section suivante vous explique comment visualiser et vous connecter à vos instances de serveur à l'aide du AWS Toolkit for Visual Studio.

Pour consulter les EC2 instances Amazon d'un environnement et s'y connecter

1. Dans Visual Studio, dans AWS Explorer, développez le EC2 nœud Amazon et double-cliquez sur Instances.
2. Cliquez avec le bouton droit sur l'ID d' EC2 instance de l'instance Amazon exécutée dans l'équilibreur de charge de votre application dans la colonne Instance et sélectionnez Open Remote Desktop dans le menu contextuel.



3. Sélectionnez Utiliser une EC2 paire de clés pour vous connecter et collez le contenu du fichier de clé privée que vous avez utilisé pour déployer votre application dans le champ Clé privée. Sinon, entrez vos nom utilisateur et mot de passe dans les zones de texte User name (Nom d'utilisateur) and Password (Mot de passe).

> **Note**
>
> Si la paire de clés est stockée à l'intérieur du Toolkit, la zone de texte ne s'affiche pas.

4. Cliquez sur OK.

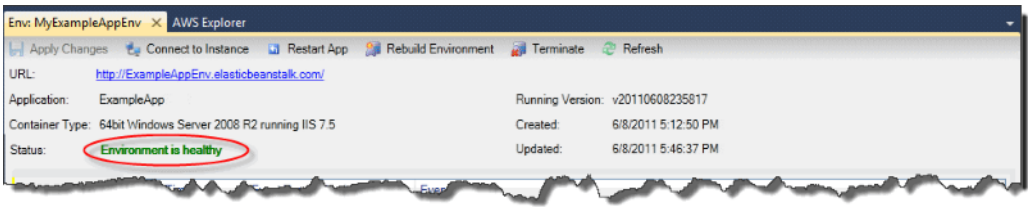
Surveillance de l'intégrité d'une application

Lorsque vous exécutez un site web de production, il est important de savoir que votre application est disponible et répond aux demandes. Pour aider à la surveillance de la réactivité de votre application, Elastic Beanstalk offre des fonctions où vous pouvez surveiller des statistiques de votre application et créer des alertes qui se déclenchent quand des seuils sont dépassés.

Pour de plus amples informations sur la surveillance de l'état fournie par Elastic Beanstalk, veuillez consulter Création de rapports d'intégrité de base.

Vous pouvez accéder aux informations opérationnelles relatives à votre application en utilisant le AWS Toolkit for Visual Studio ou la console AWS de gestion.

La boîte à outils affiche le statut de votre environnement et l'état de votre application dans le champ Status (Statut).



Pour surveiller l'intégrité de l'application

1. Dans le AWS Toolkit for Visual Studio, dans AWS Explorer, développez le nœud Elastic Beanstalk, puis étendez le nœud de votre application.
2. Cliquez avec le bouton droit sur votre environnement Elastic Beanstalk, puis cliquez sur View Status (Afficher le statut).
3. Dans l'onglet de l'environnement de votre application, cliquez sur Monitoring (Surveillance).

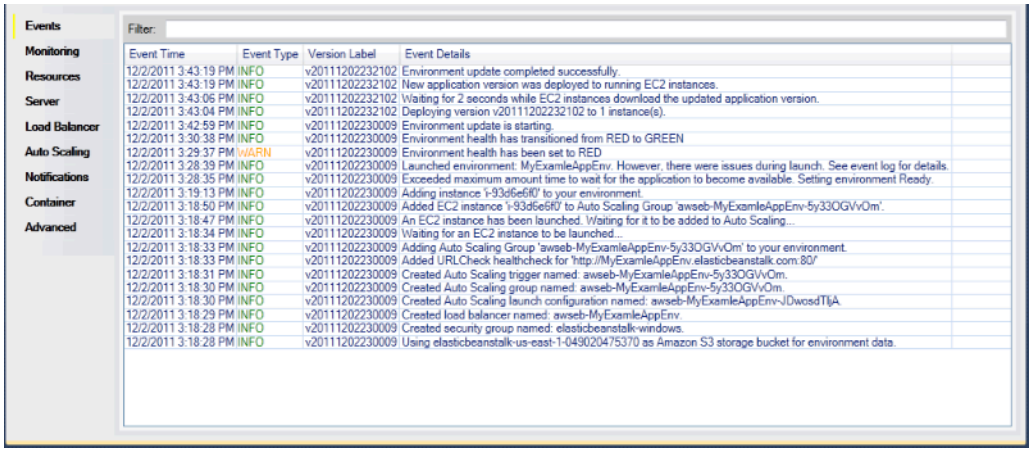
   Le panneau Monitoring (Surveillance) comprend un ensemble de graphiques illustrant l'utilisation des ressources pour votre environnement d'application spécifique.

> **Note**
>
> Par défaut, la plage de temps est définie sur la dernière heure. Pour modifier ce paramètre, dans la liste Time Range (Plage horaire), cliquez sur une plage de temps différente.

Vous pouvez utiliser le AWS Toolkit for Visual Studio ou la console AWS de gestion pour afficher les événements associés à votre application.

Pour afficher des événements d'application

1. Dans le AWS Toolkit for Visual Studio, dans AWS Explorer, développez le nœud Elastic Beanstalk et le nœud de votre application.
2. Cliquez avec le bouton droit sur votre AWS environnement Elastic Beanstalk dans Explorer, puis cliquez sur Afficher le statut.
3. Dans l'onglet de l'environnement de votre application, cliquez sur Events (Événements).



Déploiement d'applications Elastic Beanstalk dans .NET à l'aide d'outils de déploiement AWS

AWS propose deux outils de ligne de commande pour déployer des applications .NET Core et ASP.NET Core sur Elastic Beanstalk :

- AWS Outil de déploiement .NET : outil de ligne de commande moderne qui simplifie le déploiement d'applications .NET sur des AWS services, notamment Elastic Beanstalk. Cet outil fournit une expérience interactive et peut détecter automatiquement le meilleur AWS service pour votre application. Pour plus d'informations, consultez la section Outil de déploiement AWS .NET.
- AWS Extensions pour .NET CLI : ensemble d'extensions .NET CLI qui fournissent des commandes pour déployer des applications .NET Core sur Elastic AWS Beanstalk et d'autres services. Ces extensions s'intègrent au flux de travail standard .NET CLI. Pour plus d'informations, consultez AWS Extensions pour .NET CLI.

Migration de votre application .NET sur site vers Elastic Beanstalk

AWS Elastic Beanstalk fournit un chemin de migration rationalisé pour vos applications Windows exécutées sur Internet Information Services (IIS) via l'interface de ligne de commande Elastic Beanstalk (EB CLI). La eb migrate commande découvre automatiquement vos sites, applications et répertoires virtuels IIS, préserve leurs configurations et les déploie AWS dans le cloud.

Cette fonctionnalité de migration intégrée propose une approche plus simple qui réduit la complexité et le temps généralement associés aux migrations vers le cloud. Le processus de migration permet de préserver les fonctionnalités de l'application et l'intégrité de la configuration pendant la transition vers AWS.

Pour obtenir des instructions complètes et détaillées sur la migration de vos applications IIS vers AWS Elastic Beanstalk, reportez-vous au Migration des applications IIS chapitre de ce guide.

Recommandations pour les composants retirés de Windows Server sur Elastic Beanstalk

Cette rubrique fournit des recommandations si vos applications s'exécutent actuellement sur les branches de la plate-forme Windows Server 2012 R2 retirées. Il aborde également la question de la prise en charge obsolète des versions des protocoles TLS 1.0 et 1.1 sur nos points de terminaison d'API de AWS service et sur les branches de plateforme concernées.

Mise hors service des branches de la plateforme Windows Server 2012 R2

Elastic Beanstalk a retiré les branches de la plateforme Windows Server 2012 R2 le 4 décembre 2023 et AMIs a rendu privées les branches associées à ces plateformes le 10 avril 2024. Cette action empêche le lancement d'instances dans vos environnements Windows Server 2012 qui utilisent l'AMI Beanstalk par défaut.

Si certains de vos environnements s'exécutent sur des branches de plate-forme Windows retirées, nous vous recommandons de les migrer vers l'une des plateformes Windows Server suivantes, qui sont à jour et entièrement prises en charge :

- Windows Server 2022 avec IIS 10.0 version 2.x
- Windows Server 2019 avec IIS 10.0 version 2.x

Pour des considérations complètes sur la migration, consultez Migration à partir de versions majeures antérieures de la plateforme Windows Server.

Pour plus d'informations sur l'obsolescence de la plateforme, consultez Stratégie de prise en charge de la plateforme Elastic Beanstalk.

> **Note**
>
> Si vous ne parvenez pas à effectuer la migration vers ces plateformes entièrement prises en charge, nous vous recommandons d'utiliser une image personnalisée AMIs créée avec Windows Server 2012 R2 ou Windows Server 2012 R2 Core AMIs comme image de base, si ce n'est pas déjà fait. Pour obtenir des instructions complètes, consultez Conservation de l'accès à une Amazon Machine Image (AMI) pour une plateforme retirée. Contactez le AWS Support Center si vous avez besoin d'un accès temporaire à une AMI pendant que vous effectuez l'une de ces étapes de migration.

Compatibilité avec TLS 1.2

Au 31 décembre 2023, AWS a commencé à appliquer pleinement le protocole TLS 1.2 sur tous les points de terminaison de l' AWS API. Cette action a supprimé la possibilité d'utiliser les versions 1.0 et 1.1 de TLS avec tous AWS APIs. Cette information a été initialement communiquée le 28 juin 2022. Pour éviter tout risque d'impact sur la disponibilité, mettez à niveau tous les environnements

exécutant les versions de plate-forme identifiées ici vers une version plus récente dès que possible, si ce n'est déjà fait.

Incidence potentielle

Les versions des plateformes Elastic Beanstalk qui exécutent le protocole TLS v1.1 ou une version antérieure sont concernées. Cette modification a un impact sur les actions environnementales qui incluent, sans toutefois s'y limiter, les déploiements de configuration, les déploiements d'applications, le dimensionnement automatique, le lancement d'un nouvel environnement, la rotation des journaux, les rapports de santé améliorés et la publication des journaux d'applications dans le compartiment Amazon S3 associé à vos applications.

Versions de la plateforme Windows concernées

Il est conseillé aux clients disposant d'environnements Elastic Beanstalk sur la version de plateforme suivante de mettre à niveau chacun de leurs environnements correspondants vers la version de plateforme Windows 2.8.3 ou une version ultérieure, publiée le 18 février 2022.

- Windows Server 2019 – version de plateforme 2.8.2 ou versions antérieures

Il est conseillé aux clients disposant d'environnements Elastic Beanstalk sur les versions de plateforme suivantes de mettre à niveau chacun de leurs environnements correspondants vers la version de plateforme Windows 2.10.7 ou une version ultérieure, publiée le 28 décembre 2022.

- Windows Server 2016 – version de plateforme 2.10.6 ou versions antérieures
- Windows Server 2012 : toutes les versions de la plateforme ; cette plateforme a été retirée le 4 décembre 2023
- Windows Server 2008 – toutes les versions de plateforme ; cette plateforme a été retirée le 28 octobre 2019

Pour une liste des versions de plateforme Windows Server prises en charge les plus récentes, veuillez consulter la rubrique Plateformes prises en charge dans le guide Plateformes AWS Elastic Beanstalk .

Pour plus de détails et de bonnes pratiques sur la mise à jour de votre environnement, consultez Mise à jour de la version de la plateforme de votre environnement Elastic Beanstalk.

Déploiement d'applications .NET Core (Linux) avec Elastic Beanstalk

> **Consultez le .NET on AWS Developer Center**
>
> Vous êtes-vous déjà rendu dans notre centre de développement .Net ? C'est notre guichet unique pour tout ce qui concerne .NET AWS.
> Pour plus d'informations, consultez le .NET on AWS Developer Center.

Ce chapitre fournit des instructions pour configurer et déployer votre application Web .NET Core (Linux) sur AWS Elastic Beanstalk. Elastic Beanstalk facilite le déploiement, la gestion et le dimensionnement de vos applications Web .NET Core (Linux) à l'aide d'Amazon Web Services.

Vous pouvez déployer votre application en quelques minutes à l'aide de l'interface de ligne de commande Elastic Beanstalk (EB CLI) ou de la console Elastic Beanstalk. Après avoir déployé votre application Elastic Beanstalk, vous pouvez continuer à utiliser l'EB CLI pour gérer votre application et votre environnement, ou vous pouvez utiliser la console Elastic Beanstalk, ou le. AWS CLI APIs

Suivez les étapes décrites dans les step-by-step instructions QuickStart pour .NET Core sous Linux pour créer et déployer une application Web ASP.NET Core Hello World avec l'interface de ligne de commande EB.

Rubriques

- QuickStart: Déployer une application .NET Core sous Linux sur Elastic Beanstalk
- Configuration de votre environnement de développement .NET core sur Linux pour Elastic Beanstalk
- Utilisation du noyau d'Elastic Beanstalk .NET sur la plateforme Linux
- Le AWS Toolkit for Visual Studio - Travailler avec .Net Core sur Elastic Beanstalk
- Migration de .NET sur la plateforme Windows Server vers la plateforme .NET Core sur Linux sur Elastic Beanstalk

QuickStart: Déployer une application .NET Core sous Linux sur Elastic Beanstalk

Ce QuickStart didacticiel explique le processus de création d'une application .NET Core sous Linux et son déploiement dans un AWS Elastic Beanstalk environnement.

> ⚠ Non destiné à une utilisation en production
>
> Les exemples sont destinés uniquement à des fins de démonstration. N'utilisez pas d'exemples d'applications en production.

Sections

- Votre AWS compte
- Conditions préalables
- Étape 1 : Création d'une application .NET Core sous Linux
- Étape 2 : Exécutez votre application localement
- Étape 3 : Déployez votre application .NET Core sur Linux avec l'interface de ligne de commande EB
- Étape 4 : Exécutez votre application sur Elastic Beanstalk
- Étape 5 : nettoyer
- AWS ressources pour votre application
- Étapes suivantes
- Déployez avec la console Elastic Beanstalk

Votre AWS compte

Si vous n'êtes pas encore AWS client, vous devez créer un AWS compte. L'inscription vous permet d'accéder à Elastic Beanstalk AWS et aux autres services dont vous avez besoin.

Si vous avez déjà un AWS compte, vous pouvez passer àConditions préalables.

Créez un AWS compte

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez l'https://portal.aws.amazon.com/billing/inscription.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d'inscription, vous recevrez un appel téléphonique ou un SMS et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

   Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l'utilisateur racine a accès à l'ensemble des Services AWS et des ressources de ce compte. La meilleure pratique de sécurité consiste à attribuer un accès administratif à un utilisateur, et à utiliser uniquement l'utilisateur racine pour effectuer les tâches nécessitant un accès utilisateur racine.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. À tout moment, vous pouvez consulter l'activité actuelle de votre compte et gérer votre compte en accédant à https://aws.amazon.com/et en choisissant Mon compte.

Création d'un utilisateur doté d'un accès administratif

Une fois que vous vous êtes inscrit à un utilisateur administratif Compte AWS, que vous Utilisateur racine d'un compte AWS l'avez sécurisé AWS IAM Identity Center, que vous l'avez activé et que vous en avez créé un, afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l'aide pour vous connecter en utilisant l'utilisateur racine, consultez Connexion en tant qu'utilisateur racine dans le Guide de l'utilisateur Connexion à AWS .
2. Activez l'authentification multifactorielle (MFA) pour votre utilisateur racine.

Pour obtenir des instructions, consultez la section Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d'un utilisateur doté d'un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d' AWS IAM Identity Center dans le Guide de l'utilisateur AWS IAM Identity Center .
2. Dans IAM Identity Center, octroyez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connexion en tant qu'utilisateur doté d'un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l'URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l'utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribution d'un accès à d'autres utilisateurs

1. Dans IAM Identity Center, créez un ensemble d'autorisations qui respecte la bonne pratique consistant à appliquer les autorisations de moindre privilège.

   Pour obtenir des instructions, consultez Création d'un ensemble d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .
2. Attribuez des utilisateurs à un groupe, puis attribuez un accès par authentification unique au groupe.

   Pour obtenir des instructions, consultez Ajout de groupes dans le Guide de l'utilisateur AWS IAM Identity Center .

Conditions préalables

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

INTERFACE DE LIGNE DE COMMANDE (CLI) EB

Ce tutoriel utilise également l'interface de ligne de commande Elastic Beanstalk (CLI EB). Pour de plus amples informations sur l'installation et la configuration de la CLI EB, veuillez consulter Installation de la CLI EB avec un script de configuration (recommandé) et Configuration de l'interface de ligne de commande EB.

.NET Core sous Linux

Si le SDK .NET n'est pas installé sur votre ordinateur local, vous pouvez l'installer en suivant le lien Télécharger .NET sur le site Web de documentation .NET.

Vérifiez votre installation du SDK .NET en exécutant la commande suivante.

```
~$ dotnet --info
```

Étape 1 : Création d'une application .NET Core sous Linux

Créez un répertoire de projet.

```
~$ mkdir eb-dotnetcore
~$ cd eb-dotnetcore
```

Créez ensuite un exemple d'application Hello World en exécutant les commandes suivantes.

```
~/eb-dotnetcore$ dotnet new web --name HelloElasticBeanstalk
```

```
~/eb-dotnetcore$ cd HelloElasticBeanstalk
```

Étape 2 : Exécutez votre application localement

Exécutez la commande suivante pour exécuter votre application localement.

```
~/eb-dotnetcore/HelloElasticBeasntalk$ dotnet run
```

Le résultat doit ressembler au texte suivant.

```
Building...
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: https://localhost:7294
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5052
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
```

> **Note**
>
> La `dotnet` commande sélectionne un port au hasard lors de l'exécution locale de l'application. Dans cet exemple, le port est 5052. Lorsque vous déployez l'application dans votre environnement Elastic Beanstalk, elle s'exécute sur le port 5000.

Entrez l'adresse URL `http://localhost:`*`port`* dans votre navigateur Web. Pour cet exemple spécifique, la commande est`http://localhost:5052`. Le navigateur Web doit afficher « Hello World ! ».

Étape 3 : Déployez votre application .NET Core sur Linux avec l'interface de ligne de commande EB

Exécutez les commandes suivantes pour créer un environnement Elastic Beanstalk pour cette application.

Pour créer un environnement et déployer votre application .NET Core sur Linux

1. Compilez et publiez votre application dans un dossier pour la déployer dans l'environnement Elastic Beanstalk que vous êtes sur le point de créer.

```
~$ cd eb-dotnetcore/HelloElasticBeanstalk
~/eb-dotnetcore/HelloElasticBeanstalk$ dotnet publish -o site
```

2. Accédez au répertoire du site dans lequel vous venez de publier votre application.

```
~/eb-dotnetcore/HelloElasticBeanstalk$ cd site
```

3. Initialisez votre référentiel de la CLI EB avec la commande eb init :

   Tenez compte des informations suivantes concernant la version de branche de plate-forme que vous spécifiez dans la commande :

   - Remplacez *x.y.z* la commande suivante par la dernière version de la branche de plate-forme .NET 6 le AL2 023.
   - Pour trouver la dernière version de la branche de plate-forme, consultez .NET Core sur les plateformes prises en charge par Linux dans le guide AWS Elastic Beanstalk des plateformes.
   - Un exemple de nom de pile de solutions incluant le numéro de version est`64bit-amazon-linux-2023-v`**`3.1.1`**`-running-.net-6`. Dans cet exemple, la version de la branche est 3.1.1.

```
~eb-dotnetcore/HelloElasticBeanstalk/site$ eb init -p 64bit-amazon-linux-2023-
vx.y.z-running-.net-6 dotnetcore-tutorial --region us-east-2
Application dotnetcore-tutorial has been created.
```

   Cette commande crée une application nommée `dotnetcore-tutorial` et configure votre dépôt local pour créer des environnements avec la version de plate-forme .NET Core sur Linux spécifiée dans la commande.

4. (Facultatif) Exécutez à eb init nouveau pour configurer une paire de clés par défaut afin de pouvoir utiliser SSH pour vous connecter à l' EC2 instance qui exécute votre application.

```
~eb-dotnetcore/HelloElasticBeanstalk/site$ eb init
Do you want to set up SSH for your instances?
(y/n): y
```

```
Select a keypair.
1) my-keypair
2) [ Create new KeyPair ]
```

Sélectionnez une paire de clés si vous en avez déjà une, ou suivez les invites pour en créer une. Si vous ne voyez pas l'invite ou que vous avez besoin de modifier vos paramètres ultérieurement, exécutez eb init -i.

5. Créez un environnement et déployez-y votre application avec eb create. Elastic Beanstalk crée automatiquement un fichier zip pour votre application et le démarre sur le port 5000.

   to

   ```
   ~eb-dotnetcore/HelloElasticBeanstalk/site$ eb create dotnet-tutorial
   ```

   Il faut environ cinq minutes à Elastic Beanstalk pour créer votre environnement.

Étape 4 : Exécutez votre application sur Elastic Beanstalk

Lorsque le processus de création de votre environnement est terminé, ouvrez votre site Web aveceb open.

```
~eb-dotnetcore/HelloElasticBeanstalk/site$ eb open
```

Félicitations ! Vous avez déployé une application .NET Core sur Linux avec Elastic Beanstalk ! Celle-ci ouvre une fenêtre de navigation en utilisant le nom de domaine créé pour votre application.

Étape 5 : nettoyer

Vous pouvez mettre fin à votre environnement lorsque vous avez fini d'utiliser votre application. Elastic Beanstalk AWS met fin à toutes les ressources associées à votre environnement.

Pour mettre fin à votre environnement Elastic Beanstalk avec l'EB CLI, exécutez la commande suivante.

```
~eb-dotnetcore/HelloElasticBeanstalk/site$ eb terminate
```

AWS ressources pour votre application

Vous venez de créer une application à instance unique. Il s'agit d'un exemple d'application simple avec une seule EC2 instance, il ne nécessite donc pas d'équilibrage de charge ni de dimensionnement automatique. Pour les applications à instance unique, Elastic Beanstalk crée les ressources suivantes : AWS

- EC2 instance — Une machine EC2 virtuelle Amazon configurée pour exécuter des applications Web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble distinct de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse qui traite le trafic web devant votre application web, lui transmet les demandes, traite les ressources statiques et génère des journaux d'accès et d'erreurs.
- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack : Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

Elastic Beanstalk gère toutes ces ressources. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

Étapes suivantes

Dès que vous disposez d'un environnement exécutant une application, vous pouvez à tout moment déployer une nouvelle version de l'application ou une application totalement différente. Le déploiement d'une nouvelle version d'application est très rapide car il ne nécessite ni le provisionnement ni le redémarrage des instances. EC2 Vous pouvez également explorer votre nouvel environnement à l'aide de la console Elastic Beanstalk. Pour connaître les étapes détaillées, consultez la section Explorez votre environnement dans le chapitre Mise en route de ce guide.

Une fois que vous avez déployé un ou deux exemples d'applications et que vous êtes prêt à commencer à développer et à exécuter des applications .NET Core sur Linux localement, consultezConfiguration de votre environnement de développement .NET core sur Linux pour Elastic Beanstalk.

Déployez avec la console Elastic Beanstalk

Vous pouvez également utiliser la console Elastic Beanstalk pour lancer l'exemple d'application. Pour connaître les étapes détaillées, voir Création d'un exemple d'application dans le chapitre Mise en route de ce guide.

Configuration de votre environnement de développement .NET core sur Linux pour Elastic Beanstalk

Cette rubrique fournit des instructions pour configurer un environnement de développement .NET Core sur Linux afin de tester votre application localement avant de la déployer sur AWS Elastic Beanstalk. Il fait également référence à des sites Web qui fournissent des instructions d'installation pour des outils utiles.

Sections

- Installation du kit SDK .NET Core
- Installation d'un IDE
- Installation du AWS Toolkit for Visual Studio

Installation du kit SDK .NET Core

Vous pouvez utiliser le kit SDK .NET Core pour développer des applications qui s'exécutent sur Linux.

Consultez la page de téléchargements .NET pour télécharger et installer le kit SDK .NET Core.

Installation d'un IDE

Les environnements de développement intégrés (IDEs) fournissent une gamme de fonctionnalités qui facilitent le développement d'applications. Si vous n'avez pas utilisé un IDE pour le développement .NET, essayez Visual Studio Community pour démarrer.

Veuillez consulter la page Visual Studio Community pour télécharger et installer Visual Studio Community.

Installation du AWS Toolkit for Visual Studio

Il s'AWS Toolkit for Visual Studioagit d'un plugin open source pour l'IDE Visual Studio qui permet aux développeurs de développer, de déboguer et de déployer plus facilement des applications .NET à l'aide AWS de. Pour obtenir des instructions d'installation, veuillez consulter la page d'accueil Toolkit for Visual Studio.

Utilisation du noyau d'Elastic Beanstalk .NET sur la plateforme Linux

Cette rubrique explique comment configurer, créer et exécuter votre noyau .NET sur des applications Linux sur Elastic Beanstalk.

AWS Elastic Beanstalk prend en charge un certain nombre de branches de plate-forme pour différentes versions du framework .NET Core qui s'exécutent sur le système d'exploitation Linux. Voir .NET core sous Linux dans la section AWS Elastic Beanstalk Plateformes pour une liste complète.

Pour plus d'informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, consultez the section called “Extension des plateformes Linux”.

Considérations relatives à .NET Core sur la plate-forme Linux

Serveur proxy

La plateforme Elastic Beanstalk .NET Core sous Linux inclut un proxy inverse qui transmet les demandes à votre application. Par défaut, Elastic Beanstalk utilise NGINX comme serveur proxy. Vous pouvez choisir de n'utiliser aucun serveur proxy et de configurer Kestrel en tant que serveur Web. Kestrel est inclus par défaut dans les modèles de projet ASP.NET Core.

Structure de l'application

Vous pouvez publier des applications dépendantes de l'exécution qui utilisent l'environnement d'exécution .NET Core fourni par Elastic Beanstalk. Vous pouvez également publier des applications autonomes qui incluent l'environnement d'exécution .NET Core et les dépendances de votre application dans le bundle de fichiers source. Pour en savoir plus, veuillez consulter la section the section called “Regroupement d'applications”.

Configuration de plateforme

Pour configurer les processus qui s'exécutent sur les instances de serveur dans votre environnement, incluez un fichier Procfile facultatif dans votre bundle de fichiers source. Un fichier `Procfile` est obligatoire si vous avez plus d'une application dans votre bundle de fichiers source.

Nous vous recommandons de toujours fournir un fichier `Procfile` dans le bundle de fichiers source avec votre application. De cette façon, vous contrôlez précisément les processus Elastic Beanstalk exécutés pour votre application.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour modifier la configuration d'un environnement en cours d'exécution. Pour éviter de perdre la configuration de votre environnement en le résiliant, vous pouvez utiliser des configurations enregistrées pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des fichiers de configuration. Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

Les paramètres appliqués dans la console Elastic Beanstalk remplacent les mêmes paramètres des fichiers de configuration, s'ils existent. Cela vous permet d'utiliser les paramètres par défaut dans les fichiers de configuration et de les remplacer par des paramètres spécifiques à l'environnement dans la console. Pour plus d'informations sur la priorité et les autres méthodes de modification des paramètres, consultez Options de configuration.

Configuration de votre environnement .NET Core sous Linux

Les paramètres de la plateforme .NET Core sous Linux vous permettent d'affiner le comportement de vos instances Amazon EC2. Vous pouvez modifier la configuration des instances Amazon EC2 de l'environnement Elastic Beanstalk à l'aide de la console Elastic Beanstalk.

Utilisez la console Elastic Beanstalk pour permettre la rotation des journaux sur Amazon S3 et configurer des variables que votre application peut lire à partir de l'environnement.

Pour configurer votre environnement .NET Core sous Linux à l'aide de la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.

Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Enable log file rotation to Amazon S3 (Permettre la rotation du fichier journal sur Amazon S3) – Indique si les fichiers journaux des instances Amazon EC2 de votre application doivent être copiés dans le compartiment Amazon S3 associé à votre application.

Propriétés de l'environnement

La section Environment Properties (Propriétés de l'environnement) vous permet de spécifier des paramètres de configuration de l'environnement sur les instances Amazon EC2 exécutant votre application. Les propriétés de l'environnement sont passées en tant que paires clé-valeur à l'application.

Dans l'environnement .NET Core sous Linux en cours d'exécution dans Elastic Beanstalk, les variables d'environnement sont accessibles à l'aide de `Environment.GetEnvironmentVariable("variable-name")`. Par exemple, vous pouvez lire une propriété nommée `API_ENDPOINT` sur une variable avec le code suivant :

```
string endpoint = Environment.GetEnvironmentVariable("API_ENDPOINT");
```

Pour plus d'informations, consultez Variables d'environnement et autres paramètres du logiciel.

Espace de noms de la configuration .NET Core sous Linux

Vous pouvez utiliser un fichier de configuration pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être spécifiques à la plate-forme ou s'appliquer à toutes les plateformes du service Elastic Beanstalk dans son ensemble. Les options de configuration sont organisées en espaces de noms.

La plateforme .NET Core sous Linux prend en charge les options des espaces de noms suivants en plus des options prises en charge pour tous les environnements Elastic Beanstalk :

- `aws:elasticbeanstalk:environment:proxy`— Choisissez d'utiliser NGINX ou de ne pas utiliser de serveur proxy. Les valeurs valides sont `nginx` ou `none`.

L'exemple de fichier de configuration suivant illustre l'utilisation d'options de configuration spécifiques à .NET Core sous Linux :

Example.ebextensions/proxy-settings.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: none
```

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d' AWS CLI. Pour plus d'informations, consultez Options de configuration.

Regroupement d'applications pour la plateforme .NET Core sur Linux Elastic Beanstalk

Vous pouvez exécuter à la fois des applications .NET Core dépendantes de l'exécution et autonomes sur AWS Elastic Beanstalk.

Une application dépendante de l'exécution utilise un environnement d'exécution .NET Core fourni par Elastic Beanstalk pour exécuter votre application. Elastic Beanstalk utilise le fichier `runtimeconfig.json` de votre bundle de fichiers source afin de déterminer l'environnement

d'exécution à utiliser pour votre application. Elastic Beanstalk choisit l'environnement d'exécution compatible le plus récent disponible pour votre application.

Une application autonome inclut l'environnement d'exécution .NET Core, votre application et ses dépendances. Pour utiliser une version de l'environnement d'exécution .NET Core qu'Elastic Beanstalk n'inclut pas dans ses plateformes, fournissez une application autonome.

Exemples

Vous pouvez compiler à la fois des applications autonomes et dépendantes de l'environnement d'exécution à l'aide de la commande `dotnet publish`. Pour en savoir plus sur la publication d'applications .NET Core, veuillez consulter la présentation de la publication d'applications .NET Core dans la documentation .NET Core.

L'exemple de structure de fichiers suivant définit une application unique qui utilise un environnement d'exécution .NET Core fourni par Elastic Beanstalk.

```
### appsettings.Development.json
### appsettings.json
### dotnetcoreapp.deps.json
### dotnetcoreapp.dll
### dotnetcoreapp.pdb
### dotnetcoreapp.runtimeconfig.json
### web.config
### Procfile
### .ebextensions
### .platform
```

Vous pouvez inclure plusieurs applications dans votre bundle de fichiers source. L'exemple suivant définit deux applications à exécuter sur le même serveur Web. Pour exécuter plusieurs applications, vous devez inclure un fichier Procfile dans votre bundle de fichiers source. Pour un exemple complet d'application, consultez le dotnet-core-linux-multiplefichier -apps.zip.

```
### DotnetMultipleApp1
#   ### Amazon.Extensions.Configuration.SystemsManager.dll
#   ### appsettings.Development.json
#   ### appsettings.json
#   ### AWSSDK.Core.dll
#   ### AWSSDK.Extensions.NETCore.Setup.dll
#   ### AWSSDK.SimpleSystemsManagement.dll
```

```
#   ### DotnetMultipleApp1.deps.json
#   ### DotnetMultipleApp1.dll
#   ### DotnetMultipleApp1.pdb
#   ### DotnetMultipleApp1.runtimeconfig.json
#   ### Microsoft.Extensions.PlatformAbstractions.dll
#   ### Newtonsoft.Json.dll
#   ### web.config
### DotnetMultipleApp2
#   ### Amazon.Extensions.Configuration.SystemsManager.dll
#   ### appsettings.Development.json
#   ### appsettings.json
#   ### AWSSDK.Core.dll
#   ### AWSSDK.Extensions.NETCore.Setup.dll
#   ### AWSSDK.SimpleSystemsManagement.dll
#   ### DotnetMultipleApp2.deps.json
#   ### DotnetMultipleApp2.dll
#   ### DotnetMultipleApp2.pdb
#   ### DotnetMultipleApp2.runtimeconfig.json
#   ### Microsoft.Extensions.PlatformAbstractions.dll
#   ### Newtonsoft.Json.dll
#   ### web.config
### Procfile
### .ebextensions
### .platform
```

Utilisation d'un Procfile pour configurer votre environnement .NET Core sur Linux Elastic Beanstalk

Pour exécuter plusieurs applications sur le même serveur web, vous devez inclure dans votre bundle de fichiers source un fichier `Procfile` qui indique à Elastic Beanstalk les applications à exécuter.

Nous vous recommandons de toujours fournir un fichier `Procfile` dans le bundle de fichiers source avec votre application. De cette façon, vous contrôlez précisément les processus Elastic Beanstalk qui s'exécutent pour votre application et les arguments que ces processus reçoivent.

L'exemple suivant utilise un fichier `Procfile` pour spécifier deux applications qu'Elastic Beanstalk doit exécuter sur le même serveur web.

Example Procfile

```
web: dotnet ./dotnet-core-app1/dotnetcoreapp1.dll
```

```
web2: dotnet ./dotnet-core-app2/dotnetcoreapp2.dll
```

Pour plus de détails sur l'écriture et l'utilisation d'un`Procfile`, voirBuildfile et Procfile.

Configuration du serveur proxy

AWS Elastic Beanstalk utilise NGINX comme proxy inverse pour relayer les demandes vers votre application. Elastic Beanstalk fournit une configuration NGINX par défaut que vous pouvez étendre ou remplacer complètement par votre propre configuration.

Par défaut, Elastic Beanstalk configure le proxy NGINX pour transmettre les demandes à votre application sur le port 5000. Vous pouvez remplacer le port par défaut en définissant la propriété d'environnement P0RT sur le port que votre application écoute.

> **Note**
>
> Le port sur lequel votre application écoute n'affecte pas le port sur lequel le serveur NGINX écoute pour recevoir les demandes de l'équilibreur de charge.

Configuration du serveur proxy sur la version de votre plateforme

Toutes les AL2 plateformes AL2023/prennent en charge une fonctionnalité de configuration de proxy uniforme. Pour plus d'informations sur la configuration du serveur proxy sur les versions de votre plate-forme exécutant AL2023/AL2, consultezConfiguration du proxy inverse.

L'exemple de fichier de configuration suivant étend la configuration NGINX de votre environnement. La configuration dirige les demandes `/api` vers une deuxième application Web qui écoute sur le port 5200 du serveur Web. Par défaut, Elastic Beanstalk transmet les demandes à une seule application qui écoute sur le port 5000.

Example **`01_custom.conf`**

```
location /api {
     proxy_pass          http://127.0.0.1:5200;
     proxy_http_version  1.1;

     proxy_set_header   Upgrade $http_upgrade;
     proxy_set_header   Connection $http_connection;
     proxy_set_header   Host $host;
```

```
    proxy_cache_bypass $http_upgrade;
    proxy_set_header   X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header   X-Forwarded-Proto $scheme;
}
```

Le AWS Toolkit for Visual Studio - Travailler avec .Net Core sur Elastic Beanstalk

Cette rubrique explique comment vous pouvez effectuer les tâches suivantes à l'aide de AWS Toolkit for Visual Studio :

- Créer une application web ASP.NET Core à l'aide d'un modèle Visual Studio.
- Créer un environnement Elastic Beanstalk Amazon Linux.
- Déployer l'application web ASP.NET Core dans le nouvel environnement Amazon Linux.

Cette rubrique explique également comment vous pouvez utiliser le AWS Toolkit for Visual Studio pour gérer vos environnements d'applications Elastic Beanstalk et surveiller l'état de santé de votre application.

AWS Toolkit for Visual Studio Il s'agit d'un plugin de l'IDE Visual Studio. Avec la boîte à outils, vous pouvez déployer et gérer des applications dans Elastic Beanstalk pendant que vous travaillez dans votre environnement Visual Studio.

Sections

- Conditions préalables
- Créer un nouveau projet d'application
- Création d'un environnement Elastic Beanstalk et déploiement de votre application
- Résiliation d'un environnement
- Gestion de vos environnements d'application Elastic Beanstalk
- Surveillance de l'intégrité d'une application

Conditions préalables

Avant de commencer ce didacticiel, vous devez installer le AWS Toolkit for Visual Studio. Pour obtenir des instructions, veuillez consulter Configuration de l' AWS Toolkit for Visual Studio.

Si vous n'avez jamais utilisé le kit d'outils auparavant, la première chose à faire après l'avoir installé est d'enregistrer vos AWS informations d'identification auprès du kit d'outils. Pour de plus amples informations sur ce point, veuillez consulter Fournire les informations d'identification AWS.

Créer un nouveau projet d'application

Si vous n'avez pas de projet d'application .NET Core dans Visual Studio, vous pouvez facilement en créer un en utilisant l'un des modèles de projet Visual Studio.

Pour créer un projet d'application web ASP.NET Core

1. Dans Visual Studio, dans le menu File (Fichier), choisissez New (Nouveau), puis choisissez Project (Projet).
2. Dans la boîte de dialogue Create a new project (Créer un nouveau projet) sélectionnez C#, Linux, puis Cloud.
3. Dans la liste des modèles de projet qui s'affiche, sélectionnez ASP.NET Core Web Application (Application web ASP.NET Core), puis sélectionnez Next (Suivant).

   > **Note**
   >
   > Si vous ne voyez pas l'application web ASP.NET Core répertoriée dans les modèles de projet, vous pouvez l'installer lorsque vous êtes dans Visual Studio.
   >
   > 1. Faites défiler jusqu'au bas de la liste des modèles et sélectionnez le lien Installer d'autres outils et fonctionnalités qui se trouve sous la liste des modèles.
   > 2. Si vous êtes invité à autoriser l'application Visual Studio à apporter des modifications à votre appareil, sélectionnez Yes (Oui).
   > 3. Choisissez l'onglet Workloads (Charges de travail), puis sélectionnez ASP.NET and web development (ASP.NET et développement web).
   > 4. Sélectionnez le bouton Modify (Modifier). Le Visual Studio Installer (programme d'installation de Visual Studio) installe le modèle de projet.
   > 5. Une fois le programme d'installation terminé, quittez le panneau pour revenir à l'endroit où vous vous êtes arrêté dans Visual Studio.

4. Dans la boîte de dialogue Configure your new project (Configurer votre nouveau projet) saisissez un nom de projet. Le nom de la solution est par défaut le nom de votre projet. Ensuite, choisissez Créer.

5. Dans la boîte de dialogue Create a new ASP.NET Core web application (Créer une nouvelle application web ASP.NET Core) sélectionnez .NET Core, puis ASP.NET Core 3.1. Dans la liste des types d'application affichée, sélectionnez Web Application (Application web), puis cliquez sur le bouton Create (Créer).

Visual Studio affiche la boîte de dialogue Creating Project (Création du projet) au moment de la création de votre application. Une fois la génération de votre application terminée, Visual Studio affiche un panneau avec le nom de votre application.

Création d'un environnement Elastic Beanstalk et déploiement de votre application

Cette section décrit comment créer un environnement Elastic Beanstalk pour votre application et déployer votre application dans cet environnement.

Pour créer un environnement et déployer votre application

1. Dans Visual Studio, sélectionnez View (Afficher), puis Solution Explorer (Explorateur de solutions).
2. Dans Solution Explorer (Explorateur de solutions), ouvrez le menu contextuel (clic droit) de votre application, puis sélectionnez Publish to (Publier dans) AWS Elastic Beanstalk.
3. Dans l'assistant Publish to AWS Elastic Beanstalk (Publier dans Amazon Elastic Beanstalk), entrez les informations de votre compte.

    a. Pour Account profile to use (Profil de compte à utiliser), sélectionnez votre compte par défaut ou cliquez sur l'icône Add another account (Ajouter un autre compte) pour entrer les informations d'un nouveau compte.

    b. Pour Region, sélectionnez la région où vous souhaitez déployer votre application. Pour plus d'informations sur AWS les régions disponibles, consultez la section AWS Elastic Beanstalk Points de terminaison et quotas dans le Références générales AWS. Si vous sélectionnez une région qui n'est pas prise en charge par Elastic Beanstalk, l'option de déploiement sur Elastic Beanstalk devient indisponible.

    c. Sélectionnez Create a new application environment (Créer un nouvel environnement d'application), puis choisissez Next (Suivant).
4. Dans la boîte de dialogue Application Environment (Environnement d'application) entrez les détails de votre nouvel environnement d'application.

5. Dans la boîte de dialogue AWSd'options suivante, définissez EC2 les options Amazon et les autres options AWS associées pour votre application déployée.

   a. Pour le type de conteneur, sélectionnez Amazon Linux 2 v 64 bits ***<n.n.n>*** exécutant .NET Core.

   > **Note**
   >
   > Nous vous recommandons de sélectionner la version actuelle de la plateforme de Linux. Cette version contient les derniers correctifs de sécurité et de bogues inclus dans notre dernière Amazon Machine Image (AMI).

   b. Pour Instance Type (Type d'instance), sélectionnez t2.micro. (Le choix d'un type d'instance micro réduit le coût associé à l'exécution de l'instance.)

   c. Pour Key pair (Paire de clés), sélectionnez Create new key pair (Créer une paire de clés). Entrez un nom pour la nouvelle paire de clés, puis choisissez OK. (Dans cet exemple, nous utilisons **`myuseastkeypair`**.) Une paire de clés permet d'accéder à distance à vos instances Amazon EC2 . Pour plus d'informations sur les paires de EC2 clés Amazon, consultez la section Utilisation des informations d'identification dans le guide de l'utilisateur d'Amazon Elastic Compute Cloud.

   d. Si l'application est simple et génère un faible trafic, sélectionnez Single instance environment (Environnement à une seule instance). Pour de plus amples informations, consultez Types d'environnement.

   e. Sélectionnez Suivant.

   Pour plus d'informations sur les AWS options qui ne sont pas utilisées dans cet exemple, consultez les pages suivantes :

   - Pour Use custom AMI (Utiliser une AMI personnalisée), veuillez consulter Utilisation d'une image machine Amazon (AMI) personnalisée dans votre environnement Elastic Beanstalk.
   - Si vous ne sélectionnez pas Single instance environment (Environnement à une seule instance), vous devez choisir un type d'équilibrage de charge. Pour plus d'informations, consultez Équilibreur de charge pour votre environnement Elastic Beanstalk.
   - Elastic Beanstalk utilise la configuration par défaut Amazon VPC (Amazon Virtual Private Cloud) si vous n'avez pas choisi Use non-default VPC (Utiliser un VPC autre que le VPC

par défaut). Pour plus d'informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon VPC.

- Le choix de l'option Activer les déploiements mobiles divise un déploiement en lots afin d'éviter les temps d'arrêt potentiels pendant les déploiements. Pour de plus amples informations, veuillez consulter Déploiement d'applications dans des environnements Elastic Beanstalk.
- L'option Relational Database Access (Accès de base de données relationnelle) vous permet de connecter votre environnement Elastic Beanstalk à une base de données Amazon RDS précédemment créée avec des groupes de sécurité de base de données Amazon RDS. Pour de plus amples informations, veuillez consulter Contrôle d'accès par groupe de sécurité dans le Guide de l'utilisateur Amazon RDS.

6. Sélectionnez Next (Suivant) dans la boîte de dialogue Permissions (Autorisations).
7. Sélectionnez Next (Suivant) dans la boîte de dialogue Applications Options (Options d'applications).
8. Passez en revue vos options de déploiement. Après avoir vérifié que vos paramètres sont corrects, sélectionnez Déployer.

Votre application Web ASP.NET Core est exportée en tant que fichier de déploiement Web. Votre fichier est ensuite chargé dans Amazon S3 et enregistré en tant que nouvelle version d'application auprès d'Elastic Beanstalk. La fonctionnalité de déploiement Elastic Beanstalk surveille votre environnement jusqu'à ce qu'il devienne disponible avec le code nouvellement déployé. L'information Status (Statut) de votre environnement s'affiche sur l'onglet Env:<nom de l'environnement>. Une fois que le statut devient Environment is healthy (L'environnement est sain), sélectionnez l'adresse URL pour lancer l'application web.

Résiliation d'un environnement

Pour éviter d'encourir des frais pour les AWS ressources non utilisées, vous pouvez utiliser le pour mettre fin AWS Toolkit for Visual Studio à un environnement en cours d'exécution.

> **Note**
>
> Vous pouvez toujours lancer un nouvel environnement en utilisant la même version ultérieurement.

Pour résilier un environnement

1. Développez le nœud Elastic Beanstalk et le nœud de l'application. Dans AWS Explorer (Explorateur AWS), ouvrez le menu contextuel (clic droit) de votre environnement d'application et sélectionnez Terminate Environment (Résilier l'environnement).
2. Lorsque vous y êtes invité, sélectionnez Oui (Yes) afin de confirmer que vous souhaitez résilier l'environnement. Elastic Beanstalk met quelques minutes à arrêter les ressources en cours d'exécution dans l' AWS environnement.

L'information relative à Status (Statut) de votre environnement sous l'onglet Env:<nom de l'environnement> deviendra Terminating (Résiliation en cours), puis Terminated (Résilié).

> **Note**
>
> Lorsque vous résiliez votre environnement, le CNAME associé à l'environnement résilié devient disponible pour que tout le monde puisse l'utiliser.

Gestion de vos environnements d'application Elastic Beanstalk

Cette section décrit les paramètres de service spécifiques que vous pouvez modifier dans AWS Toolkit for Visual Studio dans le cadre de la configuration d'environnement de votre application. Avec le AWS Toolkit for Visual Studio et la console de AWS gestion, vous pouvez modifier le provisionnement et la configuration des AWS ressources utilisées par vos environnements d'applications. Pour plus d'informations sur la gestion de vos environnements d'applications à l'aide de la console AWS de gestion, consultezGestion des environnements Elastic Beanstalk.

Modification des paramètres de configuration de l'environnement

Lorsque vous déployez votre application, Elastic Beanstalk configure AWS plusieurs services de cloud computing connectés. Vous pouvez contrôler la façon dont ces services individuels sont configurés à l'aide du AWS Toolkit for Visual Studio.

Pour modifier les paramètres d'environnement d'une application

1. Dans Visual Studio, dans le menu File (Fichier), choisissez AWS Explorer (Explorateur AWS).

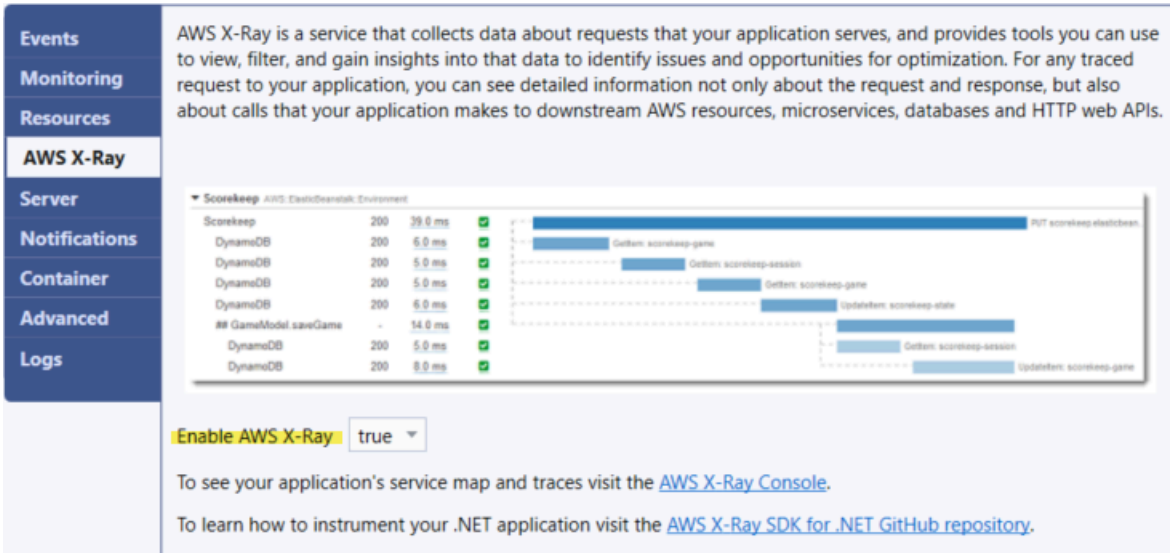
2. Développez le nœud Elastic Beanstalk et le nœud de votre application. Ouvrez le menu contextuel (clic droit) de votre environnement d'application et sélectionnez View Status (Afficher le statut).

   Vous pouvez à présent configurer des paramètres pour les éléments suivants :

   - AWS X-Ray
   - Serveur
   - Équilibreur de charge (s'applique uniquement aux environnements à plusieurs instances)
   - Auto Scaling (s'applique uniquement aux environnements à plusieurs instances)
   - Notifications
   - Conteneur
   - Options de configuration avancées

Configuration de AWS X-Ray à l'aide du AWS kit d'outils pour Visual Studio

AWS X-Ray fournit des fonctionnalités de suivi des demandes, de collecte d'exceptions et de profilage. Le panneau AWS X-Ray vous permet d'activer ou de désactiver X-Ray pour votre application. Pour plus d'informations sur X-Ray, veuillez consulter le Guide du développeur AWS X-Ray.

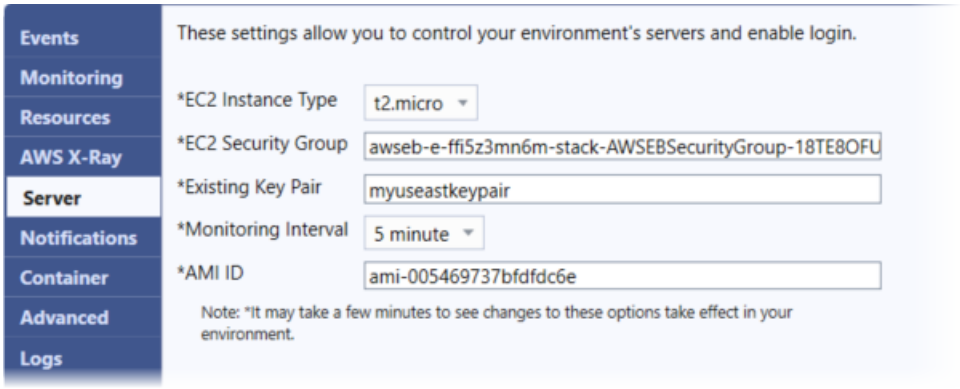


Configuration d' EC2 instances à l'aide du AWS kit d'outils pour Visual Studio

Vous pouvez utiliser Amazon Elastic Compute Cloud (Amazon EC2) pour lancer et gérer des instances de serveur dans les centres de données d'Amazon. Vous pouvez utiliser les instances de

EC2 serveur Amazon à tout moment, aussi longtemps que vous le souhaitez et à toutes fins légales. Les instances sont disponibles dans différentes tailles et configurations. Pour plus d'informations, consultez Amazon EC2.

Vous pouvez modifier la configuration de votre EC2 instance Amazon à l'aide de l'onglet Server situé dans l'onglet Environnement de votre application dans le AWS Toolkit for Visual Studio.



Types d' EC2 instances Amazon

Instance type (Type d'instance) affiche les types d'instance disponibles pour votre application Elastic Beanstalk. Changez le type d'instance pour sélectionner un serveur dont les caractéristiques (y compris la taille de la mémoire et la puissance d'UC) sont les mieux adaptées à votre application. Par exemple, les applications exécutant des opérations intensives et de longue durée peuvent nécessiter plus de puissance de calcul et de mémoire.

Pour plus d'informations sur les types d' EC2 instances Amazon disponibles pour votre application Elastic Beanstalk, consultez la section Types d'instances du manuel Amazon Elastic Compute Cloud User Guide.

Groupes EC2 de sécurité Amazon

Vous pouvez contrôler l'accès à votre application Elastic Beanstalk à l'aide EC2 d'un groupe de sécurité Amazon. Un groupe de sécurité définit les règles de pare-feu de vos instances. Ces règles déterminent le trafic réseau entrant qui doit être acheminé vers votre instance. Tout autre trafic entrant est éliminé. Vous pouvez modifier les règles pour un groupe à la fois. Les nouvelles règles sont appliquées automatiquement pour toutes les instances en cours d'exécution et les instances lancées par la suite.

Vous pouvez spécifier quels groupes de EC2 sécurité Amazon contrôlent l'accès à votre application Elastic Beanstalk. Pour ce faire, entrez les noms de groupes de EC2 sécurité Amazon spécifiques (en séparant plusieurs groupes de sécurité par des virgules) dans la zone de texte EC2 Security

Groups. Pour ce faire, vous pouvez utiliser la console AWS de gestion ou le AWS Toolkit for Visual Studio.

Pour créer un groupe de sécurité à l'aide du AWS kit d'outils pour Visual Studio

1. Dans Visual Studio, dans AWS Explorer, développez le EC2 nœud Amazon, puis sélectionnez Security Groups.
2. Cliquez sur Create Security Group (Créer un groupe de sécurité) et entrez un nom et une description pour votre groupe de sécurité.
3. Sélectionnez OK.

Pour plus d'informations sur les groupes EC2 de sécurité Amazon, consultez la section Utilisation des groupes de sécurité dans le guide de l'utilisateur d'Amazon Elastic Compute Cloud.

Paires EC2 de clés Amazon

Vous pouvez vous connecter en toute sécurité aux EC2 instances Amazon mises en service pour votre application Elastic Beanstalk à l'aide d'une paire de clés Amazon. EC2

> **Important**
>
> Vous devez créer une paire de EC2 clés Amazon et configurer vos EC2 instances Amazon mises en service par Elastic Beanstalk pour pouvoir accéder à ces instances. Vous pouvez créer votre paire de clés à l'aide de l' AWS assistant Publish to du AWS Toolkit for Visual Studio lorsque vous déployez votre application sur Elastic Beanstalk. Si vous souhaitez créer des paires de clés supplémentaires à l'aide de la boîte à outils, procédez comme suit. Vous pouvez également configurer vos paires de EC2 clés Amazon à l'aide de la console AWS de gestion. Pour obtenir des instructions sur la création d'une paire de clés pour Amazon EC2, consultez le guide de démarrage Amazon Elastic Compute Cloud.

La zone de texte Existing Key Pair vous permet de spécifier le nom d'une paire de EC2 clés Amazon que vous pouvez utiliser pour vous connecter en toute sécurité aux EC2 instances Amazon qui exécutent votre application Elastic Beanstalk.

Pour spécifier le nom d'une paire de EC2 clés Amazon

1. Développez le EC2 nœud Amazon et sélectionnez Key Pairs.

2. Sélectionnez Create Key Pair (Créer une paire de clés) et saisissez le nom de la paire de clés.
3. Sélectionnez OK.

Pour plus d'informations sur les paires de EC2 clés Amazon, consultez la section Utilisation des EC2 informations d'identification Amazon dans le guide de l'utilisateur d'Amazon Elastic Compute Cloud. Pour plus d'informations sur la connexion aux EC2 instances Amazon, consultez

Intervalle de surveillance

Par défaut, seules les CloudWatch métriques Amazon de base sont activées. Elles renvoient des données toutes les cinq minutes. Vous pouvez activer des CloudWatch mesures plus détaillées d'une minute en sélectionnant 1 minute pour l'intervalle de surveillance dans la section Serveur de l'onglet Configuration de votre environnement dans le. AWS Toolkit for Eclipse

> **Note**
>
> Des frais CloudWatch de service Amazon peuvent s'appliquer pour les mesures d'intervalle d'une minute. Consultez Amazon CloudWatch pour plus d'informations.

ID d'AMI personnalisé

Vous pouvez remplacer l'AMI par défaut utilisée pour vos EC2 instances Amazon par votre propre AMI personnalisée en saisissant l'identifiant de votre AMI personnalisée dans le champ ID d'AMI personnalisé de la section Serveur de l'onglet Configuration de votre environnement dans le AWS Toolkit for Eclipse.

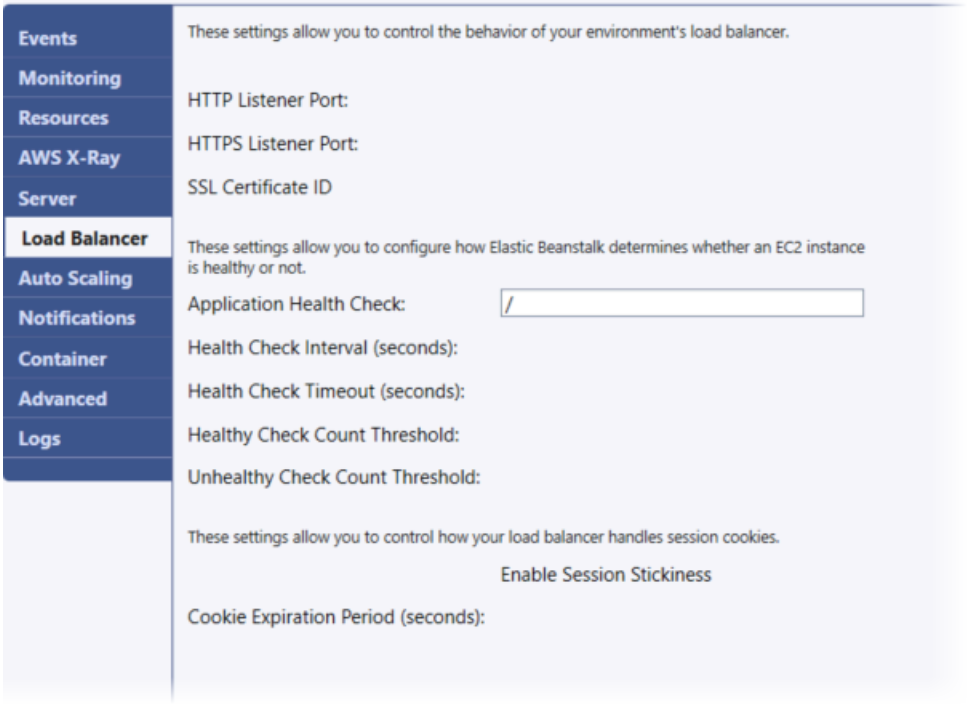
> **Important**
>
> L'utilisation de votre propre image AMI est une tâche avancée qui doit être effectuée avec soin. Si vous avez besoin d'une AMI personnalisée, nous vous recommandons de démarrer par l'AMI Elastic Beanstalk par défaut, puis de la modifier. Pour être considérées comme saines, Elastic Beanstalk EC2 attend des instances Amazon qu'elles répondent à un ensemble d'exigences, notamment qu'elles soient dotées d'un gestionnaire d'hôte actif. Si ces conditions ne sont pas satisfaites, il se peut que votre environnement ne fonctionne pas correctement.

Configuration d'Elastic Load Balancing à l'aide du AWS kit d'outils pour Visual Studio

Elastic Load Balancing est un service d'Amazon Web Services qui vous aide à améliorer la disponibilité et l'évolutivité de votre application. Ce service vous permet de répartir facilement les charges d'applications entre deux ou plusieurs EC2 instances Amazon. Elastic Load Balancing améliore la disponibilité en fournissant une redondance supplémentaire et prend en charge la croissance du trafic pour votre application.

Avec Elastic Load Balancing, vous pouvez répartir et équilibrer automatiquement le trafic d'applications entrant entre toutes vos instances en cours d'exécution. Vous pouvez également ajouter facilement de nouvelles instances lorsque l'augmentation de la capacité de votre application est requise.

Elastic Beanstalk fournit automatiquement Elastic Load Balancing lorsque vous déployez une application. Vous pouvez modifier la configuration de l'instance Amazon de l'environnement Elastic Beanstalk à l'aide EC2 de l'onglet Load Balancer situé dans l'onglet environnement de votre application dans Toolkit AWS for Visual Studio.



Les sections suivantes décrivent les paramètres Elastic Load Balancing que vous pouvez configurer pour votre application.

Ports

L'équilibreur de charge configuré pour traiter les demandes relatives à votre application Elastic Beanstalk envoie des demandes aux instances EC2 Amazon qui exécutent votre application. L'équilibreur de charge provisionné peut écouter les demandes sur les ports HTTP et HTTPS et acheminer les demandes vers les EC2 instances Amazon de votre AWS Elastic Beanstalk application. Par défaut, l'équilibreur de charge gère les demandes sur le port HTTP. Pour que cela fonctionne, au moins un des ports (HTTP ou HTTPS) doit être activé.

> **Important**
>
> Assurez-vous que le port que vous avez spécifié n'est pas verrouillé ; sinon, vous ne pourrez pas vous connecter à votre application Elastic Beanstalk.

Contrôle du port HTTP

Pour désactiver le port HTTP, sélectionnez OFF (désactivé) pour HTTP Listener Port (Port d'écoute HTTP). Pour activer le port HTTP, vous sélectionnez un port HTTP (par exemple, 80) dans la liste.

> **Note**
>
> Pour accéder à votre environnement à l'aide d'un port autre que le port 80, par exemple le port 8080, vous pouvez ajouter un écouteur à l'équilibreur de charge existant et configurer le nouvel écouteur de sorte qu'il écoute sur ce port.
> Par exemple, en utilisant les équilibreurs de charge AWS CLI pour Classic, tapez la commande suivante en la *LOAD_BALANCER_NAME* remplaçant par le nom de votre équilibreur de charge pour Elastic Beanstalk.
>
> ```
> aws elb create-load-balancer-listeners --load-balancer-name LOAD_BALANCER_NAME
>  --listeners "Protocol=HTTP, LoadBalancerPort=8080, InstanceProtocol=HTTP,
>  InstancePort=80"
> ```

Par exemple, à l'aide de AWS CLI for Application Load Balancers, tapez la commande suivante, en la *LOAD_BALANCER_ARN* remplaçant par l'ARN de votre équilibreur de charge pour Elastic Beanstalk.

```
aws elbv2 create-listener --load-balancer-arn LOAD_BALANCER_ARN --protocol HTTP
 --port 8080
```

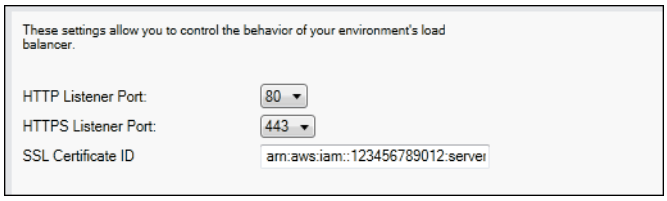
Si vous souhaitez que Elastic Beanstalk surveille votre environnement, ne supprimez pas l'écouteur sur le port 80.

Contrôle du port HTTPS

Elastic Load Balancing prend en charge le HTTPS/TLS protocole permettant de chiffrer le trafic pour les connexions des clients à l'équilibreur de charge. Les connexions entre l'équilibreur de charge et les EC2 instances utilisent le chiffrement en texte brut. Par défaut, le port HTTPS est désactivé.

Pour activer le port HTTPS

1. Créez un nouveau certificat à l'aide de AWS Certificate Manager (ACM) ou téléchargez un certificat et une clé vers Gestion des identités et des accès AWS (IAM). Pour plus d'informations sur une demande de certificat ACM, consultez Request a Certificate (Demande de certificat) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur AWS Certificate Manager). Pour plus d'informations sur l'importation de certificats tiers dans ACM, consultez Importing Certificates (Importation de certificats) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur AWS Certificate Manager). Si ACM n'est pas disponible dans votre région, utilisez Gestion des identités et des accès AWS (IAM) pour télécharger un certificat tiers. Les services ACM et IAM stockeront le certificat et fourniront un Amazon Resource Name (ARN) pour le certificat SSL. Pour de plus amples informations sur la création et le chargement des certificats dans IAM, veuillez consulter Utilisation des certificats de serveur dans le Guide de l'utilisateur IAM.
2. Spécifiez le port HTTPS en sélectionnant un port pour HTTPS Listener Port (Port d'écoute HTTPS).



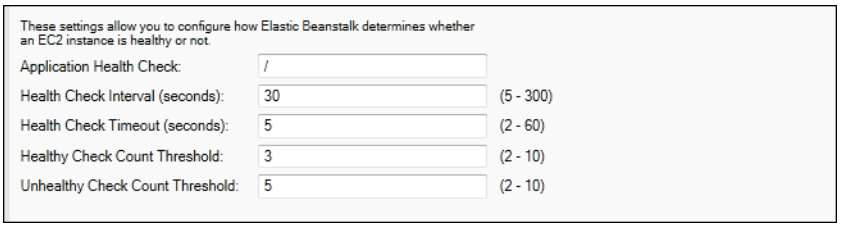
3. Pour SSL Certificate ID (ID du certificat SSL), saisissez l'ARN (Amazon Resources Name) de votre certificat SSL. Par exemple, **`arn:aws:iam::123456789012:server-certificate/abc/certs/build`** ou **`arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678`**. Utilisez le certificat SSL que vous avez créé ou chargé à l'étape 1.

Pour désactiver le port HTTPS, sélectionnez OFF (désactivé) pour HTTPS Listener Port (Port d'écoute HTTPS).

Vérifications de l'état

La définition de la vérification de l'état inclut une URL à interroger pour l'intégrité de l'instance. Par défaut, Elastic Beanstalk utilise TCP:80 pour les conteneurs non hérités et HTTP:80 pour les conteneurs hérités. Vous pouvez remplacer l'URL par défaut par une URL qui correspond à une ressource existante dans votre application (par exemple, `/myapp/default.aspx`) en entrant celle-ci dans la zone URL de vérification de l'état de l'application. Si vous remplacez l'URL par défaut, Elastic Beanstalk utilise HTTP pour interroger la ressource. Pour vérifier si vous utilisez un type de conteneur hérité, consultez the section called “Pourquoi certaines versions de plate-forme sont-elles marquées héritées ?”

Vous pouvez contrôler les paramètres du bilan de santé à l'aide de la section EC2 Instance Health Check du panneau d'équilibrage de charge.



La définition de la vérification de l'état inclut une URL à interroger pour l'intégrité de l'instance. Remplacez l'URL par défaut par une URL qui correspond à une ressource existante dans votre application (par exemple, `/myapp/index.jsp`) en entrant celle-ci dans la zone URL de vérification de l'état de l'application.

La liste suivante décrit les paramètres de vérification de l'état que vous pouvez définir pour votre application.

- Pour Health Check Interval (secondes), entrez le nombre de secondes pendant lesquelles Elastic Load Balancing attend entre les tests de santé pour les EC2 instances Amazon de votre application.

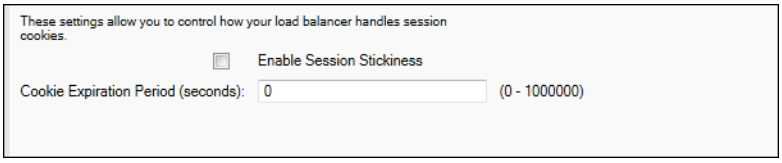
- Pour Délai de vérification de l'état (secondes), spécifiez le nombre de secondes d'attente d'une réponse pour Elastic Load Balancing avant de considérer que l'instance ne répond pas.
- Pour Seuil du nombre de vérifications de l'état saines et Seuil du nombre de vérifications de l'état non saines, spécifiez le nombre d'analyses d'URL consécutives réussies et non réussies avant qu'Elastic Load Balancing ne modifie l'état de l'instance. Par exemple, si vous spécifiez **5** pour Seuil du nombre de vérifications de l'état non saines, l'URL doit renvoyer un message d'erreur ou une expiration du délai cinq fois de suite avant qu'Elastic Load Balancing considère que la vérification de l'état est un échec.

Séances

Par défaut, un équilibreur de charge achemine chaque demande de façon indépendante à l'instance de serveur ayant la plus petite charge. Par comparaison, une session permanente lie la séance d'un utilisateur à une instance de serveur spécifique afin que toutes les demandes provenant de l'utilisateur pendant la séance soient envoyées à la même instance de serveur.

Elastic Beanstalk utilise des cookies HTTP générés par l'équilibreur de charge lorsque des sessions permanentes sont activées pour une application. L'équilibreur de charge utilise un cookie spécial généré par l'équilibreur de charge pour suivre l'instance d'application pour chaque demande. Lorsque l'équilibreur de charge reçoit une demande, il vérifie d'abord si ce cookie est présent dans la demande. Si elle est présente, la demande est envoyée à l'instance d'application spécifiée dans le cookie. S'il n'y a pas de cookie, l'équilibreur de charge choisit une instance d'application à partir de l'algorithme d'équilibrage de charge existant. Un cookie est inséré dans la réponse pour lier les demandes suivantes provenant du même utilisateur à cette instance d'application. La configuration de la politique définit l'expiration d'un cookie, ce qui établit la durée de validité de chaque cookie.

Vous pouvez utiliser la section Sessions de l'onglet Équilibreur de charge pour spécifier si l'équilibreur de charge de votre application autorise la permanence de session.



Pour de plus amples informations sur Elastic Load Balancing, veuillez consulter le Guide du développeur Elastic Load Balancing.

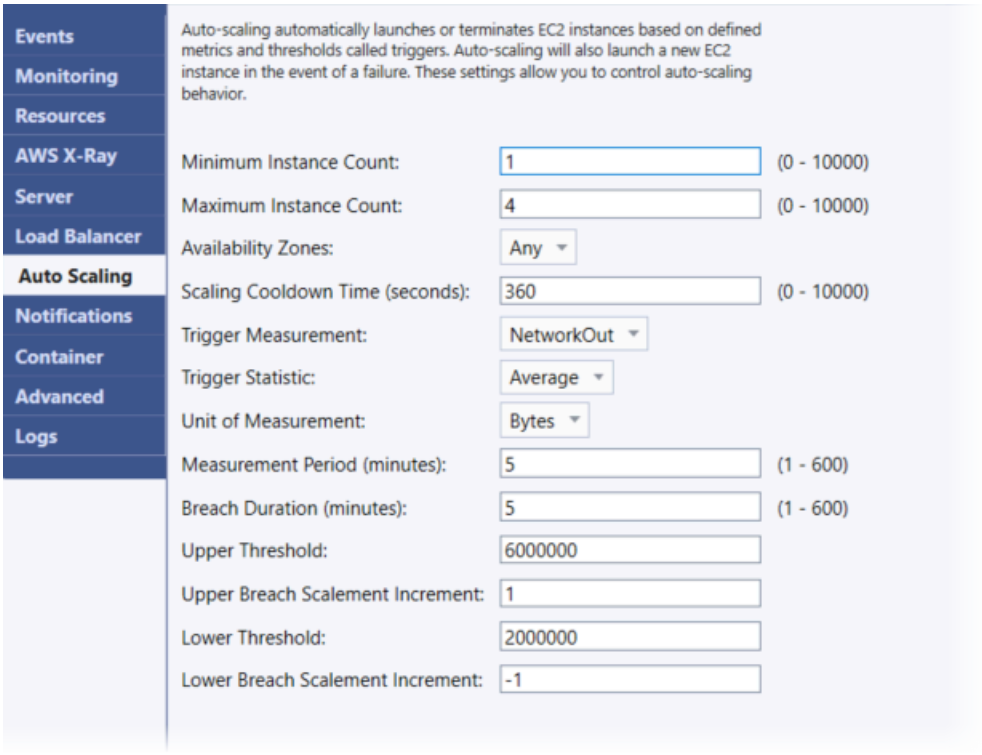
Configuration d'Auto Scaling à l'aide du AWS kit d'outils pour Visual Studio

Amazon EC2 Auto Scaling est un service Web Amazon conçu pour lancer ou arrêter automatiquement des EC2 instances Amazon en fonction de déclencheurs définis par l'utilisateur. Vous pouvez configurer des groupes Auto Scaling et y associer des déclencheurs afin de mettre à l'échelle automatiquement les ressources de calcul en fonction de métriques comme l'utilisation de la bande passante ou l'utilisation de l'UC. Amazon EC2 Auto Scaling travaille avec Amazon CloudWatch pour récupérer les métriques des instances de serveur exécutant votre application.

Amazon EC2 Auto Scaling vous permet de prendre un groupe d' EC2 instances Amazon et de définir différents paramètres pour que le nombre de ce groupe augmente ou diminue automatiquement. Amazon EC2 Auto Scaling peut ajouter ou supprimer EC2 des instances Amazon de ce groupe afin de vous aider à gérer facilement les modifications du trafic vers votre application.

Amazon EC2 Auto Scaling surveille également l'état de santé de chaque EC2 instance Amazon lancée. Si une instance se termine de façon inattendue, Amazon EC2 Auto Scaling détecte la résiliation et lance une instance de remplacement. Cette fonctionnalité vous permet de maintenir automatiquement un nombre fixe et souhaité d' EC2 instances Amazon.

Elastic Beanstalk fournit EC2 Amazon Auto Scaling pour votre application. Vous pouvez modifier la configuration de l'instance Amazon de l'environnement Elastic Beanstalk à l'aide EC2 de l'onglet Auto Scaling situé dans l' AWS onglet Environnement de votre application dans le Toolkit for Visual Studio.

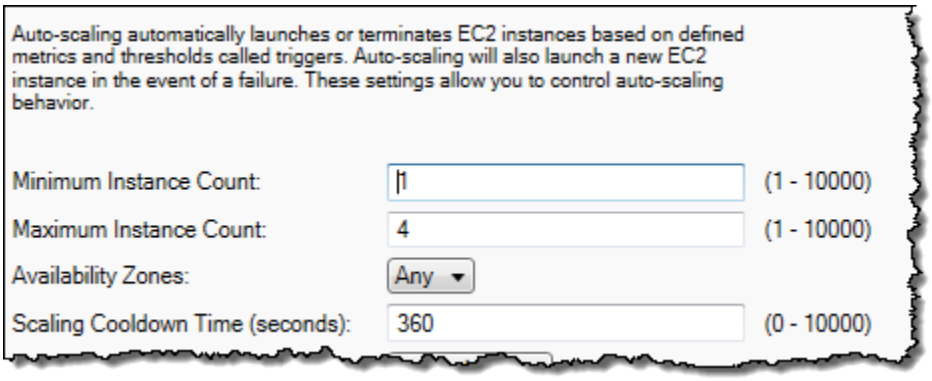


La section suivante explique comment configurer les paramètres Auto Scaling pour votre application.

Lancement de la configuration

Vous pouvez modifier la configuration de lancement pour contrôler la manière dont votre application Elastic Beanstalk approvisionne les ressources EC2 Amazon Auto Scaling.

Les zones Minimum Instance Count (Nombre minimum d'instances) et Maximum Instance Count (Nombre maximum d'instances) vous permettent de spécifier les tailles maximale et minimale du groupe Auto Scaling utilisé par votre application Elastic Beanstalk.



> **Note**
>
> Pour conserver un nombre fixe d' EC2 instances Amazon, définissez le nombre minimum d'instances et le nombre maximal d'instances sur la même valeur.

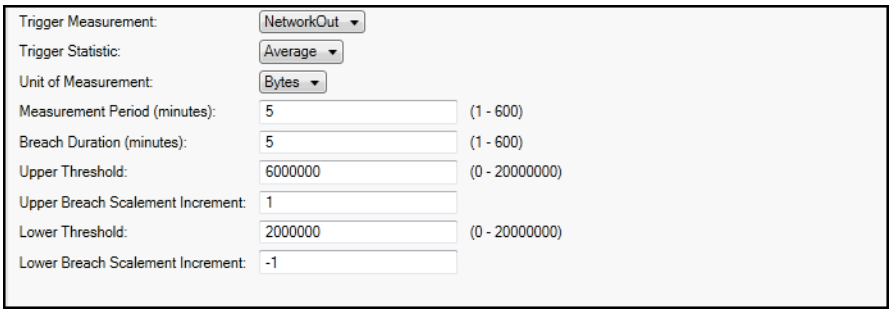
La case Zones de disponibilité vous permet de spécifier le nombre de zones de disponibilité dans lesquelles vous souhaitez que vos EC2 instances Amazon se trouvent. Il est important de définir ce nombre si vous souhaitez créer des applications à tolérance de panne. Si une zone de disponibilité est défaillante, l'exécution de vos instances se poursuivra dans vos autres zones de disponibilité.

> **Note**
>
> Il est actuellement impossible de spécifier la zone de disponibilité dans laquelle se situera votre instance.

Triggers

Un déclencheur est un mécanisme Amazon EC2 Auto Scaling que vous configurez pour indiquer au système quand vous souhaitez augmenter (scaler) ou diminuer (augmenter) le nombre d'instances. Vous pouvez configurer des déclencheurs pour qu'ils se déclenchent sur n'importe quelle métrique publiée sur Amazon CloudWatch (par exemple, l'utilisation du processeur) et déterminer si les conditions que vous avez spécifiées sont remplies. Lorsque les seuils inférieurs ou supérieurs des conditions que vous avez spécifiées pour la métrique ont été dépassés pendant la période spécifiée, le déclencheur lance un processus de longue durée que nous appelons une activité de dimensionnement.

Vous pouvez définir un déclencheur de dimensionnement pour votre application Elastic Beanstalk à l'aide de Toolkit for AWS Visual Studio.



Les déclencheurs Amazon EC2 Auto Scaling fonctionnent en surveillant une CloudWatch métrique Amazon spécifique d'une instance donnée. Les métriques incluent l'utilisation de l'UC, le trafic réseau et l'activité du disque. Utilisez le paramètre Mesure du déclencheur pour sélectionner une métrique associée à votre déclencheur.

La liste suivante décrit les paramètres de déclenchement que vous pouvez configurer à l'aide de la console AWS de gestion.

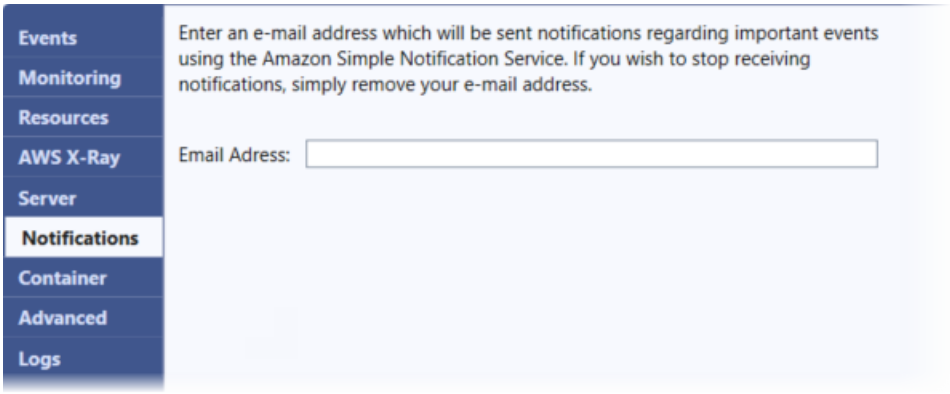
- Vous pouvez spécifier les statistiques que le déclencheur devrait utiliser. Vous pouvez sélectionner Minimum, Maximum, Sum (Somme) ou Average (Moyenne) pour Statistique du déclencheur.
- Pour Unité de mesure, spécifiez l'unité de mesure du déclencheur.
- La valeur figurant dans le champ Période de mesure indique à quelle fréquence Amazon CloudWatch mesure les statistiques pour votre déclencheur. La valeur Breach duration correspond à la durée pendant laquelle une métrique peut aller au-delà de sa limite définie (telle que spécifiée dans Upper threshold et Lower threshold) avant l'activation du déclencheur.

- Pour l'incrément d'échelle de violation supérieur et l'incrément d'échelle de violation inférieur, spécifiez le nombre d' EC2instances Amazon à ajouter ou à supprimer lors de l'exécution d'une activité de dimensionnement.

Pour plus d'informations sur Amazon EC2 Auto Scaling, consultez la section Amazon EC2 Auto Scaling de la documentation Amazon Elastic Compute Cloud.

Configuration des notifications à l'aide du AWS kit d'outils pour Visual Studio

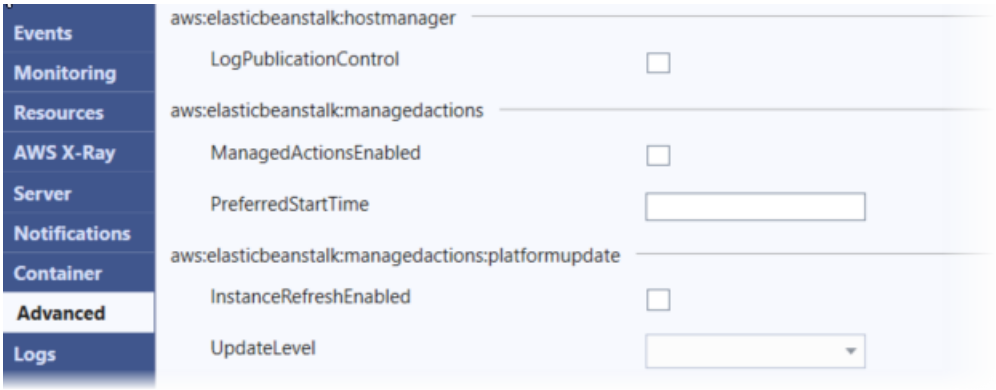
Elastic Beanstalk utilise Amazon Simple Notification Service (Amazon SNS) pour vous informer des événements importants qui concernent votre application. Pour activer les notifications Amazon SNS, entrez votre adresse e-mail dans la zone Email Address (Adresse e-mail). Pour désactiver ces notifications, supprimez votre adresse e-mail de la zone.



Configuration d'options d'environnement supplémentaires à l'aide de la AWS boîte à outils pour Visual Studio

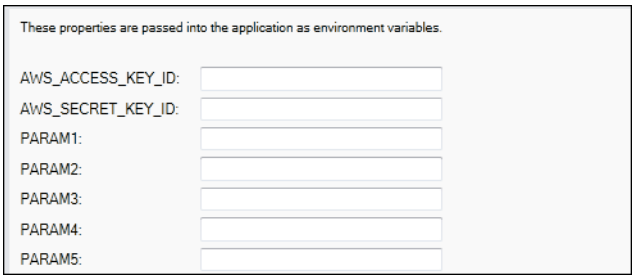
Elastic Beanstalk définit un grand nombre d'options de configuration que vous pouvez utiliser pour configurer le comportement de votre environnement et les ressources qu'il contient. Les options de configuration sont organisées en espaces de noms comme `aws:autoscaling:asg`. Chaque espace de noms définit les options pour le groupe Auto Scaling d'un environnement. Le panneau Advanced (Avancé) répertorie les espaces de noms des options de configuration dans l'ordre alphabétique. Vous pouvez le mettre à jour après la création de l'environnement.

Pour obtenir une liste complète des espaces de noms et des options, y compris les valeurs par défaut et celles prises en charge pour chacun, veuillez consulter Options générales pour tous les environnements et Historique de la plateforme .NET Core sous Linux.



Configuration des conteneurs .NET Core à l'aide du AWS kit d'outils pour Visual Studio

Le panneau Container (Conteneur) vous permet de spécifier des variables d'environnement que vous pouvez lire à partir du code de votre application.



Surveillance de l'intégrité d'une application

Cette rubrique explique comment surveiller l'état du site Web de votre application. Il est important de savoir que votre site de production est disponible et qu'il répond aux demandes. Elastic Beanstalk fournit des fonctionnalités pour vous aider à surveiller la réactivité de votre application. Il surveille les statistiques relatives à votre application et vous avertit lorsque les seuils sont dépassés.

Pour de plus amples informations sur la surveillance de l'état fournie par Elastic Beanstalk, veuillez consulter Création de rapports d'intégrité de base.

Vous pouvez accéder aux informations opérationnelles relatives à votre application en utilisant le AWS Toolkit for Visual Studio ou la console AWS de gestion.

La boîte à outils affiche le statut de votre environnement et l'état de votre application dans le champ Status (Statut).

Pour surveiller l'intégrité de l'application

1. Dans le AWS Toolkit for Visual Studio, dans AWS Explorer, développez le nœud Elastic Beanstalk, puis étendez le nœud de votre application.
2. Ouvrez le menu contextuel (clic droit) de votre environnement d'application et sélectionnez View Status (Afficher le statut).
3. Dans l'onglet de l'environnement de votre application, sélectionnez Monitoring (Surveillance).

   Le panneau Monitoring (Surveillance) comprend un ensemble de graphiques illustrant l'utilisation des ressources pour votre environnement d'application spécifique.

   > **Note**
   >
   > Par défaut, la plage de temps est définie sur la dernière heure. Pour modifier ce paramètre, dans la liste Time Range (Plage de temps), sélectionnez une plage de temps différente.

Vous pouvez utiliser le AWS Toolkit for Visual Studio ou la console AWS de gestion pour afficher les événements associés à votre application.

Pour afficher des événements d'application

1. Dans le AWS Toolkit for Visual Studio, dans AWS Explorer, développez le nœud Elastic Beanstalk et le nœud de votre application.
2. Ouvrez le menu contextuel (clic droit) de votre environnement d'application et sélectionnez View Status (Afficher le statut).
3. Dans l'onglet de l'environnement de votre application, sélectionnez Events (Événements).

Migration de .NET sur la plateforme Windows Server vers la plateforme .NET Core sur Linux sur Elastic Beanstalk

Vous pouvez migrer des applications qui s'exécutent sur les plateformes .NET sous Windows Server vers les plateformes .NET Core sous Linux. Vous trouverez ci-après quelques considérations relatives à la migration depuis Windows vers les plateformes Linux.

Considérations relatives à la migration vers la plateforme .NET Core sous Linux

Area	Modifications et informations
Configuration de l'application	Sur les plateformes Windows, vous utilisez un manifeste de déploiement pour spécifier les applications qui s'exécutent dans votre environnement. Les plateformes .NET Core sous Linux utilisent un fichier Procfile pour spécifier les applications qui s'exécutent sur les instances de votre environnement. Pour plus de détails sur le regroupement d'applications, veuillez consulter the section called “Regroupement d'applications”.
Serveur proxy	Sur les plateformes Windows, vous utilisez IIS en tant que serveur proxy de votre application. Les plateformes .NET Core sous Linux incluent nginx en tant que proxy inverse par défaut. Vous pouvez choisir de n'utiliser aucun serveur proxy et d'utiliser Kestrel en tant que serveur Web de votre application. Pour en savoir plus, veuillez consulter la section the section called “Serveur proxy”.
Routage	Sur les plateformes Windows, vous utilisez IIS dans votre code d'application et incluez un manifeste de déploiement pour configurer le chemin d'accès IIS. Pour la plateforme .NET Core sous Linux, vous utilisez le routage ASP .NET Core dans votre code d'application et mettez à jour la configuration nginx de votre environnement. Pour en savoir plus, veuillez consulter la section the section called “Serveur proxy”.
Journaux	Les plateformes Linux et Windows diffusent différents journaux. Pour en savoir plus, consultez the section called “Comment Elastic Beanstalk configure les journaux CloudWatch”.

Déploiement d'applications Go avec Elastic Beanstalk

Ce chapitre fournit des instructions pour configurer et déployer votre application Web Go sur AWS Elastic Beanstalk. Elastic Beanstalk facilite le déploiement, la gestion et le dimensionnement de vos applications Web Go à l'aide d'Amazon Web Services.

Vous pouvez déployer votre application en quelques minutes à l'aide de l'interface de ligne de commande Elastic Beanstalk (EB CLI) ou de la console Elastic Beanstalk. Après avoir déployé votre application Elastic Beanstalk, vous pouvez continuer à utiliser l'EB CLI pour gérer votre application et votre environnement, ou vous pouvez utiliser la console Elastic Beanstalk, ou le. AWS CLI APIs

Suivez les step-by-step instructions QuickStart pour Go pour créer et déployer une application Web Hello World Go avec l'interface de ligne de commande EB.

Rubriques

- QuickStart: Déployer une application Go sur Elastic Beanstalk
- Configuration de votre environnement de développement Go pour Elastic Beanstalk
- Utilisation de la plateforme Go Elastic Beanstalk

QuickStart: Déployer une application Go sur Elastic Beanstalk

Ce QuickStart didacticiel explique le processus de création d'une application Go et de son déploiement dans un AWS Elastic Beanstalk environnement.

> ⚠ Non destiné à une utilisation en production
>
> Les exemples sont destinés uniquement à des fins de démonstration. N'utilisez pas d'exemples d'applications en production.

Sections

- Votre AWS compte
- Conditions préalables
- Étape 1 : créer une application Go

- Étape 2 : Déployez votre application Go avec l'interface de ligne de commande EB
- Étape 3 : Exécutez votre application sur Elastic Beanstalk
- Étape 4 : nettoyer
- AWS ressources pour votre application
- Étapes suivantes
- Déployez avec la console Elastic Beanstalk

Votre AWS compte

Si vous n'êtes pas encore AWS client, vous devez créer un AWS compte. L'inscription vous permet d'accéder à Elastic Beanstalk AWS et aux autres services dont vous avez besoin.

Si vous avez déjà un AWS compte, vous pouvez passer àConditions préalables.

Créez un AWS compte

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez l'https://portal.aws.amazon.com/billing/inscription.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d’inscription, vous recevrez un appel téléphonique ou un SMS et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

   Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l’utilisateur racine a accès à l’ensemble des Services AWS et des ressources de ce compte. La meilleure pratique de sécurité consiste à attribuer un accès administratif à un utilisateur, et à utiliser uniquement l’utilisateur racine pour effectuer les tâches nécessitant un accès utilisateur racine.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. À tout moment, vous pouvez consulter l'activité actuelle de votre compte et gérer votre compte en accédant à https://aws.amazon.com/et en choisissant Mon compte.

Création d’un utilisateur doté d’un accès administratif

Après vous être inscrit à un Compte AWS, sécurisez Utilisateur racine d'un compte AWS AWS IAM Identity Center, activez et créez un utilisateur administratif afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l’aide pour vous connecter en utilisant l’utilisateur racine, consultez Connexion en tant qu’utilisateur racine dans le Guide de l’utilisateur Connexion à AWS .
2. Activez l’authentification multifactorielle (MFA) pour votre utilisateur racine.

   Pour obtenir des instructions, voir Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d’un utilisateur doté d’un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d’ AWS IAM Identity Center dans le Guide de l’utilisateur AWS IAM Identity Center .
2. Dans IAM Identity Center, octroyez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connexion en tant qu’utilisateur doté d’un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l’URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l’utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribution d'un accès à d'autres utilisateurs

1. Dans IAM Identity Center, créez un ensemble d'autorisations qui respecte la bonne pratique consistant à appliquer les autorisations de moindre privilège.

   Pour obtenir des instructions, consultez Création d'un ensemble d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .
2. Attribuez des utilisateurs à un groupe, puis attribuez un accès par authentification unique au groupe.

   Pour obtenir des instructions, consultez Ajout de groupes dans le Guide de l'utilisateur AWS IAM Identity Center .

Conditions préalables

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

INTERFACE DE LIGNE DE COMMANDE (CLI) EB

Ce tutoriel utilise également l'interface de ligne de commande Elastic Beanstalk (CLI EB). Pour de plus amples informations sur l'installation et la configuration de la CLI EB, veuillez consulter Installation de la CLI EB avec un script de configuration (recommandé) et Configuration de l'interface de ligne de commande EB.

Étape 1 : créer une application Go

Créez un répertoire de projet.

```
~$ mkdir eb-go
~$ cd eb-go
```

Créez ensuite une application qui vous allez déployer à l'aide d'Elastic Beanstalk. Nous allons créer un service RESTful Web « Hello World ».

Cet exemple affiche un message d'accueil personnalisé qui varie selon le chemin d'accès utilisé pour accéder au service.

Créez un fichier texte dans le répertoire nommé `application.go` avec le contenu suivant :

Example~/eb-go/application.go

```
package main

import (
 "fmt"
 "net/http"
)

func handler(w http.ResponseWriter, r *http.Request) {
 if r.URL.Path == "/" {
  fmt.Fprintf(w, "Hello World! Append a name to the URL to say hello. For example, use
 %s/Mary to say hello to Mary.", r.Host)
 } else {
  fmt.Fprintf(w, "Hello, %s!", r.URL.Path[1:])
 }
}

func main() {
 http.HandleFunc("/", handler)
 http.ListenAndServe(":5000", nil)
}
```

Étape 2 : Déployez votre application Go avec l'interface de ligne de commande EB

Créez ensuite votre environnement d'applications et déployez votre application configurée avec Elastic Beanstalk.

Création d'un environnement et déploiement de votre application Go

1. Initialisez votre référentiel de la CLI EB avec la commande eb init :

```
~/eb-go$ eb init -p go go-tutorial --region us-east-2
Application go-tutorial has been created.
```

Cette commande crée une application nommée `go-tutorial` et configure votre référentiel local pour créer des environnements avec la dernière version de la plateforme Go.

2. (Facultatif) Exécutez à eb init nouveau pour configurer une paire de clés par défaut afin de pouvoir utiliser SSH pour vous connecter à l' EC2 instance qui exécute votre application.

```
~/eb-go$ eb init
Do you want to set up SSH for your instances?
(y/n): y
Select a keypair.
1) my-keypair
2) [ Create new KeyPair ]
```

Sélectionnez une paire de clés si vous en avez déjà une, ou suivez les invites pour en créer une. Si vous ne voyez pas l'invite ou que vous avez besoin de modifier vos paramètres ultérieurement, exécutez eb init -i.

3. Créez un environnement et déployez-y votre application avec eb create. Elastic Beanstalk crée automatiquement un fichier zip pour votre application et le démarre sur le port 5000.

```
~/eb-go$ eb create go-env
```

Il faut environ cinq minutes à Elastic Beanstalk pour créer votre environnement.

Étape 3 : Exécutez votre application sur Elastic Beanstalk

Lorsque le processus de création de votre environnement est terminé, ouvrez votre site Web aveceb open.

```
~/eb-go$ eb open
```

Félicitations ! Vous avez déployé une application Go avec Elastic Beanstalk ! Celle-ci ouvre une fenêtre de navigation en utilisant le nom de domaine créé pour votre application.

Étape 4 : nettoyer

Vous pouvez mettre fin à votre environnement lorsque vous avez fini d'utiliser votre application. Elastic Beanstalk AWS met fin à toutes les ressources associées à votre environnement.

Pour terminer votre environnement Elastic Beanstalk avec l'EB CLI, exécutez la commande suivante.

```
~/eb-go$ eb terminate
```

AWS ressources pour votre application

Vous venez de créer une application à instance unique. Il s'agit d'un exemple d'application simple avec une seule EC2 instance, il ne nécessite donc pas d'équilibrage de charge ni de dimensionnement automatique. Pour les applications à instance unique, Elastic Beanstalk crée les ressources suivantes : AWS

- EC2 instance — Une machine EC2 virtuelle Amazon configurée pour exécuter des applications Web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble distinct de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse qui traite le trafic web devant votre application web, lui transmet les demandes, traite les ressources statiques et génère des journaux d'accès et d'erreurs.
- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack : Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.

- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*. *region*.elasticbeanstalk.com.

Elastic Beanstalk gère toutes ces ressources. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

Étapes suivantes

Dès que vous disposez d'un environnement exécutant une application, vous pouvez à tout moment déployer une nouvelle version de l'application ou une application totalement différente. Le déploiement d'une nouvelle version d'application est très rapide car il ne nécessite ni le provisionnement ni le redémarrage des instances. EC2 Vous pouvez également explorer votre nouvel environnement à l'aide de la console Elastic Beanstalk. Pour connaître les étapes détaillées, consultez la section Explorez votre environnement dans le chapitre Mise en route de ce guide.

Une fois que vous avez déployé un exemple d'application ou deux, et que vous êtes prêt à développer et exécuter les applications Go en local, consultez Configuration de votre environnement de développement Go pour Elastic Beanstalk.

Déployez avec la console Elastic Beanstalk

Vous pouvez également utiliser la console Elastic Beanstalk pour lancer l'exemple d'application. Pour connaître les étapes détaillées, voir Création d'un exemple d'application dans le chapitre Mise en route de ce guide.

Configuration de votre environnement de développement Go pour Elastic Beanstalk

Cette rubrique fournit des instructions pour configurer un environnement de développement Go afin de tester votre application localement avant de la déployer AWS Elastic Beanstalk. Il fait également référence à des sites Web qui fournissent des instructions d'installation pour des outils utiles.

Installation de Go

Pour exécuter des applications Go localement, installez Go. Si vous n'avez pas besoin d'une version spécifique, procurez-vous la dernière version prise en charge par Elastic Beanstalk. Pour obtenir la liste des versions prises en charge, accédez à Go dans le document Plateformes AWS Elastic Beanstalk .

Téléchargez Go à l'adresse https://golang. org/doc/install.

Installation du AWS SDK pour Go

Si vous devez gérer AWS des ressources depuis votre application, installez le AWS SDK for Go à l'aide de la commande suivante.

```
$ go get github.com/aws/aws-sdk-go
```

Pour plus d'informations, veuillez consulter Kit SDK AWS pour Go.

Utilisation de la plateforme Go Elastic Beanstalk

Cette rubrique explique comment configurer, créer et exécuter vos applications Go sur Elastic Beanstalk.

AWS Elastic Beanstalk prend en charge un certain nombre de branches de plate-forme pour différentes versions du langage de programmation Go. Voir Go dans le document AWS Elastic Beanstalk Plateformes pour une liste complète.

Pour des applications Go simples, il existe deux façons de déployer votre application :

- Fournissez un groupe source avec un fichier source à la racine appelé `application.go` qui contient le package principal pour votre application. Elastic Beanstalk crée le binaire à l'aide de la commande suivante :

  ```
  go build -o bin/application application.go
  ```

  Une fois que l'application est créée, Elastic Beanstalk la démarre sur le port 5000.
- Fournissez un groupe source avec un fichier binaire appelé `application`. Le fichier binaire peut être situé soit à la racine du bundle de fichiers source, soit dans le répertoire `bin/` du bundle de fichiers source. Si vous placez le fichier binaire `application` dans les deux emplacements, Elastic Beanstalk utilise le fichier du répertoire `bin/`.

  Elastic Beanstalk lance cette application sur le port 5000.

Dans les deux cas, grâce à nos branches de plateforme Go prises en charge, vous pouvez également fournir les exigences du module dans un fichier appelé`go.mod`. Pour plus d'informations, consultez Migrer vers des modules Go dans le blog Go.

Pour des applications Go plus complexes, il existe deux façons de déployer votre application :

- Fournissez un bundle source qui inclut vos fichiers sources d'application, avec un Buildfile et un Procfile. Le Buildfile inclut une commande pour générer l'application et le Procfile inclut des instructions pour exécuter l'application.
- Fournissez un groupe source qui inclut vos fichiers binaires d'application, avec un Procfile. Le Procfile inclut des instructions pour exécuter l'application.

La plateforme Go inclut un serveur proxy pour servir les ressources statiques et transférer le trafic vers votre application. Vous pouvez étendre ou remplacer la configuration du serveur proxy par défaut pour les scénarios avancés.

Pour plus d'informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, consultez the section called “Extension des plateformes Linux”.

Configuration de votre environnement Go

Les paramètres de la plateforme Go vous permettent d'affiner le comportement de vos EC2 instances Amazon. Vous pouvez modifier la configuration de l'instance Amazon de l'environnement Elastic Beanstalk à l'aide EC2 de la console Elastic Beanstalk.

Utilisez la console Elastic Beanstalk pour permettre la rotation des journaux sur Amazon S3 et configurer des variables que votre application peut lire à partir de l'environnement.

Pour configurer votre environnement Go dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.

Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Activer la rotation des fichiers journaux vers Amazon S3 : indique si les fichiers journaux des EC2 instances Amazon de votre application sont copiés dans le compartiment Amazon S3 associé à votre application.

Fichiers statiques

Pour améliorer les performances, la section des Fichiers statiques vous permet de configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Pour chaque répertoire, vous définissez le chemin virtuel sur le mappage de répertoires. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Pour en savoir plus sur la configuration des fichiers statiques à l'aide des fichiers de configuration ou de la console Elastic Beanstalk, consultez the section called “Fichiers statiques”.

Propriétés de l'environnement

La section Propriétés de l'environnement vous permet de définir les paramètres de configuration de l'environnement sur EC2 les instances Amazon qui exécutent votre application. Les propriétés de l'environnement sont passées en tant que paires clé-valeur à l'application.

Dans l'environnement Go en cours d'exécution dans Elastic Beanstalk, les variables d'environnement sont accessibles à l'aide de la fonction `os.Getenv`. Par exemple, vous pouvez lire une propriété nommée `API_ENDPOINT` sur une variable avec le code suivant :

```
endpoint := os.Getenv("API_ENDPOINT")
```

Pour plus d'informations, consultez Variables d'environnement et autres paramètres du logiciel.

Espaces de noms de la configuration Go

Vous pouvez utiliser un fichier de configuration pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être spécifiques à la plate-forme ou s'appliquer à toutes les plateformes du service Elastic Beanstalk dans son ensemble. Les options de configuration sont organisées en espaces de noms.

La plate-forme Go ne définit aucun espace de noms spécifique à la plate-forme. Vous pouvez configurer le proxy pour qu'il traite les fichiers statiques à l'aide de l'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles`. Pour plus de détails et un exemple, reportez-vous à la section the section called “Fichiers statiques”.

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d' AWS CLI. Pour plus d’informations, consultez Options de configuration.

Plateforme Go d'AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Go utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans cette section.

> **Remarques**
>
> - Les informations contenues dans cette rubrique s'appliquent uniquement aux branches de plate-forme basées sur l'AMI Amazon Linux (AL1). AL2Les branches de AL2 plate-forme 023/ sont incompatibles avec les versions précédentes de la plateforme Amazon Linux AMI (AL1) et nécessitent des paramètres de configuration différents.
> - Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Espaces de noms de configuration Go — AMI Amazon Linux () AL1

Vous pouvez utiliser un fichier de configuration pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être spécifiques à la plate-forme ou s'appliquer à toutes les plateformes du service Elastic Beanstalk dans son ensemble. Les options de configuration sont organisées en espaces de noms.

> **Note**
>
> Les informations contenues dans cette rubrique s'appliquent uniquement aux branches de plate-forme basées sur l'AMI Amazon Linux (AL1). AL2Les branches de AL2 plate-forme 023/

sont incompatibles avec les versions précédentes de la plateforme Amazon Linux AMI (AL1) et nécessitent des paramètres de configuration différents.

La plateforme Go de l'AMI Amazon Linux prend en charge l'espace de noms de configuration spécifique à la plateforme en plus des espaces de noms pris en charge par toutes les plateformes. L'espace de noms `aws:elasticbeanstalk:container:golang:staticfiles` vous permet de définir des options qui mappe des chemins d'accès sur votre application web vers des dossiers dans le groupe source de votre application incluant le contenu statique.

Par exemple, ce fichier de configuration indique au serveur proxy de servir les fichiers dans le dossier `staticimages` du chemin `/images` :

Example.ebextensions/go-settings.config

```
option_settings:
  aws:elasticbeanstalk:container:golang:staticfiles:
    /html: statichtml
    /images: staticimages
```

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d' AWS CLI. Pour plus d'informations, consultez Options de configuration.

Configuration de commandes de démarrage personnalisées avec un Procfile sur Elastic Beanstalk

Pour spécifier des commandes personnalisées pour démarrer une application Go, incluez un fichier nommé `Procfile` à la racine de votre groupe source.

Pour plus de détails sur l'écriture et l'utilisation d'un`Procfile`, voirBuildfile et Procfile.

Example Procfile

```
web: bin/server
queue_process: bin/queue_processor
foo: bin/fooapp
```

Vous devez appeler l'application principale `web` et la répertorier comme première commande dans votre fichier `Procfile`. Elastic Beanstalk expose la principale application `web` sur l'URL racine de l'environnement ; par exemple, `http://my-go-env.elasticbeanstalk.com`.

Elastic Beanstalk exécute également n'importe quelle application dont le nom n'a pas le préfixe `web_`, mais ces applications ne sont pas disponibles à l'extérieur de votre instance.

Elastic Beanstalk s'attend à ce que les processus s'exécutant à partir du fichier `Procfile` le fassent en continu. Elastic Beanstalk surveille ces applications et redémarre tout processus qui s'arrête. Pour les processus de courte durée, utilisez une commande Buildfile.

Utilisation d'un Procfile sur AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Go utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans cette section.

> **Remarques**
>
> - Les informations contenues dans cette rubrique s'appliquent uniquement aux branches de plate-forme basées sur l'AMI Amazon Linux (AL1). AL2Les branches de AL2 plate-forme 023/ sont incompatibles avec les versions précédentes de la plateforme Amazon Linux AMI (AL1) et nécessitent des paramètres de configuration différents.
> - Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Passage de port — AMI Amazon Linux (AL1)

> **Note**
>
> Les informations contenues dans cette rubrique s'appliquent uniquement aux branches de plate-forme basées sur l'AMI Amazon Linux (AL1). AL2Les branches de AL2 plate-forme 023/ sont incompatibles avec les versions précédentes de la plateforme Amazon Linux AMI (AL1) et nécessitent des paramètres de configuration différents.

Elastic Beanstalk configure le proxy nginx pour transmettre les demandes à votre application sur le numéro de port indiqué dans la propriété d'environnement PORT de votre application. Votre application doit toujours écouter sur ce port. Vous pouvez accéder à cette variable dans votre d'application en appelant la méthode `os.Getenv("PORT")`.

Elastic Beanstalk utilise le numéro de port spécifié dans la propriété d'environnement PORT pour le port de la première application dans le fichier `Procfile`. Il augmente ensuite le numéro de port de 100 pour chaque application ultérieure dans le fichier `Procfile`. Si la propriété d'environnement PORT n'est pas définie, Elastic Beanstalk utilise 5000 pour le port initial.

Dans l'exemple précédent, la propriété d'environnement PORT pour l'application `web` est 5000, l'application `queue_process` est 5100 et l'application `foo` est 5200.

Vous pouvez spécifier le port initial en définissant l'option PORT avec l'espace de noms aws:elasticbeanstalk:application:environment, comme illustré dans l'exemple suivant.

```
option_settings:
  - namespace:  aws:elasticbeanstalk:application:environment
    option_name:  PORT
    value:  <first_port_number>
```

Pour de plus amples informations sur la définition des propriétés d'environnement pour votre application, veuillez consulter Paramètres d'option.

Construction et configuration personnalisées avec un fichier de construction sur Elastic Beanstalk

Pour spécifier une commande personnalisée de configuration et de build pour votre application Go, incluez un fichier nommé `Buildfile` à la racine de votre groupe source. Le nom de fichier est sensible à la casse. Utilisez le format suivant pour le `Buildfile` :

```
<process_name>: <command>
```

La commande dans votre `Buildfile` doit correspondre à l'expression régulière suivante : `^[A-Za-z0-9_]+:\s*.+$`.

Elastic Beanstalk ne surveille pas l'application exécutée avec un fichier `Buildfile`. Utilisez un `Buildfile` pour les commandes qui s'exécutent pendant de courtes durées et s'arrêtent après avoir

terminé leurs tâches. Pour les processus d'applications longue durée qui ne doivent pas se fermer, utilisez plutôt le Procfile.

Dans l'exemple suivant d'un `Buildfile`, `build.sh` est un script shell qui se trouve à la racine du bundle de fichiers source :

```
make: ./build.sh
```

Tous les chemins d'accès dans le `Buildfile` sont par rapport à la racine du groupe source. Si vous savez à l'avance où les fichiers résident sur l'instance, vous pouvez inclure des chemins d'accès absolus dans le `Buildfile`.

Configuration du serveur proxy

Elastic Beanstalk utilise nginx comme proxy inverse pour mapper votre application à votre équilibreur de charge Elastic Load Balancing sur le port 80. Elastic Beanstalk fournit une configuration nginx par défaut que vous pouvez étendre ou remplacer totalement par votre propre configuration.

Par défaut, Elastic Beanstalk configure le serveur proxy nginx pour transmettre les demandes à votre application sur le port 5000. Vous pouvez remplacer le port par défaut en définissant la propriété d'environnement `PORT` sur le port que votre application écoute.

> **Note**
>
> Le port que votre application écoute n'affecte pas le port que le serveur nginx écoute pour recevoir des demandes de l'équilibreur de charge.

Configuration du serveur proxy sur la version de votre plateforme

Toutes les AL2 plateformes AL2 023/ prennent en charge une fonctionnalité de configuration de proxy uniforme. Pour plus d'informations sur la configuration du serveur proxy sur les versions de votre plate-forme exécutant AL2 023/AL2, consultez. Configuration du proxy inverse

Configuration du proxy sur l'AMI Amazon Linux (antérieure à Amazon Linux 2)

> **Remarques**
>
> - Les informations contenues dans cette rubrique s'appliquent uniquement aux branches de plate-forme basées sur l'AMI Amazon Linux (AL1). AL2Les branches de AL2 plate-forme

023/ sont incompatibles avec les versions précédentes de la plateforme Amazon Linux AMI (AL1) et nécessitent des paramètres de configuration différents.

- Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Si votre environnement Elastic Beanstalk Go utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations de cette section.

Extension et remplacement de la configuration du proxy par défaut — Amazon Linux AMI () AL1

Elastic Beanstalk utilise nginx comme proxy inverse pour mapper votre application vers votre équilibreur de charge sur le port 80. Si vous souhaitez fournir votre propre configuration nginx, vous pouvez remplacer la configuration par défaut fournie par Elastic Beanstalk en incluant le fichier `.ebextensions/nginx/nginx.conf` dans votre bundle de fichiers source. Si ce fichier est présent, Elastic Beanstalk utilise à la place le fichier de configuration par défaut nginx.

Si vous souhaitez inclure des directives en plus de celles du bloc `nginx.conf` `http`, vous pouvez également fournir des fichiers de configuration supplémentaires dans le répertoire `.ebextensions/nginx/conf.d/` de votre bundle de fichiers source. Tous les fichiers de ce répertoire doivent avoir l'extension `.conf`.

Pour tirer parti des fonctionnalités fournies par Elastic Beanstalk, comme Rapports et surveillance de l'état de santé améliorés dans Elastic Beanstalk, mappages d'application automatiques et fichiers statiques, vous devez inclure la ligne suivante dans le bloc `server` de votre fichier de configuration nginx :

```
include conf.d/elasticbeanstalk/*.conf;
```

Déploiement d'applications Java avec Elastic Beanstalk

Ce chapitre fournit des instructions pour configurer et déployer vos applications Java sur AWS Elastic Beanstalk. Elastic Beanstalk facilite le déploiement, la gestion et le dimensionnement de vos applications Web Java à l'aide d'Amazon Web Services.

Vous pouvez déployer votre application en quelques minutes à l'aide de l'interface de ligne de commande Elastic Beanstalk (EB CLI) ou de la console Elastic Beanstalk. Après avoir déployé votre application Elastic Beanstalk, vous pouvez continuer à utiliser l'EB CLI pour gérer votre application et votre environnement, ou vous pouvez utiliser la console Elastic Beanstalk, ou le. AWS CLI APIs

Suivez les step-by-step instructions QuickStart pour Java pour créer et déployer une application Web Java Hello World à l'aide de l'interface de ligne de commande EB. Si vous souhaitez obtenir des step-by-step instructions pour créer une application Hello World Java JSP simple à déployer avec l'EB CLI sur notre plateforme basée sur Tomcat, essayez le. QuickStart pour Java sur Tomcat

Les branches de la plateforme Java

AWS Elastic Beanstalk prend en charge deux plateformes pour les applications Java.

- Tomcat : plate-forme basée sur Apache Tomcat, un conteneur Web open source pour les applications qui utilisent des servlets Java et JavaServer Pages (JSPs) pour répondre aux requêtes HTTP. Tomcat facilite le développement d'applications web en fournissant une configuration de sécurité multithread déclarative, ainsi qu'une personnalisation étendue. Elastic Beanstalk possède des branches de plateforme pour chacune des versions majeures actuelles de Tomcat. Pour de plus amples informations, veuillez consulter Plateforme Tomcat.
- Java SE : plate-forme pour les applications qui n'utilisent pas de conteneur Web ou qui en utilisent un autre que Tomcat, tel que Jetty ou. GlassFish Vous pouvez inclure n'importe quelle bibliothèque Java Archives (JARs) utilisée par votre application dans le bundle source que vous déployez sur Elastic Beanstalk. Pour de plus amples informations, veuillez consulter Plateforme Java SE.

Les branches récentes des plateformes Tomcat et Java SE sont basées sur Amazon Linux 2 et versions ultérieures, et utilisent Corretto AWS , la distribution Java SE. Les noms de ces branches de plate-forme incluent le mot Corretto au lieu de Java.

Pour obtenir la liste des versions actuelles de la plateforme, veuillez consulter Tomcat et Java SE dans le guide Plateformes AWS Elastic Beanstalk .

AWS outils

AWS fournit plusieurs outils pour travailler avec Java et Elastic Beanstalk. Quelle que soit la branche de plateforme que vous choisissez, vous pouvez utiliser le AWS SDK for Java pour utiliser AWS d'autres services depuis votre application Java. Le AWS SDK pour Java est un ensemble de bibliothèques qui vous permettent d' AWS APIs utiliser le code de votre application sans écrire les appels HTTP bruts à partir de zéro.

Si vous préférez gérer vos applications depuis la ligne de commande, installez l'interface de ligne de commande Elastic Beanstalk (EB CLI) et utilisez-la pour créer, surveiller et gérer vos environnements Elastic Beanstalk. Si vous exécutez plusieurs environnements pour votre application, l'interface de ligne de commande (CLI) EB s'intègre avec Git pour vous permettre d'associer chacun de vos environnements à une branche Git différente.

Rubriques

- QuickStart: Déployer une application Java sur Elastic Beanstalk
- QuickStart: Déployer une application Web Java JSP pour Tomcat sur Elastic Beanstalk
- Configuration de votre environnement de développement Java
- Autres exemples d'applications et de didacticiels Elastic Beanstalk pour Java
- Utilisation de la plateforme Elastic Beanstalk Tomcat
- Utilisation de la plateforme Java SE Elastic Beanstalk
- Ajouter une instance de base de données Amazon RDS à votre environnement Java Elastic Beanstalk
- Outils et ressources Java

QuickStart: Déployer une application Java sur Elastic Beanstalk

Ce QuickStart didacticiel explique le processus de création d'une application Java et de son déploiement dans un AWS Elastic Beanstalk environnement.

> ⚠ Non destiné à une utilisation en production
>
> Les exemples sont destinés uniquement à des fins de démonstration. N'utilisez pas d'exemples d'applications en production.

Sections

- Votre AWS compte
- Conditions préalables
- Étape 1 : Création d'une application Java
- Étape 2 : Exécutez votre application localement
- Étape 3 : Déployez votre application Java avec l'interface de ligne de commande EB
- Étape 4 : Exécutez votre application sur Elastic Beanstalk
- Étape 5 : nettoyer
- AWS ressources pour votre application
- Étapes suivantes
- Déployez avec la console Elastic Beanstalk

Votre AWS compte

Si vous n'êtes pas encore AWS client, vous devez créer un AWS compte. L'inscription vous permet d'accéder à Elastic Beanstalk AWS et aux autres services dont vous avez besoin.

Si vous avez déjà un AWS compte, vous pouvez passer àConditions préalables.

Créez un AWS compte

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez l'https://portal.aws.amazon.com/billing/inscription.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d’inscription, vous recevrez un appel téléphonique ou un SMS et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

   Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l’utilisateur racine a accès à l’ensemble des Services AWS et des ressources de ce compte. La meilleure pratique de sécurité consiste à attribuer un accès administratif à un

utilisateur, et à utiliser uniquement l'utilisateur racine pour effectuer les tâches nécessitant un accès utilisateur racine.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. À tout moment, vous pouvez consulter l'activité actuelle de votre compte et gérer votre compte en accédant à https://aws.amazon.com/et en choisissant Mon compte.

Création d'un utilisateur doté d'un accès administratif

Une fois que vous vous êtes inscrit à un utilisateur administratif Compte AWS, que vous Utilisateur racine d'un compte AWS l'avez sécurisé AWS IAM Identity Center, que vous l'avez activé et que vous en avez créé un, afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l'aide pour vous connecter en utilisant l'utilisateur racine, consultez Connexion en tant qu'utilisateur racine dans le Guide de l'utilisateur Connexion à AWS .
2. Activez l'authentification multifactorielle (MFA) pour votre utilisateur racine.

   Pour obtenir des instructions, consultez la section Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d'un utilisateur doté d'un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d' AWS IAM Identity Center dans le Guide de l'utilisateur AWS IAM Identity Center .
2. Dans IAM Identity Center, octroyez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connexion en tant qu'utilisateur doté d'un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l'URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l'utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribution d'un accès à d'autres utilisateurs

1. Dans IAM Identity Center, créez un ensemble d'autorisations qui respecte la bonne pratique consistant à appliquer les autorisations de moindre privilège.

   Pour obtenir des instructions, consultez Création d'un ensemble d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .
2. Attribuez des utilisateurs à un groupe, puis attribuez un accès par authentification unique au groupe.

   Pour obtenir des instructions, consultez Ajout de groupes dans le Guide de l'utilisateur AWS IAM Identity Center .

Conditions préalables

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

INTERFACE DE LIGNE DE COMMANDE (CLI) EB

Ce tutoriel utilise également l'interface de ligne de commande Elastic Beanstalk (CLI EB). Pour de plus amples informations sur l'installation et la configuration de la CLI EB, veuillez consulter

Installation de la CLI EB avec un script de configuration (recommandé) et Configuration de l'interface de ligne de commande EB.

Java et Maven

Si Amazon Corretto n'est pas installé sur votre machine locale, vous pouvez l'installer en suivant les instructions d'installation du guide de l'utilisateur d'Amazon Corretto.

Vérifiez votre installation Java en exécutant la commande suivante.

```
~$ java -version
```

Ce didacticiel utilise Maven. Suivez les instructions de téléchargement et d'installation sur le site Web du projet Apache Maven. Pour plus d'informations sur Maven, consultez le centre des utilisateurs de Maven sur le site Web du projet Apache Maven.

Vérifiez votre installation Maven en exécutant la commande suivante.

```
~$ mvn -v
```

Étape 1 : Création d'une application Java

Créez un répertoire de projet.

```
~$ mkdir eb-java
~$ cd eb-java
```

Créez ensuite une application qui vous allez déployer à l'aide d'Elastic Beanstalk. Nous allons créer un service RESTful Web « Hello World ».

Cet exemple utilise le framework Spring Boot. Cette application ouvre un écouteur sur le port 5000. Elastic Beanstalk transmet les demandes à votre application sur le port 5000 par défaut.

Créez les fichiers suivants :

Ce fichier crée une application Spring Boot simple.

Example~/eb-java/src/main/java/com/example/Application.java

```
package com.example;

import org.springframework.boot.SpringApplication;
```

```
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class Application {

    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }
}
```

Ce fichier crée un mappage qui renvoie une chaîne que nous définissons ici.

Example~/eb-java/src/main/java/com/example/Controller.java

```
package com.example;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class Controller {

    @GetMapping("/")
    public String index() {
       return "Hello Elastic Beanstalk!";
    }
}
```

Ce fichier définit la configuration du projet Maven.

Example~/eb-java/pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/
maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>3.2.3</version>
```

```
  </parent>

  <groupId>com.example</groupId>
  <artifactId>BeanstalkJavaExample</artifactId>
  <version>1.0-SNAPSHOT</version>

  <properties>
    <java.version>21</java.version>
  </properties>

  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-data-rest</artifactId>
    </dependency>
  </dependencies>

  <build>
    <plugins>
      <plugin>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-maven-plugin</artifactId>
      </plugin>
    </plugins>
  </build>

</project>
```

Ce fichier de propriétés remplace le port par défaut qui est 5000. Il s'agit du port par défaut vers lequel Elastic Beanstalk envoie le trafic pour les applications Java.

Example~/eb-java/application.properties

```
server.port=5000
```

Étape 2 : Exécutez votre application localement

Package de votre application à l'aide de la commande suivante :

```
~/eb-java$ mvn clean package
```

Exécutez votre application localement à l'aide de la commande suivante :

```
~/eb-java$ java -jar target/BeanstalkJavaExample-1.0-SNAPSHOT.jar
```

Pendant que l'application est en cours d'exécution, naviguez jusqu'à `http://127.0.0.1:5000/` dans votre navigateur. Vous devriez voir le texte « Hello Elastic Beanstalk ! ».

Étape 3 : Déployez votre application Java avec l'interface de ligne de commande EB

Avant de déployer votre application Java sur Elastic Beanstalk, nettoyons l'application de compilation de votre répertoire et créons un Buildfile et un Procfile pour contrôler la façon dont l'application est créée et exécutée dans votre environnement Elastic Beanstalk.

Pour préparer et configurer le déploiement de l'application

1. Nettoyez l'application créée.

   ```
   ~/eb-java$ mvn clean
   ```

2. Créez votre `Buildfile`.

   Example~/eb-java/Buildfile

   ```
   build: mvn clean package
   ```

   Ceci `Buildfile` indique la commande utilisée pour créer votre application. Si vous n'incluez pas un `Buildfile` pour une application Java, Elastic Beanstalk n'essaie pas de créer votre application.

3. Créez votre `Procfile`.

   Example~/eb-java/Procfile

   ```
   web: java -jar target/BeanstalkJavaExample-1.0-SNAPSHOT.jar
   ```

   Ceci `Procfile` indique la commande utilisée pour exécuter votre application. Si vous n'incluez pas de fichier `Procfile` pour une application Java, Elastic Beanstalk part du principe qu'il existe un fichier JAR à la racine de votre bundle source et essaie de l'exécuter à l'aide de la commande. `java -jar`

Maintenant que vous avez configuré les fichiers de configuration pour créer et démarrer votre application, vous êtes prêt à la déployer.

Pour créer un environnement et déployer votre application Java

1. Initialisez votre référentiel de la CLI EB avec la commande eb init :

   ```
   ~/eb-java eb init -p corretto java-tutorial --region us-east-2

   Application java-tutorial has been created.
   ```

   Cette commande crée une application nommée `java-tutorial` et configure votre dépôt local pour créer des environnements dotés de la dernière version de la plateforme Java.

2. (facultatif) Exécutez à nouveau la commande eb init pour configurer une paire de clés par défaut afin de pouvoir vous connecter à l'instance EC2 qui exécute votre application.

   ```
   ~/eb-java$ eb init
   Do you want to set up SSH for your instances?
   (y/n): y
   Select a keypair.
   1) my-keypair
   2) [ Create new KeyPair ]
   ```

   Sélectionnez une paire de clés si vous en avez déjà une, ou suivez les invites pour en créer une. Si vous ne voyez pas l'invite ou que vous avez besoin de modifier vos paramètres ultérieurement, exécutez eb init -i.

3. Créez un environnement et déployez-y votre application avec eb create. Elastic Beanstalk crée automatiquement un fichier zip pour votre application et le démarre sur le port 5000.

   ```
   ~/eb-java$ eb create java-env
   ```

   Il faut environ cinq minutes à Elastic Beanstalk pour créer votre environnement.

Étape 4 : Exécutez votre application sur Elastic Beanstalk

Lorsque le processus de création de votre environnement est terminé, ouvrez votre site Web aveceb open.

```
~/eb-java eb open
```

Félicitations ! Vous avez déployé une application Java avec Elastic Beanstalk ! Celle-ci ouvre une fenêtre de navigation en utilisant le nom de domaine créé pour votre application.

Étape 5 : nettoyer

Vous pouvez mettre fin à votre environnement lorsque vous avez fini d'utiliser votre application. Elastic Beanstalk AWS met fin à toutes les ressources associées à votre environnement.

Pour mettre fin à votre environnement Elastic Beanstalk avec l'EB CLI, exécutez la commande suivante.

```
~/eb-java$ eb terminate
```

AWS ressources pour votre application

Vous venez de créer une application à instance unique. Il s'agit d'un exemple d'application simple avec une seule instance EC2, de sorte qu'il ne nécessite pas d'équilibrage de charge ni de dimensionnement automatique. Pour les applications à instance unique, Elastic Beanstalk crée les ressources suivantes : AWS

- Instance EC2 – Une machine virtuelle Amazon EC2 configurée pour exécuter des applications web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble distinct de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse qui traite le trafic web devant votre application web, lui transmet les demandes, traite les ressources statiques et génère des journaux d'accès et d'erreurs.
- Groupe de sécurité de l'instance – Un groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et sont déclenchées si la charge est trop élevée ou trop faible.

Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.

- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

Elastic Beanstalk gère toutes ces ressources. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

Étapes suivantes

Dès que vous disposez d'un environnement exécutant une application, vous pouvez à tout moment déployer une nouvelle version de l'application ou une application totalement différente. Le déploiement d'une nouvelle version d'application est très rapide, car il n'est pas nécessaire de mettre en service ni de redémarrer les instances EC2. Vous pouvez également explorer votre nouvel environnement à l'aide de la console Elastic Beanstalk. Pour connaître les étapes détaillées, consultez la section Explorez votre environnement dans le chapitre Mise en route de ce guide.

> **Essayez d'autres didacticiels**
>
> Si vous souhaitez essayer d'autres didacticiels avec différents exemples d'applications, consultezExemples d'applications et de didacticiels.

Une fois que vous avez déployé un ou deux exemples d'applications et que vous êtes prêt à commencer à développer et à exécuter des applications Java en local, voirConfiguration de votre environnement de développement Java.

Déployez avec la console Elastic Beanstalk

Vous pouvez également utiliser la console Elastic Beanstalk pour lancer l'exemple d'application. Pour connaître les étapes détaillées, voir Création d'un exemple d'application dans le chapitre Mise en route de ce guide.

QuickStart: Déployer une application Web Java JSP pour Tomcat sur Elastic Beanstalk

Ce didacticiel explique le processus de création d'une application Web Java simple à l'aide de JavaServer Pages (JSPs). Si vous souhaitez regrouper plusieurs applications Web sous forme de fichiers WAR dans un seul environnement Elastic Beanstalk, consultez. Création d'une offre groupée de plusieurs fichiers WAR pour les environnements Tomcat

> ⚠ Non destiné à une utilisation en production
>
> Les exemples sont destinés uniquement à des fins de démonstration. N'utilisez pas d'exemples d'applications en production.

Sections

- Votre AWS compte
- Conditions préalables
- Étape 1 : Création d'une application Java JSP
- Étape 2 : Déployez votre application Java JSP avec l'interface de ligne de commande EB
- Étape 3 : Exécutez votre application sur Elastic Beanstalk
- Étape 4 : nettoyer
- AWS ressources pour votre application
- Étapes suivantes
- Déployez avec la console Elastic Beanstalk

Votre AWS compte

Si vous n'êtes pas encore AWS client, vous devez créer un AWS compte. L'inscription vous permet d'accéder à Elastic Beanstalk AWS et aux autres services dont vous avez besoin.

Si vous avez déjà un AWS compte, vous pouvez passer àConditions préalables.

Créez un AWS compte

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez l'https://portal.aws.amazon.com/billing/inscription.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d’inscription, vous recevrez un appel téléphonique ou un SMS et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

   Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l’utilisateur racine a accès à l’ensemble des Services AWS et des ressources de ce compte. La meilleure pratique de sécurité consiste à attribuer un accès administratif à un utilisateur, et à utiliser uniquement l’utilisateur racine pour effectuer les tâches nécessitant un accès utilisateur racine.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. À tout moment, vous pouvez consulter l'activité actuelle de votre compte et gérer votre compte en accédant à https://aws.amazon.com/et en choisissant Mon compte.

Création d’un utilisateur doté d’un accès administratif

Après vous être inscrit à un Compte AWS, sécurisez Utilisateur racine d'un compte AWS AWS IAM Identity Center, activez et créez un utilisateur administratif afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l’aide pour vous connecter en utilisant l’utilisateur racine, consultez Connexion en tant qu’utilisateur racine dans le Guide de l’utilisateur Connexion à AWS .
2. Activez l’authentification multifactorielle (MFA) pour votre utilisateur racine.

Pour obtenir des instructions, voir Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d’un utilisateur doté d’un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d’ AWS IAM Identity Center dans le Guide de l’utilisateur AWS IAM Identity Center .
2. Dans IAM Identity Center, octroyez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connexion en tant qu’utilisateur doté d’un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l’URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l’utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribution d’un accès à d’autres utilisateurs

1. Dans IAM Identity Center, créez un ensemble d’autorisations qui respecte la bonne pratique consistant à appliquer les autorisations de moindre privilège.

   Pour obtenir des instructions, consultez Création d’un ensemble d’autorisations dans le Guide de l’utilisateur AWS IAM Identity Center .
2. Attribuez des utilisateurs à un groupe, puis attribuez un accès par authentification unique au groupe.

   Pour obtenir des instructions, consultez Ajout de groupes dans le Guide de l’utilisateur AWS IAM Identity Center .

Conditions préalables

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

INTERFACE DE LIGNE DE COMMANDE (CLI) EB

Ce tutoriel utilise également l'interface de ligne de commande Elastic Beanstalk (CLI EB). Pour de plus amples informations sur l'installation et la configuration de la CLI EB, veuillez consulter Installation de la CLI EB avec un script de configuration (recommandé) et Configuration de l'interface de ligne de commande EB.

Étape 1 : Création d'une application Java JSP

Créez un répertoire de projet.

```
~$ mkdir eb-tomcat
~$ cd eb-tomcat
```

Créez ensuite une application qui vous allez déployer à l'aide d'Elastic Beanstalk. Nous allons créer une application Web « Hello World ».

Créez un fichier JSP simple nomméindex.jsp.

Example~/eb-tomcat/index.jsp

```
<html>
  <body>
    <%out.println("Hello Elastic Beanstalk!");%>
  </body>
</html>
```

Étape 2 : Déployez votre application Java JSP avec l'interface de ligne de commande EB

Exécutez les commandes suivantes pour créer un environnement Elastic Beanstalk pour cette application.

Pour créer un environnement et déployer votre application Java JSP

1. Initialisez votre référentiel de la CLI EB avec la commande eb init :

   ```
   ~/eb-tomcat$ eb init -p tomcat tomcat-tutorial --region us-east-2
   ```

   Cette commande crée une application nommée `tomcat-tutorial` et configure votre référentiel local pour créer des environnements avec la dernière version de la plateforme Tomcat.

2. (facultatif) Exécutez à nouveau la commande eb init pour configurer une paire de clés par défaut afin de pouvoir vous connecter à l'instance EC2 qui exécute votre application.

   ```
   ~/eb-go$ eb init
   Do you want to set up SSH for your instances?
   (y/n): y
   Select a keypair.
   1) my-keypair
   2) [ Create new KeyPair ]
   ```

   Sélectionnez une paire de clés si vous en avez déjà une, ou suivez les invites pour en créer une. Si vous ne voyez pas l'invite ou que vous avez besoin de modifier vos paramètres ultérieurement, exécutez eb init -i.

3. Créez un environnement et déployez-y votre application avec eb create. Elastic Beanstalk crée automatiquement un fichier zip pour votre application et le démarre sur le port 5000.

   ```
   ~/eb-tomcat$ eb create tomcat-env
   ```

   Il faut environ cinq minutes à Elastic Beanstalk pour créer votre environnement.

Étape 3 : Exécutez votre application sur Elastic Beanstalk

Lorsque le processus de création de votre environnement est terminé, ouvrez votre site Web aveceb open.

```
~/eb-tomcat$ eb open
```

Félicitations ! Vous avez déployé une application Java JSP avec Elastic Beanstalk ! Celle-ci ouvre une fenêtre de navigation en utilisant le nom de domaine créé pour votre application.

Étape 4 : nettoyer

Vous pouvez mettre fin à votre environnement lorsque vous avez fini d'utiliser votre application. Elastic Beanstalk AWS met fin à toutes les ressources associées à votre environnement.

Pour mettre fin à votre environnement Elastic Beanstalk avec l'EB CLI, exécutez la commande suivante.

```
~/eb-tomcat$ eb terminate
```

AWS ressources pour votre application

Vous venez de créer une application à instance unique. Il s'agit d'un exemple d'application simple avec une seule instance EC2. Il ne nécessite donc pas d'équilibrage de charge ni de dimensionnement automatique. Pour les applications à instance unique, Elastic Beanstalk crée les ressources suivantes : AWS

- Instance EC2 – Une machine virtuelle Amazon EC2 configurée pour exécuter des applications web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble distinct de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse qui traite le trafic web devant votre application web, lui transmet les demandes, traite les ressources statiques et génère des journaux d'accès et d'erreurs.
- Groupe de sécurité de l'instance – Un groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de

charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.

- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

Elastic Beanstalk gère toutes ces ressources. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

Étapes suivantes

Dès que vous disposez d'un environnement exécutant une application, vous pouvez à tout moment déployer une nouvelle version de l'application ou une application totalement différente. Le déploiement d'une nouvelle version d'application est très rapide, car il n'est pas nécessaire de mettre en service ni de redémarrer les instances EC2. Vous pouvez également explorer votre nouvel environnement à l'aide de la console Elastic Beanstalk. Pour connaître les étapes détaillées, consultez la section Explorez votre environnement dans le chapitre Mise en route de ce guide.

> **Essayez d'autres didacticiels**
>
> Si vous souhaitez essayer d'autres didacticiels avec différents exemples d'applications, consultezExemples d'applications et de didacticiels.

Une fois que vous avez déployé un ou deux exemples d'applications et que vous êtes prêt à commencer à développer et à exécuter des applications Java dans un conteneur Web Tomcat local, consultezConfiguration de votre environnement de développement Java.

Déployez avec la console Elastic Beanstalk

Vous pouvez également utiliser la console Elastic Beanstalk pour lancer l'exemple d'application. Pour connaître les étapes détaillées, voir Création d'un exemple d'application dans le chapitre Mise en route de ce guide.

Configuration de votre environnement de développement Java

Cette rubrique fournit des instructions pour configurer un environnement de développement Java afin de tester votre application localement avant de la déployer AWS Elastic Beanstalk. Il fait également référence à des sites Web qui fournissent des instructions d'installation pour des outils utiles.

Sections

- Installation du kit de développement Java
- Installation d'un conteneur web
- Téléchargement de bibliothèques
- Installation du AWS SDK pour Java
- Installation d'un IDE ou d'un éditeur de texte

Installation du kit de développement Java

Installez le kit de développement Java (JDK). Si vous n'avez pas de préférence, téléchargez la dernière version. Téléchargez le JDK sur oracle.com

Le JDK inclut le compilateur Java, qui vous permet de créer vos fichiers source dans des fichiers de classe qui peuvent être exécutés sur un serveur web Elastic Beanstalk.

Installation d'un conteneur web

Si vous ne possédez pas encore un autre conteneur ou framework Web, installez une version de Tomcat compatible avec Elastic Beanstalk pour votre système d'exploitation Amazon Linux. Pour obtenir la liste des versions actuelles d'Apache Tomcat prises en charge par Elastic Beanstalk, consultez Tomcat dans le document Platforms.AWS Elastic Beanstalk Téléchargez la version de Tomcat qui s'applique à votre environnement depuis le site Web d'Apache Tomcat.

Téléchargement de bibliothèques

Les plateformes Elastic Beanstalk incluent peu de bibliothèques par défaut. Téléchargez des bibliothèques que votre application va utiliser et enregistrez-les dans votre dossier de projet pour un déploiement dans le groupe source de votre application.

Si vous avez installé Tomcat localement, vous pouvez copier l'API du servlet et les bibliothèques de l'API JavaServer Pages (JSP) depuis le dossier d'installation. Si vous déployez sur une version de plateforme Tomcat, vous n'avez pas besoin d'inclure ces fichiers dans votre groupe source, mais vous devez les avoir dans votre `classpath` pour compiler toutes les classes qui les utilisent.

JUnit, Google Guava et Apache Commons fournissent plusieurs bibliothèques utiles. Visitez leurs pages d'accueil pour en savoir plus :

- Download JUnit
- Téléchargez Google Guava
- Téléchargez Apache Commons

Installation du AWS SDK pour Java

Si vous devez gérer AWS des ressources depuis votre application, installez le AWS SDK for Java. Par exemple, avec le AWS SDK pour Java, vous pouvez utiliser Amazon DynamoDB (DynamoDB) pour partager les états de session des applications Apache Tomcat sur plusieurs serveurs Web. Pour plus d'informations, consultez Gérer l'état de session Tomcat avec Amazon DynamoDB dans la documentation du AWS SDK pour Java.

Visitez la page d'accueil SDK AWS pour Java pour de plus amples informations et des instructions d'installation.

Installation d'un IDE ou d'un éditeur de texte

Les environnements de développement intégrés (IDEs) fournissent un large éventail de fonctionnalités qui facilitent le développement d'applications. Si vous n'avez pas utilisé un IDE pour le développement Java, testez Eclipse et IntelliJ puis évaluez voir ce qui vous convient le mieux.

- Installation d'IDE Eclipse pour Java EE Developers
- Installation d'IntelliJ

Un IDE peut ajouter des fichiers dans votre dossier de projet que vous pouvez ne pas souhaiter engager sur le contrôle de code source. Pour empêcher la validation de ces fichiers de contrôle de code source, utilisez `.gitignore` ou l'équivalent de votre outil de contrôle de source.

Si vous souhaitez simplement commencer le codage et que vous n'avez pas besoin de toutes les fonctionnalités d'un IDE, pensez à installer Sublime Text.

> **Note**
>
> Le 31 mai 2023, il AWS Toolkit for Eclipsea atteint sa fin de vie et n'est plus pris en charge par AWS. Pour plus de détails concernant la fin du cycle de vie du AWS Toolkit for Eclipse, consultez le fichier README.md sur le référentiel. AWS Toolkit for Eclipse GitHub

Autres exemples d'applications et de didacticiels Elastic Beanstalk pour Java

Cette section fournit des applications et des didacticiels supplémentaires. Les QuickStart pour Java sur Tomcat rubriques QuickStart pour Java et décrites plus haut dans cette rubrique vous expliquent comment lancer un exemple d'application Java avec l'interface de ligne de commande EB.

Pour commencer à utiliser les applications Java AWS Elastic Beanstalk, il vous suffit d'un bundle de sources d'applications à télécharger en tant que première version de l'application et à déployer dans un environnement. Lorsque vous créez un environnement, Elastic Beanstalk alloue toutes les ressources AWS nécessaires pour exécuter une application web évolutive.

Lancement d'un environnement avec un exemple d'application Java

Elastic Beanstalk fournit des exemples d'applications d'une seule page pour chaque plate-forme, ainsi que des exemples plus complexes illustrant l'utilisation de ressources AWS supplémentaires telles qu'Amazon RDS et des fonctionnalités spécifiques au langage ou à la plate-forme et. APIs

Les exemples de page unique correspondent au même code que celui que vous obtenez lorsque vous créez un environnement sans fournir votre propre code source. Les exemples les plus complexes sont hébergés sur un environnement Elastic Beanstalk GitHub et peuvent avoir besoin d'être compilés ou créés avant d'être déployés dans un environnement Elastic Beanstalk.

Exemples

Nom	Versions prises en charge	Type d'environnement	Source	Description
Tomc (page uniqu	Toutes les branches de plateforme Tomcat avec Corretto	Serve web Nœuc	tomcat.zip	Application web Tomcat avec une page unique (`index.jsp`) configurée pour s'afficher à la racine du site web. Pour les environnements de travail, cet exemple inclut un fichier `cron.yaml` qui configure une tâche planifiée afin d'appeler `scheduled.jsp` une fois par minute. Lorsque `scheduled.jsp` est appelé, il écrit dans un fichier journal à l'emplacement `/tmp/sample-app.log` . Enfin, un fichier de configuration est inclus dans `.ebextensions` et copie les journaux depuis `/tmp/` vers les emplacements lus par Elastic Beanstalk lorsque vous demandez les journaux de l'environnement. Si vous activez l'intégration à X-Ray dans un environnement exécutant cet exemple, l'application affiche un contenu supplémentaire concernant X-Ray et fournit une option permettant de générer des informations de débogage que vous pouvez afficher dans la console X-Ray.
Corre (page uniqu	Corretto Corretto 8	Serve web	corretto.zip	Application Corretto avec fichiers de configuration `Buildfile` et `Procfile`. Si vous activez l'intégration à X-Ray dans un environnement exécutant cet exemple, l'application affiche un contenu supplémentaire concernant X-Ray et fournit une option

Nom	Versions prises en charge	Type d'environnement	Source	Description
				permettant de générer des informations de débogage que vous pouvez afficher dans la console X-Ray.

Nom	Versions prises en charge	Type d'environnement	Source	Description
Score	Java 8	Serveur web	Clonez le dépôt sur .com GitHub	Scorekeep est une API RESTful Web qui utilise le framework Spring pour fournir une interface permettant de créer et de gérer des utilisateurs, des sessions et des jeux. Cette API est regroupée avec une application Angular 1.5 qui utilise l'API sur HTTP. L'application utilise des fonctionnalités de la plateforme Java SE pour télécharger des dépendances et des instances intégrées, ce qui réduit la taille du bundle de fichiers source. L'application inclut également des fichiers de configuration nginx qui remplacent la configuration par défaut pour servir l'application web frontale de manière statique sur le port 80 via le proxy et acheminer les requêtes vers des chemins sous `/api` vers l'API s'exécutant sur `localhost:5000` . Scorekeep inclut également une branche `xray` qui montre comment instrumenter une application Java à utiliser avec AWS X-Ray. Il montre l'instrumentation des requêtes HTTP entrantes avec un filtre de servlet, l'instrumentation automatique et manuelle du client AWS SDK, la configuration de l'enregistreur et l'instrumentation des requêtes HTTP sortantes et des clients SQL.

Nom	Versions prises en charge	Type d'environnement	Source	Description
				Pour obtenir des instructions, consultez le fichier readme ou utilisez le Tutoriel de mise en route AWS X-Ray pour essayer l'application avec X-Ray.
Does it Have Snak	Tomcat 8 avec Java 8	Serveur web	Clonez le dépôt sur .com GitHub	Y a-t-il des serpents ? est une application Web Tomcat qui montre l'utilisation des fichiers de configuration Elastic Beanstalk, d'Amazon RDS, de JDBC, de PostgreSQL, de JSPs Servlets, de Simple Tag Support, de Tag Files, de Log4J, de Bootstrap et de Jackson. Le code source de ce projet comprend un script de construction minimal qui compile les servlets et les modèles dans des fichiers de classe, et qui regroupe les fichiers requis dans une archive web que vous pouvez déployer dans un environnement Elastic Beanstalk. Pour obtenir des instructions complètes, consultez le fichier readme dans le référentiel du projet.
Locust Load Gene	Java 8	Serveur web	Clonez le dépôt sur .com GitHub	Application web qui vous permet de tester le chargement d'une autre application web exécutée dans un autre environnement Elastic Beanstalk. Elle décrit l'utilisation des fichiers `Buildfile` et `Procfile`, de DynamoDB et de Locust, un outil open source de test de charge.

Téléchargez l'un des exemples d'application et déployez-le dans Elastic Beanstalk en procédant comme suit :

Pour lancer un environnement avec une application (console)

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le volet de navigation, choisissez Applications. Sélectionnez une application existante dans la liste. Vous pouvez également choisir d'en créer un, en suivant les instructions deGestion des applications .
3. Sur la page de présentation de l'application, choisissez Create environment.

   Cela lance l'assistant de création d'environnement. L'assistant fournit un ensemble d'étapes vous permettant de créer un nouvel environnement.
4. Pour le niveau environnement, choisissez le niveau d'environnement de serveur Web ou le niveau d'environnement de travail. Vous ne pouvez pas modifier le niveau d'un environnement après sa création.

   > **Note**
   >
   > La plateforme .NET sur Windows Server ne prend pas en charge le niveau d'environnement worker.

   Les champs d'informations de l'application sont définis par défaut, en fonction de l'application que vous avez sélectionnée précédemment.

   Dans le groupe d'informations sur l'environnement, le nom de l'environnement est défini par défaut, en fonction du nom de l'application. Si vous préférez un autre nom d'environnement, vous pouvez saisir une autre valeur dans le champ. Vous pouvez éventuellement saisir un nom de domaine, sinon Elastic Beanstalk génère automatiquement une valeur. Vous pouvez également éventuellement saisir une description de l'environnement.
5. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.

   > **Note**
   >
   > Elastic Beanstalk prend en charge plusieurs versions pour la plupart des plateformes répertoriées. Par défaut, la console sélectionne la version recommandée pour la plateforme et la branche de plateforme que vous choisissez. Si votre application

nécessite une version différente, vous pouvez la sélectionner ici. Pour plus d'informations sur les versions de plateforme prises en charge, consultez the section called “Plateformes prises en charge”.

6. Pour le code d'application, plusieurs options s'offrent à vous pour continuer.
   - Pour lancer l'exemple d'application par défaut sans fournir le code source, choisissez Exemple d'application. Cette action permet de sélectionner l'application d'une seule page fournie par Elastic Beanstalk pour la plate-forme que vous avez sélectionnée précédemment.
   - Si vous avez téléchargé un exemple d'application à partir de ce guide, ou si vous avez votre propre code source pour une application, procédez comme suit.
     a. Sélectionnez Téléverser votre code.
     b. Choisissez ensuite Fichier local, puis sous Télécharger l'application, sélectionnez Choisir un fichier.
     c. Le système d'exploitation de votre machine cliente vous présentera une interface permettant de sélectionner le fichier local que vous avez téléchargé. Sélectionnez le fichier du bundle source et continuez.
7. Le choix des préréglages dépend de votre objectif pour l'environnement.
   - Si vous créez un exemple d'environnement pour en savoir plus sur Elastic Beanstalk ou un environnement de développement, choisissez Instance unique (éligible au niveau gratuit).
   - Si vous créez un environnement de production ou un environnement pour en savoir plus sur l'équilibrage de charge, choisissez l'une des options de haute disponibilité.
8. Choisissez Suivant.

Pour configurer l'accès au service

Ensuite, vous avez besoin de deux rôles. Un rôle de service permet à Elastic Beanstalk de surveiller vos instances EC2 et de mettre à niveau la plateforme de votre environnement. Un rôle de profil d'instance EC2 permet des tâches telles que la rédaction de journaux et l'interaction avec d'autres services.

Pour créer ou sélectionner le rôle de service

1. Si vous avez déjà créé un rôle de service et que vous souhaitez en choisir un existant, sélectionnez la valeur dans le menu déroulant Rôle de service et ignorez le reste de ces étapes pour créer un rôle de service.
2. Si aucune valeur n'est répertoriée pour le rôle de service, ou si vous souhaitez en créer un nouveau, passez aux étapes suivantes.
3. Pour Rôle de service, choisissez Créer un rôle.
4. Pour Type d’entité de confiance, choisissez Service AWS .
5. Dans le cas d'utilisation, choisissez Elastic Beanstalk — Environment.
6. Choisissez Suivant.
7. Vérifiez que les politiques d'autorisations incluent les éléments suivants, puis choisissez Next :
   - `AWSElasticBeanstalkEnhancedHealth`
   - `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy`
8. Choisissez Créer un rôle.
9. Retournez à l'onglet Configurer l'accès au service, actualisez la liste, puis sélectionnez le rôle de service nouvellement créé.

Pour créer ou sélectionner un profil d'instance EC2

1. Si vous avez déjà créé un profil d'instance EC2 et que vous souhaitez en choisir un existant, sélectionnez la valeur dans le menu déroulant du profil d'instance EC2 et ignorez le reste de ces étapes pour créer un profil d'instance EC2.
2. Si aucune valeur n'est répertoriée pour le profil d'instance EC2, ou si vous souhaitez en créer un nouveau, passez aux étapes suivantes.
3. Choisissez Créer un rôle.
4. Pour Type d’entité de confiance, choisissez Service AWS .
5. Dans le cas d'utilisation, choisissez Elastic Beanstalk — Compute.
6. Choisissez Suivant.
7. Vérifiez que les politiques d'autorisations incluent les éléments suivants, puis choisissez Next :
   - `AWSElasticBeanstalkWebTier`
   - `AWSElasticBeanstalkWorkerTier`

- `AWSElasticBeanstalkMulticontainerDocker`

8. Choisissez Créer un rôle.
9. Retournez à l'onglet Configurer l'accès au service, actualisez la liste, puis sélectionnez le profil d'instance EC2 nouvellement créé.

Pour terminer la configuration et la création de votre application

1. (Facultatif) Si vous avez déjà créé une paire de clés EC2, vous pouvez la sélectionner dans le menu déroulant du champ de paire de clés EC2. Vous l'utiliseriez pour vous connecter en toute sécurité à l'instance Amazon EC2 fournie par Elastic Beanstalk pour votre application. Si vous ignorez cette étape, vous pouvez toujours créer et attribuer une paire de clés EC2 une fois l'environnement créé. Pour de plus amples informations, veuillez consulter EC2 paire de clés.
2. Choisissez Skip to Review (Passer à la révision) sur la page Configure service access (Configurer l’accès au service).
3. La page Review (Révision) affiche un résumé de tous vos choix.

   Pour personnaliser davantage votre environnement, choisissez Edit (Modifier) à côté de l'étape qui inclut les éléments que vous souhaitez configurer. Les options suivantes peuvent être définies uniquement lors de la création de l'environnement :

   - Nom de l'environnement
   - Nom de domaine
   - Version de la plateforme
   - Processeur
   - Type d'équilibreur de charge
   - Palier

   Vous pouvez modifier les paramètres suivants après la création de l'environnement, mais ils requièrent la mise en œuvre de nouvelles instances ou d'autres ressources et leur application peut prendre du temps :

   - Type d'instance, volume racine, paire de clés et rôle Gestion des identités et des accès AWS (IAM)
   - Base de données interne Amazon RDS
   - VPC

Pour de plus amples informations sur tous les paramètres disponibles, veuillez consulter Assistant de création d'un environnement.

4. Choisissez Submit (Soumettre) en bas de la page pour initialiser la création de votre nouvel environnement.

Étapes suivantes

Une fois que vous disposez d'un environnement exécutant une application, vous pouvez déployer une nouvelle version de l'application ou une application totalement différente à tout moment. Le déploiement d'une nouvelle version d'application est très rapide, car il n'est pas nécessaire de mettre en service ni de redémarrer les instances EC2.

Une fois que vous avez déployé un ou deux exemples d'application et que vous êtes prêt à développer et à exécuter des applications Java localement, consultez la section suivante afin de configurer un environnement de développement Java avec tous les outils et les bibliothèques dont vous avez besoin.

Utilisation de la plateforme Elastic Beanstalk Tomcat

Cette rubrique explique comment configurer, créer et exécuter vos applications Java qui s'exécutent sur la plateforme Elastic Beanstalk Tomcat.

La plate-forme AWS Elastic Beanstalk Tomcat est un ensemble de versions de plate-forme pour les applications Web Java qui peuvent s'exécuter dans un conteneur Web Tomcat. Tomcat s'exécute derrière un serveur proxy nginx. Chaque branche de plateforme correspond à une version majeure de Tomcat.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour modifier la configuration d'un environnement en cours d'exécution. Pour éviter de perdre la configuration de votre environnement en le résiliant, vous pouvez utiliser des configurations enregistrées pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des fichiers de configuration. Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

La plateforme Tomcat Elastic Beanstalk inclut un proxy inverse qui transmet les demandes à votre application. Vous pouvez utiliser les options de configuration pour configurer le serveur proxy de manière à traiter les ressources statiques à partir d'un dossier de votre code source afin de réduire la charge sur votre application. Pour les scénarios avancés, vous pouvez inclure vos propres `.conf`fichiers dans votre bundle de fichiers source afin d'étendre la configuration proxy Elastic Beanstalk ou de la remplacer complètement.

> **Note**
>
> Elastic Beanstalk prend en charge nginx (valeur par défaut) et Apache HTTP Server en tant que serveurs proxy sur la plateforme Tomcat. Si votre environnement Elastic Beanstalk Tomcat utilise une branche de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), vous avez également la possibilité d'utiliser Apache HTTP Server Version 2.2. Apache (dernière version) est la valeur par défaut sur ces anciennes branches de plateforme. Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Vous devez compresser les applications Java dans un fichier WAR (Web application ARchive) avec une structure spécifique. Pour plus d'informations sur la structure à respecter et sur sa liaison à la structure de votre répertoire de projet, consultez Structuration de votre dossier de projet.

Pour exécuter plusieurs applications sur le même serveur web, vous pouvez regrouper plusieurs fichiers WAR dans un bundle source unique. Chaque application d'un bundle source multiple s'exécute soit sur le chemin d'accès racine (`ROOT.war` s'exécute dans *`myapp`*`.elasticbeanstalk.com/`), soit sur un chemin d'accès situé juste en dessous (`app2.war`.war s'exécute dans *`myapp`*`.elasticbeanstalk.com/`*`app2`*`/`), selon le nom du fichier WAR. Dans un bundle source WAR unique, l'application s'exécute toujours sur le chemin d'accès racine.

Les paramètres appliqués dans la console Elastic Beanstalk remplacent les mêmes paramètres des fichiers de configuration, s'ils existent. Cela vous permet d'utiliser les paramètres par défaut dans les fichiers de configuration et de les remplacer par des paramètres spécifiques à l'environnement dans la console. Pour plus d'informations sur la priorité et les autres méthodes de modification des paramètres, consultez Options de configuration.

Pour plus d'informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, consultez the section called “Extension des plateformes Linux”.

Rubriques

- Configuration de votre environnement Tomcat
- Espaces de noms de la configuration Tomcat
- Création d'une offre groupée de plusieurs fichiers WAR pour les environnements Tomcat
- Structuration de votre dossier de projet
- Configuration du serveur proxy

Configuration de votre environnement Tomcat

La plateforme Tomcat Elastic Beanstalk fournit quelques options propres à la plateforme en plus des options standard présentes sur toutes les plateformes. Ces options vous permettent de configurer la machine virtuelle Java (JVM) qui s'exécute sur les serveurs web de votre environnement et de définir les propriétés système qui fournissent les chaînes de configuration d'information à votre application.

Vous pouvez utiliser la console Elastic Beanstalk pour activer la rotation de journal sur Amazon S3 et configurer les variables que votre application peut lire depuis l'environnement.

Pour configurer votre environnement Tomcat dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.

Options de conteneur

Vous pouvez spécifier ces options spécifiques à la plateforme :

- Proxy server (Serveur proxy) – Serveur proxy à utiliser sur vos instances d'environnement. Le serveur nginx est utilisé par défaut.

Options du conteneur JVM

La taille du tas sur la machine virtuelle Java (JVM) détermine combien d'objets peuvent être créés en mémoire par votre application avant un nettoyage de la mémoire. Vous pouvez modifier les options Initial JVM Heap Size (`-Xms option`) et Maximum JVM Heap Size (argument `-Xmx`). Plus la taille initiale du tas est importante, plus le nombre d'objets pouvant être créés avant le nettoyage de la mémoire sera élevé. Toutefois, cela signifie également que le récupérateur de mémoire mettra plus de temps pour compacter le tas. La taille de tas maximale indique la quantité maximale de mémoire que la machine virtuelle Java peut allouer lorsqu'elle augmente le tas dans le cadre d'une activité intensive.

> **Note**
>
> La mémoire disponible dépend du type d'instance Amazon EC2. Pour plus d'informations sur les types d'instances EC2 disponibles pour votre environnement Elastic Beanstalk, consultez Types d'instances dans le Guide de l'utilisateur Amazon Elastic Compute Cloud pour les instances Linux.

La génération permanente est une section du tas de la machine virtuelle Java qui stocke les définitions de classe et les métadonnées associées. Pour modifier la taille de la génération permanente, saisissez la nouvelle taille dans l'option Taille maximale de la machine virtuelle Java PermGen (`-XX:MaxPermSize`). Ce paramètre s'applique uniquement à Java 7 et versions antérieures. Cette option a été déconseillée dans le JDK 8 et remplacée par l'option Size (). MaxMetaspace `-XX:MaxMetaspaceSize`

> **Important**
>
> JDK 17 a supprimé la prise en charge de l'option Java `-XX:MaxPermSize`. L'utilisation de cette option avec un environnement s'exécutant sur une branche de plateforme Elastic Beanstalk avec Corretto 17 provoquera une erreur. Elastic Beanstalk a publié sa première branche de plateforme exécutant Tomcat avec Corretto 17 le 13 juillet 2023.
> Pour plus d'informations, veuillez consulter les ressources suivantes.
>
> - Site Web de documentation Oracle Java : Options Java supprimées (français non garanti)
> - Site Web de documentation Oracle Java : section Métadonnées de classe dans Autres considérations (français non garanti)

Pour plus d'informations sur les plateformes Elastic Beanstalk et leurs composants, consultez la section Plateformes prises en charge dans le guide Plateformes AWS Elastic Beanstalk .

Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Enable log file rotation to Amazon S3 (Permettre la rotation du fichier journal sur Amazon S3) – Indique si les fichiers journaux des instances Amazon EC2 de votre application doivent être copiés dans le compartiment Amazon S3 associé à votre application.

Fichiers statiques

Pour améliorer les performances, la section des Fichiers statiques vous permet de configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Pour chaque répertoire, vous définissez le chemin virtuel sur le mappage de répertoires. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Pour en savoir plus sur la configuration des fichiers statiques à l'aide des fichiers de configuration ou de la console Elastic Beanstalk, consultez the section called “Fichiers statiques”.

Propriétés de l'environnement

Dans la section Environment Properties (Propriétés de l'environnement), vous pouvez spécifier des paramètres de configuration de l'environnement sur les instances Amazon EC2 exécutant votre application. Les propriétés de l'environnement sont passées en tant que paires clé-valeur à l'application.

La plateforme Tomcat définit une propriété d'espace réservée nommée `JDBC_CONNECTION_STRING` pour les environnements Tomcat, permettant de transmettre une chaîne de connexion à une base de données externe.

> **Note**
>
> Si vous attachez une instance de base de données RDS à votre environnement, construisez la chaîne de connexion JDBC dynamiquement à partir des propriétés d'environnement

Amazon Relational Database Service (Amazon RDS) fournies par Elastic Beanstalk. Utilisez JDBC_CONNECTION_STRING uniquement pour les instances de base de données qui ne sont pas mises en service via Elastic Beanstalk.
Pour plus d'informations sur l'utilisation d'Amazon RDS avec votre application Java, consultez Ajouter une instance de base de données Amazon RDS à votre environnement Java Elastic Beanstalk.

Pour les versions de la plateforme Tomcat publiées avant le 26 mars 2025, les variables d'environnement sont accessibles à l'aide `System.getProperty()` de. Par exemple, vous pouvez lire une propriété nommée `API_ENDPOINT` à partir d'une variable avec le code suivant.

```
String endpoint = System.getProperty("API_ENDPOINT");
```

Les versions de la plateforme Tomcat publiées le 26 mars 2025 ou après cette date peuvent également être utilisées `System.getenv` pour accéder aux variables d'environnement en texte brut. Vous pouvez continuer à utiliser pour accéder `System.getProperty` aux variables d'environnement en texte brut. Cependant, les variables d'environnement stockées sous forme de secrets ne sont disponibles qu'à l'aide de`System.getenv`. Par exemple, vous pouvez lire une variable d'environnement nommée `API_KEY` avec le code suivant.

```
String apiKey = System.getenv("API_KEY");
```

Important

L'ajout de l'`System.getenv()`accès aux variables d'environnement dans les versions de la plateforme Tomcat publiées le 26 mars 2025 ou après cette date peut provoquer un comportement inattendu dans les applications qui accordent la priorité aux variables d'environnement sur les propriétés du système Java ou lors du passage explicite de à`System.getProperty()`. `System.getenv()`
Étant donné que les propriétés système (transmises via la ligne de commande) nécessitent un échappement du shell pour les caractères spéciaux, contrairement aux variables d'environnement, les valeurs peuvent être résolues différemment lorsque vous utilisez des variables d'environnement plutôt que des propriétés système Java.
Si votre application est concernée, pensez à :

- Suppression des caractères d'échappement des valeurs de propriétés de votre environnement lors de l'utilisation `System.getenv()`
- Configuration de votre application pour une utilisation explicite `System.getProperty()`
- Tester minutieusement votre application lors de la mise à niveau afin de garantir un comportement cohérent

Pour plus d'informations, consultez Variables d'environnement et autres paramètres du logiciel.

Espaces de noms de la configuration Tomcat

Vous pouvez utiliser un fichier de configuration pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être spécifiques à la plate-forme ou s'appliquer à toutes les plateformes du service Elastic Beanstalk dans son ensemble. Les options de configuration sont organisées en espaces de noms.

La plateforme Tomcat prend en charge les options des espaces de noms suivants en plus des options prises en charge pour tous les environnements Elastic Beanstalk :

- `aws:elasticbeanstalk:container:tomcat:jvmoptions` : modifier les paramètres de la JVM. Les options de cet espace de noms correspondent aux options de la console de gestion, comme suit :
  - `Xms` – JVM command line options (Options de ligne de commande de la JVM
  - `JVM Options` – JVM command line options (Options de ligne de commande de la JVM
- `aws:elasticbeanstalk:environment:proxy` : choisissez le serveur proxy de l'environnement.

L'exemple de fichier de configuration suivant illustre l'utilisation d'options de configuration spécifiques à Tomcat :

Example.ebextensions/tomcat-settings.config

```
option_settings:
  aws:elasticbeanstalk:container:tomcat:jvmoptions:
    Xms: 512m
    JVM Options: '-Xmn128m'
```

```
  aws:elasticbeanstalk:application:environment:
    API_ENDPOINT: mywebapi.zkpexsjtmd.us-west-2.elasticbeanstalk.com
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache
```

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d' AWS CLI. Pour plus d'informations, consultez Options de configuration.

Plateforme Tomcat d'AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Tomcat utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans cette section.

> **Remarques**
>
> - Les informations contenues dans cette rubrique s'appliquent uniquement aux branches de plate-forme basées sur l'AMI Amazon Linux (AL1). AL2023Les branches/AL2 platform sont incompatibles avec les versions précédentes de la plateforme Amazon Linux AMI (AL1) et nécessitent des paramètres de configuration différents.
> - Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Espaces de noms de configuration Tomcat — AMI Amazon Linux () AL1

La plateforme Tomcat d'AMI Amazon Linux prend en charge des options supplémentaires dans les espaces de noms suivants :

- `aws:elasticbeanstalk:container:tomcat:jvmoptions` : en plus des options mentionnées précédemment sur cette page pour cet espace de noms, les anciennes versions de la plateforme AMI Amazon Linux prennent également en charge les éléments suivants :
  - `XX:MaxPermSize` – Maximum JVM permanent generation size (Taille maximum de génération permanente de la JVM

- `aws:elasticbeanstalk:environment:proxy` : en plus de choisir le serveur proxy, configurez également la compression de réponse.

L'exemple de fichier de configuration suivant illustre l'utilisation d'options de configuration de l'espace de noms du proxy.

Example.ebextensions/tomcat-settings.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    GzipCompression: 'true'
    ProxyServer: nginx
```

Inclure les fichiers de configuration d'Elastic Beanstalk — Amazon Linux AMI () AL1

Pour déployer les fichiers de configuration `.ebextensions`, incluez-les dans la source de votre application. S'il s'agit d'une seule application, ajoutez `.ebextensions` à un fichier WAR compressé en exécutant la commande suivante :

Example

```
zip -ur your_application.war .ebextensions
```

Pour une application nécessitant plusieurs fichiers WAR, consultez Création d'une offre groupée de plusieurs fichiers WAR pour les environnements Tomcat pour plus d'informations.

Création d'une offre groupée de plusieurs fichiers WAR pour les environnements Tomcat

Si votre application web comporte plusieurs composants d'application web, vous pouvez simplifier les déploiements et réduire les coûts d'exploitation en exécutant les composants dans un environnement unique, au lieu d'exécuter un environnement distinct pour chaque composant. Cette stratégie est efficace pour les applications légères qui ne nécessitent pas beaucoup de ressources, ainsi que pour les environnements de développement et de test.

Pour déployer plusieurs applications web dans votre environnement, regroupez les fichiers WAR (web application archive) de chaque composant dans un même bundle source.

Pour créer un bundle de fichiers source d'application qui contienne plusieurs fichiers WAR, organisez ces fichiers à l'aide de la structure suivante :

```
MyApplication.zip
### .ebextensions
### .platform
### foo.war
### bar.war
### ROOT.war
```

Lorsque vous déployez un ensemble de sources contenant plusieurs fichiers WAR AWS Elastic Beanstalk dans un environnement, chaque application est accessible via un chemin différent depuis le nom de domaine racine. L'exemple précédent comprend trois applications : `foo`, `bar` et `ROOT`. `ROOT.war` est un nom de fichier spécial qui demande à Elastic Beanstalk d'exécuter cette application au niveau du domaine racine, afin que les trois applications soient disponibles aux emplacements suivants : `http://MyApplication.elasticbeanstalk.com/foo`, `http://MyApplication.elasticbeanstalk.com/bar` et `http://MyApplication.elasticbeanstalk.com`.

Le bundle source peut inclure des fichiers WAR, un dossier `.ebextensions` facultatif et un dossier `.platform` facultatif. Pour de plus amples informations sur ces dossiers de configuration facultatifs, veuillez consulter the section called “Extension des plateformes Linux”.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk à l'aide de ce lien préconfiguré : console.aws.amazon.com/elasticbeanstalk/home#/newApplication? Nom de l'application = Tutoriels et type d'environnement = LoadBalanced
2. Pour Plateforme, sélectionnez la plateforme et la branche de plateforme qui correspondent à la langue utilisée par votre application, ou la plateforme Docker pour les applications basées sur des conteneurs.
3. Pour Application code (Code d'application), choisissez Upload your code (Charger votre code).
4. Choisissez Local file (Fichier local), Choose file (Choisir un fichier), puis ouvrez le bundle source.
5. Choisissez Vérifier et lancer.
6. Vérifiez les paramètres disponibles et choisissez Créer une application.

Pour plus d'informations sur la création de groupes sources, consultez Création d'un bundle de sources d'applications Elastic Beanstalk.

Structuration de votre dossier de projet

Pour fonctionner lorsqu'ils sont déployés sur un serveur Tomcat, les fichiers WAR (web application archives) Java EE (Java Platform Enterprise Edition) compilés doivent être structurés selon certaines directives. Votre répertoire de projet n'a pas besoin de respecter les mêmes normes, mais nous vous recommandons de le structurer de la même façon afin de simplifier la compilation et l'empaquetage. De plus, si vous structurez votre dossier de projet comme le contenu du fichier WAR, vous comprendrez mieux comment les fichiers sont liés et comment ils se comportent sur un serveur web.

Dans la hiérarchie recommandée ci-dessous, le code source de l'application web est placé dans un répertoire `src` afin de l'isoler du script de génération et du fichier WAR qu'il génère.

```
~/workspace/my-app/
|-- build.sh            - Build script that compiles classes and creates a WAR
|-- README.MD           - Readme file with information about your project, notes
|-- ROOT.war            - Source bundle artifact created by build.sh
`-- src                 - Source code folder
    |-- WEB-INF         - Folder for private supporting files
    |   |-- classes     - Compiled classes
    |   |-- lib         - JAR libraries
    |   |-- tags        - Tag files
    |   |-- tlds        - Tag Library Descriptor files
    |   `-- web.xml     - Deployment Descriptor
    |-- com             - Uncompiled classes
    |-- css             - Style sheets
    |-- images          - Image files
    |-- js              - JavaScript files
    `-- default.jsp     - JSP (JavaServer Pages) webpage
```

Le contenu du dossier `src` correspond à ce que vous allez empaqueter et déployer sur le serveur, à l'exception du dossier `com`. Le dossier `com` contient vos classes non compilées (fichiers `.java`). Ils doivent être compilés et placés dans le répertoire `WEB-INF/classes` afin d'être accessibles à partir du code de votre application.

Le répertoire `WEB-INF` contient le code et les configurations qui ne sont pas publiquement traités sur le serveur web. Les autres dossiers situés à la racine du répertoire source (`css`, `images` et `js`) sont publiquement disponibles à l'emplacement correspondant sur le serveur web.

L'exemple suivant est identique au répertoire de projet précédent, sauf qu'il contient davantage de fichiers et de sous-répertoires. Cet exemple de projet inclut des balises simples, des classes de

modèle et de support, et un fichier Java Server Pages (JSP) pour une ressource `record`. Il inclut également une feuille de style et, JavaScript pour Bootstrap, un fichier JSP par défaut et une page d'erreur pour les erreurs 404.

`WEB-INF/lib` inclut un fichier JAR (Java Archive) contenant le pilote JDBC (Java Database Connectivity) pour PostgreSQL. `WEB-INF/classes` est vide, car les fichiers de classe n'ont pas encore été compilés.

```
~/workspace/my-app/
|-- build.sh
|-- README.MD
|-- ROOT.war
`-- src
    |-- WEB-INF
    |   |-- classes
    |   |-- lib
    |   |   `-- postgresql-9.4-1201.jdbc4.jar
    |   |-- tags
    |   |   `-- header.tag
    |   |-- tlds
    |   |   `-- records.tld
    |   `-- web.xml
    |-- com
    |   `-- myapp
    |       |-- model
    |       |   `-- Record.java
    |       `-- web
    |           `-- ListRecords.java
    |-- css
    |   |-- bootstrap.min.css
    |   `-- myapp.css
    |-- images
    |   `-- myapp.png
    |-- js
    |   `-- bootstrap.min.js
    |-- 404.jsp
    |-- default.jsp
    `-- records.jsp
```

Création d'un fichier WAR avec un script Shell

`build.sh` est un script shell très simple qui compile les classes Java, crée un fichier WAR et le copie dans le répertoire `webapps` de Tomcat pour les tests en local.

```
cd src
javac -d WEB-INF/classes com/myapp/model/Record.java
javac -classpath WEB-INF/lib/*:WEB-INF/classes -d WEB-INF/classes com/myapp/model/
Record.java
javac -classpath WEB-INF/lib/*:WEB-INF/classes -d WEB-INF/classes com/myapp/web/
ListRecords.java

jar -cvf ROOT.war *.jsp images css js WEB-INF
cp ROOT.war /Library/Tomcat/webapps
mv ROOT.war ../
```

Dans le fichier WAR, vous trouvez la même structure que celle du répertoire `src` figurant dans l'exemple précédent, à l'exception du dossier `src/com`. La commande `jar` crée automatiquement le fichier `META-INF/MANIFEST.MF`.

```
~/workspace/my-app/ROOT.war
|-- META-INF
|   `-- MANIFEST.MF
|-- WEB-INF
|   |-- classes
|   |   `-- com
|   |       `-- myapp
|   |           |-- model
|   |           |   `-- Records.class
|   |           `-- web
|   |               `-- ListRecords.class
|   |-- lib
|   |   `-- postgresql-9.4-1201.jdbc4.jar
|   |-- tags
|   |   `-- header.tag
|   |-- tlds
|   |   `-- records.tld
|   `-- web.xml
|-- css
|   |-- bootstrap.min.css
|   `-- myapp.css
|-- images
```

```
|   `-- myapp.png
|-- js
|   `-- bootstrap.min.js
|-- 404.jsp
|-- default.jsp
`-- records.jsp
```

Utiliser `.gitignore`

Pour éviter de valider les fichiers de classe compilés et les fichiers WAR dans votre référentiel Git, ou de voir des messages les concernant lorsque vous exécutez les commandes Git, ajoutez les types de fichiers appropriés à un fichier nommé `.gitignore` dans votre dossier de projet.

~/workspace/myapp/.gitignore

```
*.zip
*.class
```

Configuration du serveur proxy

La plateforme Tomcat utilise nginx (par défaut) ou Apache HTTP Server comme proxy inverse pour relayer les requêtes du port 80 de l'instance vers votre conteneur web Tomcat écoutant sur le port 8080. Elastic Beanstalk fournit une configuration de proxy par défaut que vous pouvez étendre ou remplacer totalement par votre propre configuration.

Configuration du serveur proxy sur la version de votre plateforme

Toutes les AL2 plateformes AL2023/prennent en charge une fonctionnalité de configuration de proxy uniforme. Pour plus d'informations sur la configuration du serveur proxy sur les versions de votre plate-forme exécutant AL2023/AL2, consultezConfiguration du proxy inverse.

Configuration du proxy sur la plateforme Tomcat d'AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Tomcat utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans cette section.

> **Remarques**
>
> - Les informations de cette rubrique s'appliquent uniquement aux branches de plate-forme basées sur l'AMI Amazon Linux (AL1). AL2023Les branches/AL2 platform sont

incompatibles avec les versions précédentes de la plateforme Amazon Linux AMI (AL1) et nécessitent des paramètres de configuration différents.

- Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Choix d'un serveur proxy pour votre environnement Tomcat — Amazon Linux AMI () AL1

Les versions de la plateforme Tomcat basées sur l'AMI Amazon Linux (antérieure à Amazon Linux 2) utilisent Apache 2.4 pour le proxy par défaut. Vous pouvez choisir d'utiliser Apache 2.2 ou nginx en incluant un fichier de configuration dans votre code source. L'exemple suivant configure Elastic Beanstalk pour utiliser nginx.

Example.ebextensions/nginx-proxy.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: nginx
```

Migration d'Apache 2.2 vers Apache 2.4 — AMI Amazon Linux () AL1

Si votre application a été développée pour Apache 2.2, lisez cette section afin d'en savoir plus sur la migration vers Apache 2.4.

À compter de la version 3.0.0 de la plateforme Tomcat, qui a été publiée avec la mise à jour de la plateforme Java avec Tomcat le 24 mai 2018, Apache 2.4 est le proxy par défaut de la plateforme Tomcat. Les fichiers Apache 2.4 `.conf` sont pour la plupart, mais pas dans leur totalité, rétrocompatibles avec ceux d'Apache 2.2. Elastic Beanstalk inclut les fichiers `.conf` par défaut qui fonctionnent correctement avec chaque version Apache. Si votre application ne personnalise pas la configuration d'Apache, comme expliqué dans Extension et remplacement de la configuration Apache par défaut — Amazon Linux AMI () AL1, elle devrait migrer vers Apache 2.4 sans aucun problème.

Si votre application étend ou remplace la configuration d'Apache, il se peut que vous ayez à apporter certaines modifications pour migrer vers Apache 2.4. Pour plus d'informations, consultez Mise à jour de la version 2.2 vers la version 2.4 sur le site The Apache Software Foundation. Temporairement,

tant que vous n'avez pas réussi à migrer vers Apache 2.4, vous pouvez choisir d'utiliser Apache 2.2 avec votre application en incluant le fichier de configuration suivant dans votre code source.

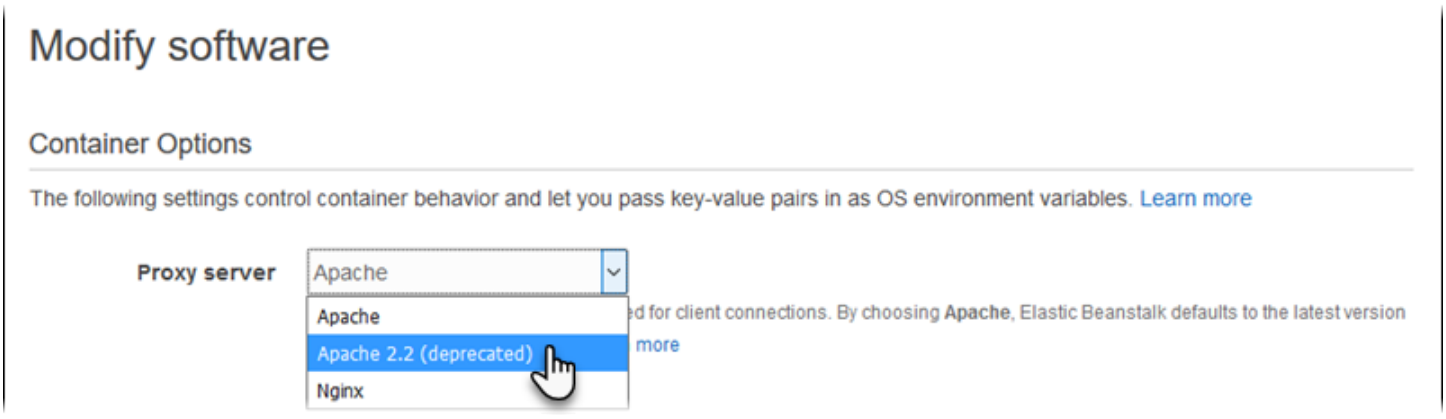
Example extensions .eb/ .config apache-legacy-proxy

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache/2.2
```

Pour une solution rapide, vous pouvez également sélectionner le serveur proxy dans la console Elastic Beanstalk.

Pour sélectionner le proxy dans votre environnement Tomcat dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Pour Serveur proxy, choisissez `Apache 2.2 (deprecated)`.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.



Extension et remplacement de la configuration Apache par défaut — Amazon Linux AMI () AL1

Vous pouvez étendre la configuration Apache Elastic Beanstalk par défaut avec vos fichiers de configuration supplémentaires. Sinon, vous pouvez remplacer complètement la configuration Apache Elastic Beanstalk par défaut.

> **Note**
>
> - Toutes les plateformes Amazon Linux 2 prennent en charge une configuration de proxy uniforme. Pour plus d'informations sur la configuration du serveur proxy sur les versions de la plateforme Tomcat exécutant Amazon Linux 2, consultezConfiguration du proxy inverse.
> - Si vous migrez votre application Elastic Beanstalk vers une plateforme Amazon Linux 2, assurez-vous de consulter également les informations de la section the section called “Migrer vers AL2 023/ AL2”.

Pour étendre la configuration Apache Elastic Beanstalk par défaut, ajoutez les fichiers de configuration `.conf` à un dossier nommé `.ebextensions/httpd/conf.d` dans le bundle de fichiers source de votre application. La configuration Apache Elastic Beanstalk par défaut inclut automatiquement les fichiers `.conf` dans ce dossier.

```
~/workspace/my-app/
|-- .ebextensions
|   -- httpd
|      -- conf.d
|         -- myconf.conf
|         -- ssl.conf
-- index.jsp
```

Par exemple, la configuration Apache 2.4 suivante ajoute un écouteur sur le port 5000 :

Example. ebextensions/httpd/conf.d/port5000.conf

```
listen 5000
<VirtualHost *:5000>
  <Proxy *>
    Require all granted
  </Proxy>
  ProxyPass / http://localhost:8080/ retry=0
```

```
  ProxyPassReverse / http://localhost:8080/
  ProxyPreserveHost on

  ErrorLog /var/log/httpd/elasticbeanstalk-error_log
</VirtualHost>
```

Pour remplacer complètement la configuration Apache Elastic Beanstalk par défaut, incluez une configuration dans votre bundle de fichiers source sur `.ebextensions/httpd/conf/httpd.conf`.

```
~/workspace/my-app/
|-- .ebextensions
|   `-- httpd
|       `-- conf
|           `-- httpd.conf
`-- index.jsp
```

Pour remplacer la configuration Apache Elastic Beanstalk par défaut, ajoutez les lignes suivantes à votre fichier `httpd.conf` afin d'extraire les configurations Elastic Beanstalk pour Rapports et surveillance de l'état de santé améliorés dans Elastic Beanstalk, la compression des réponses et les fichiers statiques.

```
IncludeOptional conf.d/*.conf
IncludeOptional conf.d/elasticbeanstalk/*.conf
```

Si votre environnement utilise Apache 2.2 comme proxy, remplacez les directives `IncludeOptional` par `Include`. Pour plus de détails sur le comportement de ces deux directives dans les deux versions d'Apache, voir Include dans Apache 2.4, IncludeOptional dans Apache 2.4 et Include dans Apache 2.2.

> **Note**
>
> Pour remplacer l'écouteur par défaut sur le port 80, incluez un fichier nommé `00_application.conf` dans `.ebextensions/httpd/conf.d/elasticbeanstalk/` afin de remplacer la configuration Elastic Beanstalk.

Pour obtenir un exemple concret, veuillez consulter le fichier de configuration par défaut Elastic Beanstalk dans `/etc/httpd/conf/httpd.conf` sur une instance de votre environnement. Tous

les fichiers du dossier `.ebextensions/httpd` de votre bundle de fichiers source sont copiés dans `/etc/httpd` au cours des déploiements.

Extension de la configuration nginx par défaut — Amazon Linux AMI () AL1

Pour étendre la configuration nginx par défaut d'Elastic Beanstalk, ajoutez les fichiers de configuration `.conf` dans un dossier nommé `.ebextensions/nginx/conf.d/` dans le bundle de fichiers source de votre application. La configuration nginx d'Elastic Beanstalk inclut automatiquement les fichiers `.conf` dans ce dossier.

```
~/workspace/my-app/
|-- .ebextensions
|   `-- nginx
|       `-- conf.d
|           |-- elasticbeanstalk
|           |   `-- my-server-conf.conf
|           `-- my-http-conf.conf
`-- index.jsp
```

Les fichiers dotés de l'extension .conf du dossier `conf.d` sont inclus dans le bloc `http` de la configuration par défaut. Les fichiers du dossier `conf.d/elasticbeanstalk` sont inclus dans le bloc `server` au sein du bloc `http`.

Pour remplacer complètement la configuration nginx par défaut d'Elastic Beanstalk, incluez une configuration dans votre bundle de fichiers source à l'emplacement `.ebextensions/nginx/nginx.conf`.

```
~/workspace/my-app/
|-- .ebextensions
|   `-- nginx
|       `-- nginx.conf
`-- index.jsp
```

> **Remarques**
>
> - Pour remplacer la configuration nginx d'Elastic Beanstalk, ajoutez la ligne suivante au bloc `server` de votre configuration afin d'extraire les configurations Elastic Beanstalk pour l'écouteur du port 80, la compression des réponses et les fichiers statiques.

```
include conf.d/elasticbeanstalk/*.conf;
```

- Pour remplacer l'écouteur par défaut sur le port 80, incluez un fichier nommé `00_application.conf` dans `.ebextensions/nginx/conf.d/elasticbeanstalk/` afin de remplacer la configuration Elastic Beanstalk.
- Incluez également la ligne suivante dans le bloc `http` de votre configuration afin d'extraire les configurations Elastic Beanstalk pour Rapports et surveillance de l'état de santé améliorés dans Elastic Beanstalk et la journalisation.

```
    include       conf.d/*.conf;
```

Pour obtenir un exemple concret, veuillez consulter le fichier de configuration par défaut Elastic Beanstalk dans `/etc/nginx/nginx.conf` sur une instance de votre environnement. Tous les fichiers du dossier `.ebextensions/nginx` de votre bundle de fichiers source sont copiés dans `/etc/nginx` au cours des déploiements.

Utilisation de la plateforme Java SE Elastic Beanstalk

Cette rubrique décrit comment configurer, créer et exécuter vos applications Java qui s'exécutent sur la plate-forme AWS Elastic Beanstalk Java SE.

La plateforme Elastic Beanstalk Java SE est un ensemble de versions de plate-forme pour les applications Web Java qui peuvent s'exécuter de manière autonome à partir d'un fichier JAR compilé. Vous pouvez compiler votre application en local ou télécharger le code source à l'aide d'un script de compilation pour le compiler sur l'instance. Les versions de la plate-forme Java SE sont regroupées en branches de plate-forme, chacune correspondant à une version majeure de Java.

> **Note**
>
> Elastic Beanstalk n'analyse pas le fichier JAR de votre application. Conservez les fichiers dont Elastic Beanstalk a besoin en dehors du fichier JAR. Par exemple, incluez le fichier `cron.yaml` d'un environnement de travail à la racine du bundle source de votre application, en regard du fichier JAR.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour modifier la configuration d'un environnement en cours d'exécution. Pour éviter de perdre la configuration de votre environnement en le résiliant, vous pouvez utiliser des configurations enregistrées pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des fichiers de configuration. Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

La plateforme Java SE Elastic Beanstalk comprend un serveur nginx qui agit comme un proxy inverse, traitant le contenu statique mis en cache et passant les demandes à votre application. La plateforme fournit également des options de configuration qui vous permettent de configurer le serveur proxy de manière à traiter les ressources statiques à partir d'un dossier de votre code source afin de réduire la charge sur votre application. Pour les scénarios avancés, vous pouvez inclure vos propres fichiers .conf dans votre bundle source afin d'étendre la configuration proxy d'Elastic Beanstalk ou de la remplacer complètement.

Si vous fournissez un seul fichier JAR pour votre source d'application (seul, pas dans un bundle de fichiers source), Elastic Beanstalk renomme votre fichier JAR en `application.jar`, puis l'exécute à l'aide de `java -jar application.jar`. Pour configurer les processus qui s'exécutent sur les instances de serveur dans votre environnement, incluez un fichier Procfile facultatif dans votre bundle de fichiers source. Un fichier `Procfile` est requis si vous avez plusieurs fichiers JAR à la racine de votre bundle de fichiers source, ou si vous souhaitez personnaliser la commande Java pour définir des options JVM.

Nous vous recommandons de toujours fournir un `Procfile` dans le bundle source avec votre application. De cette façon, vous contrôlez précisément les processus Elastic Beanstalk qui s'exécutent pour votre application et les arguments que ces processus reçoivent.

Pour compiler des classes Java et exécuter d'autres commandes de génération sur les EC2 instances de votre environnement au moment du déploiement, incluez un fichier de compilation dans le bundle de sources de votre application. A vous `Buildfile` permet de déployer votre code source tel quel et de le compiler sur le serveur au lieu de le compiler localement JARs . La plateforme Java SE inclut des outils de développement courants pour vous permettre de développer sur le serveur.

Pour de plus amples informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, veuillez consulter the section called “Extension des plateformes Linux”.

Configuration de votre environnement Java SE

Les paramètres de la plateforme Java SE vous permettent d'affiner le comportement de vos EC2 instances Amazon. Vous pouvez modifier la configuration de l'instance Amazon de l'environnement Elastic Beanstalk à l'aide EC2 de la console Elastic Beanstalk.

Utilisez la console Elastic Beanstalk pour permettre la rotation des journaux sur Amazon S3 et configurer des variables que votre application peut lire à partir de l'environnement.

Pour configurer votre environnement Java SE dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.

Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Activer la rotation des fichiers journaux vers Amazon S3 : indique si les fichiers journaux des EC2 instances Amazon de votre application sont copiés dans le compartiment Amazon S3 associé à votre application.

Fichiers statiques

Pour améliorer les performances, la section des Fichiers statiques vous permet de configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Pour chaque répertoire, vous définissez le chemin virtuel sur le mappage de répertoires. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Pour en savoir plus sur la configuration des fichiers statiques à l'aide des fichiers de configuration ou de la console Elastic Beanstalk, consultez the section called “Fichiers statiques”.

Propriétés de l'environnement

La section Propriétés de l'environnement vous permet de définir les paramètres de configuration de l'environnement sur EC2 les instances Amazon qui exécutent votre application. Les propriétés de l'environnement sont passées en tant que paires clé-valeur à l'application.

Dans l'environnement Java SE en cours d'exécution dans Elastic Beanstalk, les variables d'environnement sont accessibles à l'aide de `System.getenv()`. Par exemple, vous pouvez lire une propriété nommée `API_ENDPOINT` sur une variable avec le code suivant :

```
String endpoint = System.getenv("API_ENDPOINT");
```

Pour plus d'informations, consultez Variables d'environnement et autres paramètres du logiciel.

Espaces de noms de la configuration Java SE

Vous pouvez utiliser un fichier de configuration pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être spécifiques à la plate-forme ou s'appliquer à toutes les plateformes du service Elastic Beanstalk dans son ensemble. Les options de configuration sont organisées en espaces de noms.

La plate-forme Java SE ne définit aucun espace de noms spécifique à la plate-forme. Vous pouvez configurer le proxy pour qu'il traite les fichiers statiques à l'aide de l'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles`. Pour plus de détails et un exemple, reportez-vous à la section the section called “Fichiers statiques”.

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d' AWS CLI. Pour plus d'informations, consultez Options de configuration.

La plate-forme Java SE d'AMI Linux Amazon (antérieure à Amazon Linux 2)

Si votre environnement Java SE Elastic Beanstalk utilise une version de plate-forme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans cette section.

Remarques

- Les informations contenues dans cette rubrique s'appliquent uniquement aux branches de plate-forme basées sur l'AMI Amazon Linux (AL1). AL2Les branches de AL2 plate-forme 023/ sont incompatibles avec les versions précédentes de la plateforme Amazon Linux AMI (AL1) et nécessitent des paramètres de configuration différents.
- Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Espaces de noms de configuration Java SE — AMI Amazon Linux () AL1

Vous pouvez utiliser un fichier de configuration pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être spécifiques à la plate-forme ou s'appliquer à toutes les plateformes du service Elastic Beanstalk dans son ensemble. Les options de configuration sont organisées en espaces de noms.

La plateforme Java SE prend en charge l'espace de noms de configuration spécifique à la plateforme en plus des espaces de noms pris en charge par toutes les plateformes. L'espace de noms `aws:elasticbeanstalk:container:java:staticfiles` vous permet de définir des options qui mappe des chemins d'accès sur votre application web vers des dossiers dans le groupe source de votre application incluant le contenu statique.

Par exemple, cet extrait option_settings définit deux options dans l'espace de noms de fichiers statiques. La première mappe le chemin d'accès `/public` dans un dossier nommé `public` et la deuxième mappe le chemin d'accès `/images` dans un dossier nommé `img` :

```
option_settings:
  aws:elasticbeanstalk:container:java:staticfiles:
    /html: statichtml
    /images: staticimages
```

Les dossiers que vous mappez à l'aide de cet espace de noms doivent être de véritables dossiers à la racine de votre groupe source. Vous ne pouvez pas mapper un chemin d'accès à un dossier dans un fichier JAR.

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d' AWS CLI. Pour plus d'informations, consultez Options de configuration.

Construire JARs sur le serveur avec un Buildfile

Vous pouvez créer les fichiers de classe et les fichiers JAR de votre application sur les EC2 instances de votre environnement en invoquant une commande de construction à partir d'un `Buildfile` fichier de votre bundle source.

Les commandes dans un `Buildfile` sont exécutées une seule fois et doivent se terminer à la fin, tandis que les commandes dans un Procfile sont censées s'exécuter pendant la durée de vie de l'application et seront redémarrées si elles s'arrêtent. Pour exécuter le JARs dans votre application, utilisez un`Procfile`.

Pour plus de détails sur le placement et la syntaxe de a`Buildfile`, consultezBuildfile et Procfile.

L'exemple de `Buildfile` ci-dessous exécute Apache Maven pour générer une application web à partir du code source. Pour un exemple d'application qui utilise cette fonctionnalité, consultez Exemples d'applications web Java.

Example BuildFile

```
build: mvn assembly:assembly -DdescriptorId=jar-with-dependencies
```

La plateforme Java SE inclut les outils de génération suivants que vous pouvez appeler à partir de votre script de build :

- `javac` : compilateur Java
- `ant` : Apache Ant
- `mvn` : Apache Maven
- `gradle` : Gradle

Configuration du processus de l'application avec un Procfile

Si vous avez plus d'un fichier JAR à la racine du groupe source de votre application, vous devez inclure un fichier `Procfile` qui indique à Elastic Beanstalk quels JAR exécuter. Vous pouvez

également inclure un fichier `Procfile` pour une seule application JAR pour configurer la machine virtuelle Java (JVM) qui exécute votre application.

Nous vous recommandons de toujours fournir un `Procfile` dans le bundle source avec votre application. De cette façon, vous contrôlez précisément les processus Elastic Beanstalk qui s'exécutent pour votre application et les arguments que ces processus reçoivent.

Pour plus de détails sur l'écriture et l'utilisation d'un, `Procfile` voirBuildfile et Procfile.

Example Procfile

```
web: java -Xms256m -jar server.jar
cache: java -jar mycache.jar
web_foo: java -jar other.jar
```

La commande qui exécute le JAR principal dans votre application doit être appelée `web`, et il doit s'agir de la première commande figurant dans votre `Procfile`. Le serveur nginx transmet à cette application toutes les requêtes HTTP qu'il reçoit de l'équilibreur de charge de votre environnement.

Elastic Beanstalk part du principe que toutes les entrées dans Procfile doivent s'exécuter en permanence et redémarre automatiquement toute application définie dans le Procfile qui s'arrête. Pour exécuter des commandes qui s'arrêteront et ne devraient pas être redémarrées, utilisez un `Buildfile`.

Utilisation d'un Procfile sur AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Java SE Elastic Beanstalk utilise une version de plate-forme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans cette section.

> **Remarques**
>
> - Les informations de cette rubrique s'appliquent uniquement aux branches de plate-forme basées sur l'AMI Amazon Linux (AL1). AL2Les branches de AL2 plate-forme 023/ sont incompatibles avec les versions précédentes de la plateforme Amazon Linux AMI (AL1) et nécessitent des paramètres de configuration différents.
> - Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023

actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Passage de port — AMI Amazon Linux (AL1)

Par défaut, Elastic Beanstalk configure le serveur proxy nginx pour transmettre les demandes à votre application sur le port 5000. Vous pouvez remplacer le port par défaut en définissant la propriété d'environnement `PORT` sur le port que votre application écoute.

Si vous utilisez un `Procfile` pour exécuter plusieurs applications, Elastic Beanstalk sur les versions de plateforme AMI Amazon Linux s'attend à ce que chaque application supplémentaire écoute sur un port 100 supérieur au précédent. Elastic Beanstalk définit la variable PORT accessible à partir de chaque application avec le port sur lequel il s'attend à ce que l'application s'exécute. Vous pouvez accéder à cette variable dans votre code d'application en appelant `System.getenv("PORT")`.

Dans l'exemple `Procfile` précédent, l'application `web` écoute sur le port 5000, `cache` écoute sur le port 5100 et `web_foo` écoute sur le port 5200. `web` configure son port d'écoute en lisant la variable `PORT` et ajoute 100 à ce nombre pour déterminer sur quel port `cache` écoute afin de pouvoir lui envoyer des demandes.

Configuration du serveur proxy

Elastic Beanstalk utilise nginx comme proxy inverse pour mapper votre application à votre équilibreur de charge Elastic Load Balancing sur le port 80. Elastic Beanstalk fournit une configuration nginx par défaut que vous pouvez étendre ou remplacer totalement par votre propre configuration.

Par défaut, Elastic Beanstalk configure le serveur proxy nginx pour transmettre les demandes à votre application sur le port 5000. Vous pouvez remplacer le port par défaut en définissant la propriété d'environnement `PORT` sur le port que votre application écoute.

> **Note**
>
> Le port que votre application écoute n'affecte pas le port que le serveur nginx écoute pour recevoir des demandes de l'équilibreur de charge.

Configuration du serveur proxy sur la version de votre plateforme

Toutes les AL2 plateformes AL2 023/ prennent en charge une fonctionnalité de configuration de proxy uniforme. Pour plus d'informations sur la configuration du serveur proxy sur les versions de votre plate-forme exécutant AL2 023/AL2, consultez. Configuration du proxy inverse

Configuration du proxy sur l'AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Java SE Elastic Beanstalk utilise une version de plate-forme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans cette section.

> **Remarques**
>
> - Les informations de cette rubrique s'appliquent uniquement aux branches de plate-forme basées sur l'AMI Amazon Linux (AL1). AL2Les branches de AL2 plate-forme 023/ sont incompatibles avec les versions précédentes de la plateforme Amazon Linux AMI (AL1) et nécessitent des paramètres de configuration différents.
> - Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Extension et remplacement de la configuration du proxy par défaut — Amazon Linux AMI () AL1

Pour étendre la configuration nginx par défaut d'Elastic Beanstalk, ajoutez les fichiers de configuration `.conf` dans un dossier nommé `.ebextensions/nginx/conf.d/` dans le bundle de fichiers source de votre application. La configuration nginx Elastic Beanstalk inclut automatiquement les fichiers `.conf` dans ce dossier.

```
~/workspace/my-app/
|-- .ebextensions
|   `-- nginx
|       `-- conf.d
|           `-- myconf.conf
`-- web.jar
```

Pour remplacer complètement la configuration nginx Elastic Beanstalk par défaut, incluez une configuration dans votre bundle de fichiers source su `.ebextensions/nginx/nginx.conf`:

```
~/workspace/my-app/
|-- .ebextensions
|   `-- nginx
|       `-- nginx.conf
`-- web.jar
```

Si vous substituez une configuration nginx Elastic Beanstalk, ajoutez la ligne suivante à votre `nginx.conf` pour extraire les configurations Elastic Beanstalk pour Rapports et surveillance de l'état de santé améliorés dans Elastic Beanstalk, les mappages d'application automatiques et les fichiers statiques.

```
 include conf.d/elasticbeanstalk/*.conf;
```

L'exemple de configuration suivant provenant de l'exemple d'application Scorekeep remplace la configuration Elastic Beanstalk par défaut pour servir une application web statique depuis le sous-répertoire `public` de `/var/app/current`, où la plateforme Java SE copie le code source de l'application. L'emplacement `/api` achemine le trafic vers des routes sous `/api/` jusqu'à l'application Spring à l'écoute sur le port 5000. Le reste du trafic est servi par l'application web au chemin racine.

Example

```
user                    nginx;
error_log               /var/log/nginx/error.log warn;
pid                     /var/run/nginx.pid;
worker_processes        auto;
worker_rlimit_nofile    33282;

events {
    worker_connections  1024;
}

http {
  include       /etc/nginx/mime.types;
  default_type  application/octet-stream;

  log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                    '$status $body_bytes_sent "$http_referer" '
                    '"$http_user_agent" "$http_x_forwarded_for"';

  include       conf.d/*.conf;
```

```
  map $http_upgrade $connection_upgrade {
      default     "upgrade";
  }

  server {
      listen        80 default_server;
      root /var/app/current/public;

      location / {
      }git pull


      location /api {
          proxy_pass          http://127.0.0.1:5000;
          proxy_http_version  1.1;

          proxy_set_header    Connection          $connection_upgrade;
          proxy_set_header    Upgrade             $http_upgrade;
          proxy_set_header    Host                $host;
          proxy_set_header    X-Real-IP           $remote_addr;
          proxy_set_header    X-Forwarded-For     $proxy_add_x_forwarded_for;
      }

      access_log    /var/log/nginx/access.log main;

      client_header_timeout 60;
      client_body_timeout   60;
      keepalive_timeout     60;
      gzip                  off;
      gzip_comp_level       4;

      # Include the Elastic Beanstalk generated locations
      include conf.d/elasticbeanstalk/01_static.conf;
      include conf.d/elasticbeanstalk/healthd.conf;
  }
}
```

Ajouter une instance de base de données Amazon RDS à votre environnement Java Elastic Beanstalk

Cette rubrique fournit des instructions pour créer un Amazon RDS à l'aide de la console Elastic Beanstalk. Vous pouvez utiliser une instance de base de données Amazon Relational Database

Service (Amazon RDS) pour stocker les données que votre application recueille et modifie. La base de données peut être associée à votre environnement et gérée par Elastic Beanstalk, ou créée et gérée en externe.

Si vous utilisez Amazon RDS pour la première fois, ajoutez une instance de base de données à un environnement de test à l'aide de la console Elastic Beanstalk et assurez-vous que votre application peut s'y connecter.

Pour ajouter une instance DB à votre environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

L'ajout d'une instance DB prend environ 10 minutes. Une fois la mise à jour de l'environnement terminée, le nom d'hôte de l'instance DB et les autres informations de connexion sont disponibles dans votre application, via les propriétés d'environnement suivantes :

Nom de la propriété	Description	Valeur de la propriété
`RDS_HOSTNAME`	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
`RDS_PORT`	Port sur lequel l'instance de base de données accepte des connexions. La valeur par	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.

Nom de la propriété	Description	Valeur de la propriété
	défaut varie selon les moteurs de base de données.	
RDS_DB_NAME	Nom de la base de données, **ebdb**.	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

Pour de plus amples informations sur la configuration d'une instance de base de données interne, veuillez consulter Ajout d'une base de données à votre environnement Elastic Beanstalk. Pour de plus amples informations sur la configuration d'une base de données externe à utiliser avec Elastic Beanstalk, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon RDS.

Pour vous connecter à la base de données, ajoutez le fichier JAR du pilote approprié à votre application, chargez la classe de pilote dans votre code et créez un objet de connexion avec les propriétés d'environnement fournies par Elastic Beanstalk.

Sections

- Téléchargement du pilote JDBC
- Connexion à une base de données (plateformes Java SE)
- Connexion à une base de données (plateformes Tomcat)
- Résolution des problèmes de connexion à une base de données

Téléchargement du pilote JDBC

Vous avez besoin du fichier JAR du pilote JDBC pour le moteur de base de données que vous choisissez. Enregistrez le fichier JAR dans votre code source et incluez-le dans votre chemin de classe lorsque vous compilez la classe qui crée les connexions à la base de données.

Vous trouverez le pilote le plus récent pour votre moteur de base de données dans les emplacements suivants :

- MySQL : MySQL Connector/J
- Oracle SE-1 : pilote Oracle JDBC
- Postgres : pilote PostgreSQL JDBC
- SQL Server : pilote Microsoft JDBC

Pour utiliser le pilote JDBC, appelez `Class.forName()` afin de le charger avant de créer la connexion avec `DriverManager.getConnection()` dans votre code.

JDBC utilise une chaîne de connexion au format suivant :

```
jdbc:driver://hostname:port/dbName?user=userName&password=password
```

Vous pouvez récupérer le nom d'hôte, le port, le nom de la base de données, le nom d'utilisateur et le mot de passe dans les variables d'environnement qu'Elastic Beanstalk fournit à votre application. Le nom du pilote est spécifique à votre type de base de données et à la version de votre pilote. Voici des exemples de noms de pilote :

- `mysql` pour MySQL
- `postgresql` pour PostgreSQL
- `oracle:thin` pour Oracle Thin
- `oracle:oci` pour Oracle OCI
- `oracle:oci8` pour Oracle OCI 8
- `oracle:kprb` pour Oracle KPRB
- `sqlserver` pour SQL Server

Connexion à une base de données (plateformes Java SE)

Dans un environnement Java SE, utilisez `System.getenv()` pour lire les variables de connexion à partir de l'environnement. L'exemple de code suivant montre une classe qui crée une connexion à une base de données PostgreSQL.

```
private static Connection getRemoteConnection() {
    if (System.getenv("RDS_HOSTNAME") != null) {
      try {
      Class.forName("org.postgresql.Driver");
      String dbName = System.getenv("RDS_DB_NAME");
      String userName = System.getenv("RDS_USERNAME");
      String password = System.getenv("RDS_PASSWORD");
      String hostname = System.getenv("RDS_HOSTNAME");
      String port = System.getenv("RDS_PORT");
      String jdbcUrl = "jdbc:postgresql://" + hostname + ":" + port + "/" + dbName + "?
user=" + userName + "&password=" + password;
      logger.trace("Getting remote connection with connection string from environment
 variables.");
      Connection con = DriverManager.getConnection(jdbcUrl);
      logger.info("Remote connection successful.");
      return con;
    }
    catch (ClassNotFoundException e) { logger.warn(e.toString());}
    catch (SQLException e) { logger.warn(e.toString());}
    }
    return null;
  }
```

Connexion à une base de données (plateformes Tomcat)

Dans un environnement Tomcat, les propriétés de l'environnement sont fournies sous la forme de propriétés système auxquelles vous pouvez accéder via `System.getProperty()`.

L'exemple de code suivant montre une classe qui crée une connexion à une base de données PostgreSQL.

```
private static Connection getRemoteConnection() {
    if (System.getProperty("RDS_HOSTNAME") != null) {
      try {
      Class.forName("org.postgresql.Driver");
      String dbName = System.getProperty("RDS_DB_NAME");
```

```
      String userName = System.getProperty("RDS_USERNAME");
      String password = System.getProperty("RDS_PASSWORD");
      String hostname = System.getProperty("RDS_HOSTNAME");
      String port = System.getProperty("RDS_PORT");
      String jdbcUrl = "jdbc:postgresql://" + hostname + ":" + port + "/" + dbName + "?
user=" + userName + "&password=" + password;
      logger.trace("Getting remote connection with connection string from environment
 variables.");
      Connection con = DriverManager.getConnection(jdbcUrl);
      logger.info("Remote connection successful.");
      return con;
    }
    catch (ClassNotFoundException e) { logger.warn(e.toString());}
    catch (SQLException e) { logger.warn(e.toString());}
    }
    return null;
  }
```

Si vous rencontrez des difficultés pour obtenir une connexion ou exécuter des instructions SQL, essayez d'insérer le code suivant dans un fichier JSP. Ce code se connecte à une instance de base de données, crée une table et écrit dedans.

```
<%@ page import="java.sql.*" %>
<%
  // Read RDS connection information from the environment
  String dbName = System.getProperty("RDS_DB_NAME");
  String userName = System.getProperty("RDS_USERNAME");
  String password = System.getProperty("RDS_PASSWORD");
  String hostname = System.getProperty("RDS_HOSTNAME");
  String port = System.getProperty("RDS_PORT");
  String jdbcUrl = "jdbc:mysql://" + hostname + ":" +
    port + "/" + dbName + "?user=" + userName + "&password=" + password;

  // Load the JDBC driver
  try {
    System.out.println("Loading driver...");
    Class.forName("com.mysql.jdbc.Driver");
    System.out.println("Driver loaded!");
  } catch (ClassNotFoundException e) {
    throw new RuntimeException("Cannot find the driver in the classpath!", e);
  }

  Connection conn = null;
```

```
  Statement setupStatement = null;
  Statement readStatement = null;
  ResultSet resultSet = null;
  String results = "";
  int numresults = 0;
  String statement = null;

  try {
    // Create connection to RDS DB instance
    conn = DriverManager.getConnection(jdbcUrl);

    // Create a table and write two rows
    setupStatement = conn.createStatement();
    String createTable = "CREATE TABLE Beanstalk (Resource char(50));";
    String insertRow1 = "INSERT INTO Beanstalk (Resource) VALUES ('EC2 Instance');";
    String insertRow2 = "INSERT INTO Beanstalk (Resource) VALUES ('RDS Instance');";

    setupStatement.addBatch(createTable);
    setupStatement.addBatch(insertRow1);
    setupStatement.addBatch(insertRow2);
    setupStatement.executeBatch();
    setupStatement.close();

  } catch (SQLException ex) {
    // Handle any errors
    System.out.println("SQLException: " + ex.getMessage());
    System.out.println("SQLState: " + ex.getSQLState());
    System.out.println("VendorError: " + ex.getErrorCode());
  } finally {
    System.out.println("Closing the connection.");
    if (conn != null) try { conn.close(); } catch (SQLException ignore) {}
  }

  try {
    conn = DriverManager.getConnection(jdbcUrl);

    readStatement = conn.createStatement();
    resultSet = readStatement.executeQuery("SELECT Resource FROM Beanstalk;");

    resultSet.first();
    results = resultSet.getString("Resource");
    resultSet.next();
    results += ", " + resultSet.getString("Resource");
```

```
    resultSet.close();
    readStatement.close();
    conn.close();

  } catch (SQLException ex) {
    // Handle any errors
    System.out.println("SQLException: " + ex.getMessage());
    System.out.println("SQLState: " + ex.getSQLState());
    System.out.println("VendorError: " + ex.getErrorCode());
  } finally {
       System.out.println("Closing the connection.");
      if (conn != null) try { conn.close(); } catch (SQLException ignore) {}
  }
%>
```

Pour afficher les résultats, insérez le code suivant dans le corps de la partie HTML du fichier JSP.

```
<p>Established connection to RDS. Read first two rows: <%= results %></p>
```

Résolution des problèmes de connexion à une base de données

> **Essayez Amazon Q Developer CLI pour un dépannage assisté par l'IA**
>
> La CLI Amazon Q Developer peut vous aider à résoudre rapidement les problèmes d'environnement. La Q CLI fournit des solutions en vérifiant l'état de l'environnement, en examinant les événements, en analysant les journaux et en posant des questions de clarification. Pour plus d'informations et des instructions détaillées, consultez la section Résolution des problèmes liés aux environnements Elastic Beanstalk avec Amazon Q Developer CLI dans les blogs. AWS

Si vous rencontrez des problèmes pour vous connecter à une base de données à partir de votre application, consultez la base de données et le journal du conteneur web.

Examen des journaux

Vous pouvez afficher tous les journaux de votre environnement Elastic Beanstalk depuis Eclipse. Si la vue AWS Explorateur n'est pas ouverte, cliquez sur la flèche située à côté de l' AWS icône orange dans la barre d'outils, puis choisissez Afficher la vue AWS Explorateur. Développez AWS

Elastic Beanstalk et le nom de votre environnement, puis ouvrez le menu contextuel (clic droit) pour le serveur. Choisissez Open in WTP Server Editor (Ouvrir dans WTP Server Editor).

Choisissez l'onglet Log (Journal) de la vue Server (Serveur) afin de visualiser les journaux cumulés provenant de votre environnement. Pour ouvrir les journaux les plus récents, cliquez sur le bouton Refresh (Actualiser) dans l'angle supérieur droit de la page.

Faites défiler la page afin de trouver les journaux Tomcat dans `/var/log/tomcat7/catalina.out`. Si vous avez chargé plusieurs fois la page web issue de notre exemple précédent, les données ci-dessous devraient s'afficher :

```
-------------------------------------
/var/log/tomcat7/catalina.out
-------------------------------------
INFO: Server startup in 9285 ms
Loading driver...
Driver loaded!
SQLException: Table 'Beanstalk' already exists
SQLState: 42S01
VendorError: 1050
Closing the connection.
Closing the connection.
```

Toutes les informations envoyées par l'application à la sortie standard s'affichent dans le journal du conteneur web. Dans l'exemple précédent, l'application tente de créer la table à chaque chargement de la page. Cela se traduit par l'interception d'une exception SQL pour chaque chargement de page après le premier.

L'exemple qui précède est acceptable. Toutefois, dans des applications réelles, vous devez conserver vos définitions de base de données dans les objets de schéma, effectuer les transactions à partir des classes de modèle et coordonner les demandes avec des servlets contrôleurs.

Connexion à une instance DB RDS

Vous pouvez vous connecter directement à l'instance de base de données RDS dans votre environnement Elastic Beanstalk à l'aide de l'application cliente MySQL.

Commencez par ouvrir le groupe de sécurité dans votre instance DB RDS afin d'autoriser le trafic provenant de votre ordinateur.

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. À côté de l'option Endpoint (Point de terminaison), choisissez le lien de la console Amazon RDS.
6. Sur la page des détails de l'instance RDS Dashboard (Tableau de bord RDS), sous Security and Network (Sécurité et réseau), sélectionnez le groupe de sécurité commençant par rds- en regard de Security Groups (Groupes de sécurité).

   > **Note**
   >
   > La base de données peut comporter plusieurs entrées associées au libellé Security Groups (Groupes de sécurité). Utilisez le premier, qui commence par awseb, uniquement si vous avez un compte plus ancien qui n'a pas d'Amazon Virtual Private Cloud (Amazon VPC) par défaut.

7. Dans Security group details (Détails du groupe de sécurité), cliquez sur l'onglet Inbound (Entrant), puis choisissez Edit (Modifier).
8. Ajoutez une règle pour MySQL (port 3306) qui autorise le trafic provenant de votre adresse IP, spécifiée au format CIDR.
9. Choisissez Enregistrer. Les modifications prennent effet immédiatement.

Revenez sur la page des détails de configuration Elastic Beanstalk pour votre environnement et notez le point de terminaison. Vous devrez utiliser le nom de domaine pour vous connecter à l'instance DB RDS.

Installez le client MySQL et lancez une connexion à la base de données sur le port 3306. Sous Windows, installez MySQL Workbench à partir de la page d'accueil MySQL et suivez les instructions.

Sous Linux, installez le client MySQL en utilisant le gestionnaire de package correspondant à votre distribution. L'exemple suivant fonctionne sur Ubuntu et d'autres dérivés Debian.

```
// Install MySQL client
$ sudo apt-get install mysql-client-5.5
...
```

```
// Connect to database
$ mysql -h aas839jo2vwhwb.cnubrrfwfka8.us-west-2.rds.amazonaws.com -u username -
ppassword ebdb
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 117
Server version: 5.5.40-log Source distribution
...
```

Une fois connecté, vous pouvez exécuter les commandes SQL pour afficher l'état de la base de données, savoir si vos tables et vos lignes ont été créées, et obtenir d'autres informations.

```
mysql> SELECT Resource from Beanstalk;
+--------------+
| Resource     |
+--------------+
| EC2 Instance |
| RDS Instance |
+--------------+
2 rows in set (0.01 sec)
```

Outils et ressources Java

Il existe plusieurs endroits auxquels vous pouvez accéder pour obtenir une aide supplémentaire lors du développement de vos applications Java.

Ressource	Description
Le forum de développement AWS Java	Posez vos questions et obtenez des commentaires.
Centre pour développeurs Java	Guichet unique pour l'exemple de code, la documenta tion, les outils et les ressources supplémentaires.

Déploiement d'applications Node.js avec Elastic Beanstalk

Ce chapitre fournit des instructions pour configurer et déployer votre application Web Node.js sur AWS Elastic Beanstalk. Il fournit également des instructions pour les tâches courantes telles que l'intégration de bases de données et l'utilisation du framework Express. Elastic Beanstalk facilite le déploiement, la gestion et le dimensionnement de vos applications Web Node.js à l'aide d'Amazon Web Services.

Vous pouvez déployer votre application en quelques minutes à l'aide de l'interface de ligne de commande Elastic Beanstalk (EB CLI) ou de la console Elastic Beanstalk. Après avoir déployé votre application Elastic Beanstalk, vous pouvez continuer à utiliser l'EB CLI pour gérer votre application et votre environnement, ou vous pouvez utiliser la console Elastic Beanstalk, ou le. AWS CLI APIs

Suivez les step-by-step instructions QuickStart pour Node.js pour créer et déployer une application Web Hello World Node.js avec l'interface de ligne de commande EB.

Rubriques

- QuickStart: Déployer une application Node.js sur Elastic Beanstalk
- Configuration de votre environnement de développement Node.js pour Elastic Beanstalk
- Utilisation de la plateforme Elastic Beanstalk Node.js
- Autres exemples d'applications et de didacticiels Elastic Beanstalk pour Node.js
- Déploiement d'une application Node.js Express sur Elastic Beanstalk
- Déploiement d'une application Node.js Express avec clustering sur Elastic Beanstalk
- Déploiement d'une application Node.js avec DynamoDB vers Elastic Beanstalk
- Ajout d'une instance de base de données Amazon RDS à votre environnement Node.js Elastic Beanstalk
- Outils et ressources Node.js

QuickStart: Déployer une application Node.js sur Elastic Beanstalk

Ce QuickStart didacticiel explique le processus de création d'une application Node.js et de son déploiement dans un AWS Elastic Beanstalk environnement.

⚠ Non destiné à une utilisation en production

Les exemples sont destinés uniquement à des fins de démonstration. N'utilisez pas d'exemples d'applications en production.

Sections

- Votre AWS compte
- Conditions préalables
- Étape 1 : Création d'une application Node.js
- Étape 2 : Exécutez votre application localement
- Étape 3 : Déployez votre application Node.js avec l'interface de ligne de commande EB
- Étape 4 : Exécutez votre application sur Elastic Beanstalk
- Étape 5 : nettoyer
- AWS ressources pour votre application
- Étapes suivantes
- Déployez avec la console Elastic Beanstalk

Votre AWS compte

Si vous n'êtes pas encore AWS client, vous devez créer un AWS compte. L'inscription vous permet d'accéder à Elastic Beanstalk AWS et aux autres services dont vous avez besoin.

Si vous avez déjà un AWS compte, vous pouvez passer àConditions préalables.

Créez un AWS compte

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez l'https://portal.aws.amazon.com/billing/inscription.
2. Suivez les instructions en ligne.

Dans le cadre de la procédure d'inscription, vous recevrez un appel téléphonique ou un SMS et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l'utilisateur racine a accès à l'ensemble des Services AWS et des ressources de ce compte. La meilleure pratique de sécurité consiste à attribuer un accès administratif à un utilisateur, et à utiliser uniquement l'utilisateur racine pour effectuer les tâches nécessitant un accès utilisateur racine.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. À tout moment, vous pouvez consulter l'activité actuelle de votre compte et gérer votre compte en accédant à https://aws.amazon.com/et en choisissant Mon compte.

Création d'un utilisateur doté d'un accès administratif

Une fois que vous vous êtes inscrit à un utilisateur administratif Compte AWS, que vous Utilisateur racine d'un compte AWS l'avez sécurisé AWS IAM Identity Center, que vous l'avez activé et que vous en avez créé un, afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l'aide pour vous connecter en utilisant l'utilisateur racine, consultez Connexion en tant qu'utilisateur racine dans le Guide de l'utilisateur Connexion à AWS .
2. Activez l'authentification multifactorielle (MFA) pour votre utilisateur racine.

   Pour obtenir des instructions, consultez la section Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d'un utilisateur doté d'un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d' AWS IAM Identity Center dans le Guide de l'utilisateur AWS IAM Identity Center .

2. Dans IAM Identity Center, octroyez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connexion en tant qu'utilisateur doté d'un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l'URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l'utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribution d'un accès à d'autres utilisateurs

1. Dans IAM Identity Center, créez un ensemble d'autorisations qui respecte la bonne pratique consistant à appliquer les autorisations de moindre privilège.

   Pour obtenir des instructions, consultez Création d'un ensemble d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .
2. Attribuez des utilisateurs à un groupe, puis attribuez un accès par authentification unique au groupe.

   Pour obtenir des instructions, consultez Ajout de groupes dans le Guide de l'utilisateur AWS IAM Identity Center .

Conditions préalables

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

INTERFACE DE LIGNE DE COMMANDE (CLI) EB

Ce tutoriel utilise également l'interface de ligne de commande Elastic Beanstalk (CLI EB). Pour de plus amples informations sur l'installation et la configuration de la CLI EB, veuillez consulter Installation de la CLI EB avec un script de configuration (recommandé) et Configuration de l'interface de ligne de commande EB.

Node.js

Installez Node.js sur votre ordinateur local en suivant la procédure d'installation de Node.js sur le site Web Node.js.

Vérifiez votre installation de Node.js en exécutant la commande suivante.

```
~$ node -v
```

Étape 1 : Création d'une application Node.js

Créez un répertoire de projet.

```
~$ mkdir eb-nodejs
~$ cd eb-nodejs
```

Créez ensuite une application qui vous allez déployer à l'aide d'Elastic Beanstalk. Nous allons créer un service RESTful Web « Hello World ».

Example~/eb-nodejs/server.js

```
const http = require('node:http');

const hostname = '127.0.0.1';
const port = 8080;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
```

```
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello Elastic Beanstalk!\n');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Cette application ouvre un écouteur sur le port 8080. Elastic Beanstalk transmet les demandes à votre application sur le port 8080 par défaut pour Node.js.

Étape 2 : Exécutez votre application localement

Exécutez la commande suivante pour exécuter votre application localement.

```
~/eb-nodejs$ node server.js
```

Vous devriez voir le texte suivant.

```
Server running at http://127.0.0.1:8080/
```

Entrez l'adresse URL `http://127.0.0.1:8080/` dans votre navigateur Web. Le navigateur doit afficher « Hello Elastic Beanstalk ! ».

Étape 3 : Déployez votre application Node.js avec l'interface de ligne de commande EB

Exécutez les commandes suivantes pour créer un environnement Elastic Beanstalk pour cette application.

Pour créer un environnement et déployer votre application Node.js

1. Initialisez votre référentiel de la CLI EB avec la commande eb init :

   ```
   ~/eb-nodejs$ eb init -p node.js nodejs-tutorial --region us-east-2
   ```

   Cette commande crée une application nommée `nodejs-tutorial` et configure votre référentiel local pour créer des environnements avec la dernière version de la plateforme Node.js.

2. (Facultatif) Exécutez à eb init nouveau pour configurer une paire de clés par défaut afin de pouvoir utiliser SSH pour vous connecter à l' EC2 instance qui exécute votre application.

```
~/eb-nodejs$ eb init
Do you want to set up SSH for your instances?
(y/n): y
Select a keypair.
1) my-keypair
2) [ Create new KeyPair ]
```

   Sélectionnez une paire de clés si vous en avez déjà une, ou suivez les invites pour en créer une. Si vous ne voyez pas l'invite ou que vous avez besoin de modifier vos paramètres ultérieurement, exécutez eb init -i.

3. Créez un environnement et déployez-y votre application avec eb create. Elastic Beanstalk crée automatiquement un fichier zip pour votre application et le déploie sur une instance de l' EC2 environnement. Après avoir déployé votre application, Elastic Beanstalk la démarre sur le port 8080.

```
~/eb-nodejs$ eb create nodejs-env
```

   Il faut environ cinq minutes à Elastic Beanstalk pour créer votre environnement.

Étape 4 : Exécutez votre application sur Elastic Beanstalk

Lorsque le processus de création de votre environnement est terminé, ouvrez votre site Web aveceb open.

```
~/eb-nodejs$ eb open
```

Félicitations ! Vous avez déployé une application Node.js avec Elastic Beanstalk ! Celle-ci ouvre une fenêtre de navigation en utilisant le nom de domaine créé pour votre application.

Étape 5 : nettoyer

Vous pouvez mettre fin à votre environnement lorsque vous avez fini d'utiliser votre application. Elastic Beanstalk AWS met fin à toutes les ressources associées à votre environnement.

Pour terminer votre environnement Elastic Beanstalk avec l'EB CLI, exécutez la commande suivante.

```
~/eb-nodejs$ eb terminate
```

AWS ressources pour votre application

Vous venez de créer une application à instance unique. Il s'agit d'un exemple d'application simple avec une seule EC2 instance, il ne nécessite donc pas d'équilibrage de charge ni de dimensionnement automatique. Pour les applications à instance unique, Elastic Beanstalk crée les ressources suivantes : AWS

- EC2 instance — Une machine EC2 virtuelle Amazon configurée pour exécuter des applications Web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble distinct de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse qui traite le trafic web devant votre application web, lui transmet les demandes, traite les ressources statiques et génère des journaux d'accès et d'erreurs.
- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

Elastic Beanstalk gère toutes ces ressources. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

Étapes suivantes

Dès que vous disposez d'un environnement exécutant une application, vous pouvez à tout moment déployer une nouvelle version de l'application ou une application totalement différente. Le déploiement d'une nouvelle version d'application est très rapide car il ne nécessite ni le provisionnement ni le redémarrage des instances. EC2 Vous pouvez également explorer votre nouvel environnement à l'aide de la console Elastic Beanstalk. Pour connaître les étapes détaillées, consultez la section Explorez votre environnement dans le chapitre Mise en route de ce guide.

> **Essayez d'autres didacticiels**
>
> Si vous souhaitez essayer d'autres didacticiels avec différents exemples d'applications, consultezAutres exemples d'applications et de didacticiels Elastic Beanstalk pour Node.js.

Une fois que vous avez déployé un ou deux exemples d'applications et que vous êtes prêt à commencer à développer et à exécuter des applications Node.js localement, consultezConfiguration de votre environnement de développement Node.js pour Elastic Beanstalk.

Déployez avec la console Elastic Beanstalk

Vous pouvez également utiliser la console Elastic Beanstalk pour lancer l'exemple d'application. Pour connaître les étapes détaillées, voir Création d'un exemple d'application dans le chapitre Mise en route de ce guide.

Configuration de votre environnement de développement Node.js pour Elastic Beanstalk

Cette rubrique fournit des instructions pour configurer un environnement de développement Node.js afin de tester votre application localement avant de la déployer AWS Elastic Beanstalk. Il fait également référence à des sites Web qui fournissent des instructions d'installation pour des outils utiles.

Rubriques

- Installation de Node.js.
- Confirmation de l'installation de npm

- Installez le AWS SDK pour Node.js
- Installation du générateur Express
- Configuration d'un cadre et d'un serveur Express

Installation de Node.js.

Installez Node.js pour exécuter des applications Node.js localement. Si vous n'avez pas de préférence, téléchargez la dernière version prise en charge par Elastic Beanstalk. Pour obtenir la liste des versions prises en charge, accédez à Node.js dans le document Plateformes AWS Elastic Beanstalk .

Téléchargez Node.js sur nodejs.org.

Confirmation de l'installation de npm

Node.js utilise le gestionnaire de package npm pour vous aider à installer les outils et les infrastructures à utiliser dans votre application. Comme npm est distribué avec Node.js, vous l'installez automatiquement lorsque vous téléchargez et installez Node.js. Pour confirmer que npm est installé, vous pouvez exécuter la commande suivante :

```
$ npm -v
```

Pour plus d'informations sur npm, visitez le site Web de npmjs.

Installez le AWS SDK pour Node.js

Si vous devez gérer AWS des ressources depuis votre application, installez le AWS SDK pour JavaScript dans Node.js. Installez le SDK avec npm :

```
$ npm install aws-sdk
```

Consultez la page d'accueil du AWS SDK pour JavaScript in Node.js pour plus d'informations.

Installation du générateur Express

Express est une infrastructure d'application web qui s'exécute sur Node.js. Pour l'utiliser, installez d'abord l'application de ligne de commande du générateur Express. Une fois le générateur Express

installé, vous pouvez exécuter la commande express pour générer une structure de projet de base pour votre application web. Une fois le projet de base, les fichiers et les dépendances installés, vous pouvez démarrer un serveur Express local sur votre ordinateur de développement.

> **Note**
>
> - Ces étapes vous guident dans l'installation du générateur Express sur un système d'exploitation Linux.
> - Pour Linux, en fonction de votre niveau d'autorisation aux répertoires du système, il se peut que vous ayez besoin de préfixer certaines de ces commandes `sudo`.

Pour installer le générateur Express dans votre environnement de développement

1. Créez un répertoire de travail pour votre cadre et votre serveur Express.

   ```
   ~$ mkdir node-express
   ~$ cd node-express
   ```

2. Installez Express globalement afin de pouvoir accéder à la commande `express`.

   ```
   ~/node-express$ npm install -g express-generator
   ```

3. En fonction de votre système d'exploitation, vous pouvez avoir besoin de définir le chemin d'accès pour exécuter la commande `express`. La sortie de l'étape précédente vous indique si vous devez définir votre variable de chemin d'accès. Voici un exemple pour Linux.

   ```
   ~/node-express$ export PATH=$PATH:/usr/local/share/npm/bin/express
   ```

   Lorsque vous suivrez les tutoriels de ce chapitre, vous devrez exécuter la commande express à partir de différents répertoires. Chaque tutoriel définit une structure de projet Express de base dans son propre répertoire.

Vous avez maintenant installé le générateur de ligne de commande Express. Vous pouvez l'utiliser pour créer un répertoire de cadre pour votre application web, configurer les dépendances et démarrer le serveur de l'application web. Nous allons maintenant passer en revue les étapes à suivre pour y parvenir dans le répertoire `node-express` que nous avons créé.

Configuration d'un cadre et d'un serveur Express

Suivez ces étapes pour créer les répertoires et le contenu du cadre Express de base. Les tutoriels de ce chapitre comprennent également ces étapes pour configurer le cadre Express de base dans chacun des répertoires d'application du tutoriel.

Pour configurer un cadre et un serveur Express

1. Exécutez la commande `express`. Il en résulte la génération de `package.json`, `app.js`, et de quelques répertoires.

   ```
   ~/node-express$ express
   ```

   Lorsque vous êtes invité à continuer, tapez **y**.

2. Configurez les dépendances locales.

   ```
   ~/node-express$ npm install
   ```

3. Vérifiez que le serveur de l'application web démarre.

   ```
   ~/node-express$ npm start
   ```

   Vous devez voir des résultats similaires à ce qui suit :

   ```
   > nodejs@0.0.0 start /home/local/user/node-express
   > node ./bin/www
   ```

   Par défaut, le serveur s'exécute sur le port 3000. Pour le tester, exécutez `curl http://localhost:3000` sur un autre terminal, ou ouvrez un navigateur sur l'ordinateur local et entrez l'adresse URL `http://localhost:3000`.

   Appuyez sur Ctrl+C afin d'arrêter le serveur.

Utilisation de la plateforme Elastic Beanstalk Node.js

Cette rubrique explique comment configurer, créer et exécuter vos applications Node.js sur Elastic Beanstalk.

AWS Elastic Beanstalk prend en charge un certain nombre de branches de plate-forme pour différentes versions du langage de programmation Node.js. Voir Node.js dans le document AWS Elastic Beanstalk Platforms pour une liste complète.

Elastic Beanstalk fournit des options de configuration que vous pouvez utiliser pour personnaliser le logiciel qui s'exécute sur des instances EC2 dans votre environnement Elastic Beanstalk. Vous pouvez configurer des variables d'environnement requises pour votre application, activer la rotation des journaux sur Amazon S3 et mapper des dossiers dans la source de votre application contenant des fichiers statiques vers des chemins desservis par le serveur proxy.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour modifier la configuration d'un environnement en cours d'exécution. Pour éviter de perdre la configuration de votre environnement en le résiliant, vous pouvez utiliser des configurations enregistrées pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des fichiers de configuration. Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

Vous pouvez inclure un fichier `Package.json` dans votre offre groupée source pour installer des packages pendant le déploiement, fournir une commande de démarrage et spécifier la version Node.js que vous souhaitez que votre application utilise. Vous pouvez inclure un fichier `npm-shrinkwrap.json` pour verrouiller les versions de dépendances.

La plateforme Node.js inclut un serveur proxy qui diffuse les ressources statiques, achemine le trafic vers votre application et compresse les réponses. Vous pouvez étendre ou remplacer la configuration du serveur proxy par défaut pour les scénarios avancés.

Il existe plusieurs options pour démarrer votre application. Vous pouvez ajouter un Procfile à votre offre groupée source pour spécifier la commande qui démarre votre application. Lorsque vous ne fournissez pas de `Procfile` mais un fichier `package.json`, Elastic Beanstalk exécute `npm start`. Si vous n'en fournissez pas non plus, Elastic Beanstalk recherche le fichier `app.js` ou `server.js`, dans cet ordre, et exécute le script.

Les paramètres appliqués dans la console Elastic Beanstalk remplacent les mêmes paramètres des fichiers de configuration, s'ils existent. Cela vous permet d'utiliser les paramètres par défaut dans les fichiers de configuration et de les remplacer par des paramètres spécifiques à l'environnement

dans la console. Pour plus d'informations sur la priorité et les autres méthodes de modification des paramètres, consultez Options de configuration.

Pour plus d'informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, consultez the section called “Extension des plateformes Linux”.

Configuration de votre environnement Node.js

Vous pouvez utiliser les paramètres de la plateforme Node.js pour affiner le comportement de vos instances Amazon EC2. Vous pouvez modifier la configuration des instances Amazon EC2 de votre environnement Elastic Beanstalk à l'aide de la console Elastic Beanstalk.

Utilisez la console Elastic Beanstalk pour permettre la rotation des journaux sur Amazon S3 et configurer des variables que votre application peut lire à partir de l'environnement.

Pour configurer votre environnement Node.js dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.

Options de conteneur

Vous pouvez spécifier ces options spécifiques à la plateforme :

- Proxy server (Serveur proxy) – Serveur proxy à utiliser sur vos instances d'environnement. Par défaut, NGINX est utilisé.

Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.

- Enable log file rotation to Amazon S3 (Permettre la rotation du fichier journal sur Amazon S3) – Indique si les fichiers journaux des instances Amazon EC2 de votre application doivent être copiés dans le compartiment Amazon S3 associé à votre application.

Fichiers statiques

Pour améliorer les performances, la section des Fichiers statiques vous permet de configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Pour chaque répertoire, vous définissez le chemin virtuel sur le mappage de répertoires. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Pour en savoir plus sur la configuration des fichiers statiques à l'aide des fichiers de configuration ou de la console Elastic Beanstalk, consultez the section called “Fichiers statiques”.

Propriétés de l'environnement

Utilisez la section Environment Properties pour spécifier des paramètres de configuration de l'environnement sur les instances Amazon EC2 exécutant votre application. Ces paramètres sont passés en tant que paires clé-valeur à l'application.

Dans l'Node.jsenvironnement qui s'exécute AWS Elastic Beanstalk, vous pouvez accéder aux variables d'environnement en exécutant`process.env.ENV_VARIABLE`.

```
var endpoint = process.env.API_ENDPOINT
```

La plateforme Node.js définit la variable d'environnement PORT sur le port vers lequel le serveur proxy transfère le trafic. Pour de plus amples informations, veuillez consulter Configuration du serveur proxy.

Pour plus d’informations, consultez Variables d'environnement et autres paramètres du logiciel.

Configuration d'un environnement Node.js Amazon Linux AMI (antérieure à Amazon Linux 2)

Les catégories de configuration logicielles de console suivantes sont prises en charge uniquement sur un environnement Elastic Beanstalk Node.js qui utilise une version de plateforme Amazon Linux AMI (antérieure à Amazon Linux 2).

Remarques

- Les informations de cette rubrique s'appliquent uniquement aux branches de plate-forme basées sur l'AMI Amazon Linux (AL1). AL2023Les branches/AL2 platform sont incompatibles avec les versions précédentes de la plateforme Amazon Linux AMI (AL1) et nécessitent des paramètres de configuration différents.
- Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Options de conteneur — AMI Amazon Linux (AL1)

Sur la page de configuration, spécifiez les informations suivantes :

- Proxy serveur (Serveur proxy) – Indique le serveur web à utiliser comme proxy pour se connecter à Node.js. Par défaut, NGINX est utilisé. Si vous sélectionnez none (aucun), les mappages de fichiers statiques ne prendront pas effet et la compression GZIP sera désactivée.
- version Node.js : spécifie la version de Node.js. Pour obtenir la liste des versions Node.js prises en charge, consultez Node.js dans le guide des plateformes AWS Elastic Beanstalk .
- GZIP compression (Compression) : indique si la compression GZIP est activée. La compression GZIP est activée par défaut.
- Node command (Commande Node) : vous permet de saisir la commande utilisée pour démarrer l'application Node.js. Une chaîne vide (par défaut) signifie qu'Elastic Beanstalk utilise`app.js`, puis `server.js`et ensuite `npm start`.

Espaces de noms de la configuration Node.js

Vous pouvez utiliser un fichier de configuration pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être spécifiques à la plate-forme ou s'appliquer à toutes les plateformes du service Elastic Beanstalk dans son ensemble. Les options de configuration sont organisées en espaces de noms.

Vous pouvez choisir le proxy à utiliser sur les instances de votre environnement à l'aide de l'espace de noms `aws:elasticbeanstalk:environment:proxy`. L'exemple suivant configure votre environnement pour qu'il utilise le serveur proxy Apache HTTPD.

Example.ebextensions/nodejs-settings.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache
```

Vous pouvez configurer le proxy pour qu'il traite les fichiers statiques à l'aide de l'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles`. Pour plus d'informations et pour voir un exemple, consultez the section called “Fichiers statiques”.

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d' AWS CLI. Pour plus d’informations, consultez Options de configuration.

Plateforme Node.js d'Amazon Linux AMI (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Node.js utilise une version de plateforme Amazon Linux AMI (antérieure à Amazon Linux 2), tenez compte des configurations et recommandations spécifiques de cette section.

> **Remarques**
>
> - Les informations de cette rubrique s'appliquent uniquement aux branches de plate-forme basées sur l'AMI Amazon Linux (AL1). AL2023Les branches/AL2 platform sont incompatibles avec les versions précédentes de la plateforme Amazon Linux AMI (AL1) et nécessitent des paramètres de configuration différents.
> - Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Node.jsoptions de configuration spécifiques à la plate-forme — AMI Amazon Linux () AL1

Elastic Beanstalk prend en charge certaines options de configuration spécifiques à la plateforme pour les versions de plateforme Node.js d'Amazon Linux AMI. Vous pouvez choisir le serveur proxy à exécuter devant votre application, une version spécifique de Node.js à exécuter et la commande utilisée pour exécuter votre application.

Pour le serveur proxy, vous pouvez utiliser un serveur proxy NGINX ou Apache. Vous pouvez définir la valeur none sur l'option `ProxyServer`. Dans ce cas, Elastic Beanstalk exécute votre application en mode autonome, pas derrière un serveur proxy. Si votre environnement exécute une application autonome, mettez à jour votre code pour écouter le port vers lequel NGINX transfère le trafic.

```
var port = process.env.PORT || 8080;

app.listen(port, function() {
  console.log('Server running at http://127.0.0.1:%s', port);
});
```

Node.jsversions linguistiques — Amazon Linux AMI (AL1)

En termes de version linguistique prise en charge, la plateforme Amazon Linux AMI Node.js est différente des autres plateformes gérées par Elastic Beanstalk. En effet, chaque version de la plateforme Node.js ne prend en charge que quelques versions linguistiques Node.js. Pour obtenir la liste des versions Node.js prises en charge, consultez Node.js dans le guide Plateformes AWS Elastic Beanstalk .

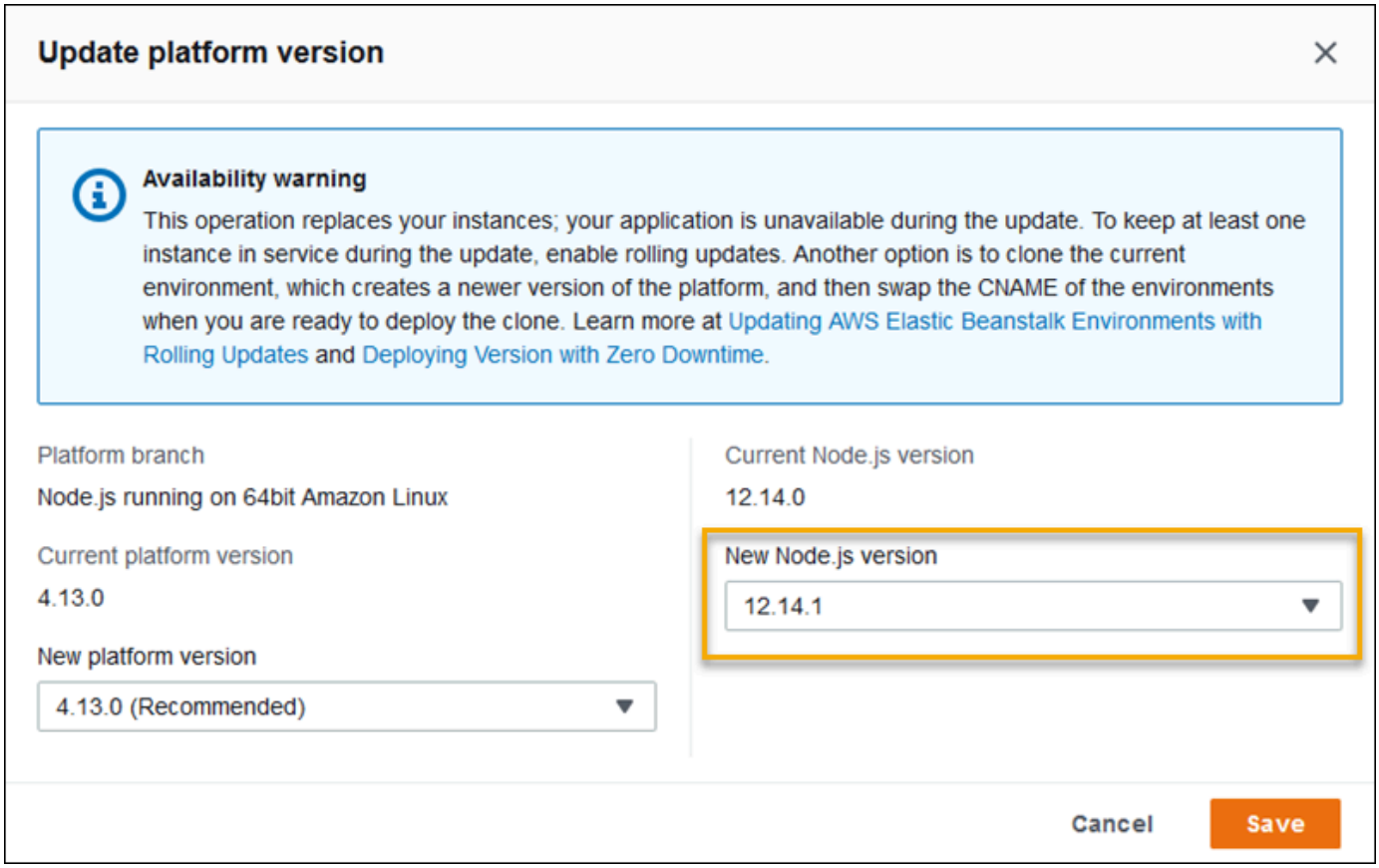
Vous pouvez utiliser une option de configuration spécifique à la plateforme pour définir la version de langage. Pour obtenir des instructions, veuillez consulter the section called "Configuration de votre environnement Node.js". Vous pouvez également utiliser la console Elastic Beanstalk pour mettre à jour la version Node.js que votre environnement utilise dans le cadre de la mise à jour de la version de votre plateforme .

> **Note**
>
> Lorsque la prise en charge de la version de Node.js que vous utilisez est supprimée de la plateforme, vous devez modifier ou supprimer le paramètre de version avant de procéder à une mise à jour de la plateforme. Cela peut se produire lorsqu'une faille de sécurité est identifiée pour une ou plusieurs versions de Node.js
> Dans ce cas, la tentative de mise à jour vers une nouvelle version de la plate-forme qui ne prend pas en charge la configuration NodeVersionéchoue. Pour éviter d'avoir à créer un

nouvel environnement, remplacez l'option de NodeVersionconfiguration par une version de Node.js prise en charge à la fois par l'ancienne version de plate-forme et par la nouvelle, ou supprimez le paramètre d'option, puis effectuez la mise à jour de la plate-forme.

Pour configurer la version Node.js de votre environnement dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans la page de présentation de l'environnement, sous Platform (Plateforme), choisissez Change (Changer).
4. Dans la boîte de dialogue Update platform version (Mettre à jour la version de la plateforme) sélectionnez une version Node.js.



5. Choisissez Enregistrer.

Node.jsespaces de noms de configuration — AMI Amazon Linux () AL1

La plateforme Amazon Linux AMI Node.js définit des options supplémentaire dans les espaces de noms `aws:elasticbeanstalk:container:nodejs:staticfiles` et `aws:elasticbeanstalk:container:nodejs`.

Le fichier de configuration suivant indique à Elastic Beanstalk d'utiliser `npm start` pour exécuter l'application. Il définit également le type de proxy sur Apache et active la compression. Enfin, il configure le proxy pour qu'il serve des fichiers statiques à partir de deux répertoires sources. Une source est constituée de fichiers HTML situés sur le chemin `html` sous la racine du site web à partir du répertoire source `statichtml`. L'autre source est constituée de fichiers image situés sur le chemin `images` sous la racine du site web à partir du répertoire source `staticimages`.

Example.ebextensions/node-settings.config

```
option_settings:
  aws:elasticbeanstalk:container:nodejs:
    NodeCommand: "npm start"
    ProxyServer: apache
    GzipCompression: true
  aws:elasticbeanstalk:container:nodejs:staticfiles:
    /html: statichtml
    /images: staticimages
```

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d' AWS CLI. Pour plus d’informations, consultez Options de configuration.

Configuration de commandes de démarrage personnalisées avec un Procfile Elastic Beanstalk

Vous pouvez inclure un fichier appelé `Procfile` à la racine de votre offre groupée source pour spécifier la commande qui démarre votre application.

Example Procfile

```
web: node index.js
```

Pour plus d'informations sur `Procfile` l'utilisation, voirBuildfile et Procfile.

> **Note**
>
> Cette fonctionnalité remplace l'option héritée `NodeCommand` dans l'espace de noms `aws:elasticbeanstalk:container:nodejs`.

Configuration des dépendances de votre application sur Elastic Beanstalk

Votre application peut avoir des dépendances sur certains modules Node.js, comme ceux que vous spécifiez dans les instructions `require()`. Ces modules sont stockés dans un répertoire `node_modules`. Lorsque votre application s'exécute, Node.js charge les modules depuis ce répertoire. Pour en savoir plus, consultez Loading from node_modules folders (Chargement depuis les dossiers node_modules) dans la documentation Node.js.

Vous pouvez spécifier ces dépendances de module à l'aide d'un fichier `package.json`. Si Elastic Beanstalk détecte ce fichier et qu'aucun répertoire `node_modules` n'est présent, Elastic Beanstalk exécute `npm install` en tant qu'utilisateur webapp. La commande `npm install` installe les dépendances dans le répertoire `node_modules` créé au préalable par Elastic Beanstalk. La commande `npm install` accède aux packages répertoriés dans le fichier `package.json` à partir du registre public npm ou d'autres emplacements. Pour en savoir plus, consultez le site Web npm Docs.

Si Elastic Beanstalk détecte le répertoire `node_modules`, Elastic Beanstalk ne s'exécute pas `npm install`, même si un fichier `package.json` existe. Elastic Beanstalk part du principe que les packages de dépendance sont disponibles dans le répertoire`node_modules` pour Node.js à des fins d'accès et de chargement.

Les sections suivantes fournissent des informations supplémentaires sur la définition des dépendances de modules Node.js pour votre application.

> **Note**
>
> Si vous rencontrez des problèmes de déploiement lors de l'exécution d'Elastic Beanstalk `npm install`, envisagez une autre approche. Incluez le répertoire `node_modules` avec les modules de dépendance dans l'offre groupée source de votre application. Cela peut contourner les problèmes liés à l'installation de dépendances à partir du registre public npm pendant que vous étudiez le problème. Comme les modules de dépendance proviennent d'un répertoire local, cela peut également contribuer à réduire le temps de déploiement. Pour de

plus amples informations, consultez Inclusion des dépendances Node.js dans un répertoire node_modules.

Spécification des dépendances Node.js avec un fichier package.json

Utilisez un fichier `package.json` à la racine de votre source de projet pour spécifier des packages de dépendance et pour fournir une commande start. Lorsqu'un fichier `package.json` est présent et qu'aucun répertoire `node_modules` n'est présent à la racine de la source de votre projet, Elastic Beanstalk exécute `npm install` en tant qu'utilisateur webapp pour installer les dépendances à partir du registre public npm. Il utilise également la commande `start` pour démarrer votre application. Pour en savoir plus sur le fichier `package.json`, consultez la section Définition des dépendances dans un fichier `package.json` sur le site Web de npm Docs.

Utilisez le mot-clé `scripts` pour fournir une commande start. Le mot-clé `scripts` est maintenant utilisé à la place de l'option héritée `NodeCommand` dans l'espace de noms `aws:elasticbeanstalk:container:nodejs`.

Example package.json – Express

```
{
    "name": "my-app",
    "version": "0.0.1",
    "private": true,
    "dependencies": {
      "ejs": "latest",
      "aws-sdk": "latest",
      "express": "latest",
      "body-parser": "latest"
    },
    "scripts": {
      "start": "node app.js"
    }
  }
```

Mode production et dépendances dev

Pour spécifier vos dépendances dans le fichier `package.json`, utilisez les attributs dependencies et devDependencies. L'attribut dependencies désigne les packages nécessaires à votre application en production. L'attribut devDependencies désigne les packages qui ne sont nécessaires que pour le développement local et les tests.

Si vous devez installer les packages devDependencies, définissez la propriété d'environnement NPM_USE_PRODUCTION sur `false`. Avec ce paramètre, nous n'utiliserons pas les options ci-dessus lors de l'exécution de npm install. Ainsi, les packages devDependencies seront installés.

Inclusion des dépendances Node.js dans un répertoire node_modules

Pour déployer les packages de dépendances sur des instances d'environnement avec votre code d'application, incluez-les dans un répertoire nommé `node_modules` à la racine de la source de votre projet. Pour en savoir plus, consultez la section Téléchargement et installation de packages localement sur le site Web de npm Docs.

Lorsque vous déployez un `node_modules` répertoire dans une version AL2023/AL2 Node.jsplatform, Elastic Beanstalk part du principe que vous fournissez vos propres packages de dépendances et évite d'installer les dépendances spécifiées dans un fichier package.json. Node.jsrecherche les dépendances dans le `node_modules` répertoire. Pour en savoir plus, consultez la section Chargement à partir des dossiers node_modules dans la documentation Node.js.

> **Note**
>
> Si vous rencontrez des problèmes de déploiement lors de l'exécution d'Elastic Beanstalk `npm install`, pensez à utiliser l'approche décrite dans cette rubrique pour contourner le problème pendant que vous étudiez le problème.

Considérations relatives à Node.js sur Amazon Linux 2

Lisez cette section si vous utilisez une branche de plateforme Node.js exécutée sur Amazon Linux 2.

Considérations relatives à Node.js sur Amazon Linux 2

> **Note**
>
> Les informations contenues dans cette rubrique s'appliquent aux branches de Node.js plateforme exécutées sur Amazon Linux 2. Le contenu décrit ici AL2 des fonctionnalités et des comportements spécifiques qui diffèrent de AL2023.

Variations de commande

Les options de commande varient en fonction de la version de npm incluse dans la branche de plateforme Amazon Linux 2 sur laquelle votre application s'exécute.

- npm v6 – Elastic Beanstalk installe les dépendances en mode production par défaut. Il utilise la commande `npm install --production`.
- npm v7 ou ultérieure – Elastic Beanstalk omet les devDependencies. Il utilise la commande `npm install --omit=dev`.

Les deux commandes listées ci-dessus n'installent pas les packages qui sont des devDependencies.

Protocoles SSH et HTTPS pour les dépendances Git

À compter de la sortie de la plateforme Amazon Linux 2 du 7 mars 2023, vous pouvez utiliser les protocoles SSH et HTTPS pour récupérer des packages depuis un référentiel Git. La branche de plateforme Node.js 16 prend en charge les protocoles SSH et HTTPS. La branche Node.js 14 ne prend en charge que le protocole HTTPS.

Example package.json – Node.js 16 prend en charge les protocoles HTTPS et SSH

```
...
"dependencies": {
  "aws-sdk": "https://github.com/aws/aws-sdk-js.git",
  "aws-chime": "git+ssh://git@github.com:aws/amazon-chime-sdk-js.git"
}
```

Versions et plages de versions

Utilisez le mot-clé `engines` du fichier `package.json` pour spécifier la version de Node.js que vous souhaitez que votre application utilise. Vous pouvez également spécifier une plage de versions à l'aide de la notation npm. Pour en savoir plus sur la syntaxe des plages de versions, consultez la section Gestion sémantique des versions à l'aide de npm sur le site web de Node.js. Le mot-clé `engines` du fichier Node.js `package.json` remplace l'option `NodeVersion` héritée dans l'espace de noms `aws:elasticbeanstalk:container:nodejs`.

> **Important**
>
> La fonctionnalité permettant de spécifier des plages de versions n'est pas disponible pour les branches de la plateforme Node.js exécutées sur AL2023. Nous ne prenons en charge qu'une seule version de Node.js au sein d'une branche Node.js spécifique sur AL2023. Si

votre fichier `package.json` spécifie une plage de versions, nous l'ignorerons et utiliserons par défaut la version de la branche de plateforme de Node.js.

Example **`package.json`** - Version Node.js unique

```
{
    ...
    "engines": { "node" : "14.16.0" }
  }
```

Example **`package.json`** - Gamme de versions Node.js

```
{
    ...
    "engines": { "node" : ">=10 <11" }
  }
```

Lorsqu'une plage de versions est indiquée, Elastic Beanstalk installe la dernière version de Node.js à la disposition de la plateforme dans cette plage. Dans cet exemple, la plage indique que la version doit être supérieure ou égale à la version 10, mais inférieure à la version 11. Par conséquent, Elastic Beanstalk installe la dernière version Node.js de 10.x.y, disponible sur la plateforme prise en charge.

Sachez que vous ne pouvez spécifier qu'une version de Node.js qui correspond à votre branche de plateforme. Par exemple, si vous utilisez la branche de plateforme Node.js 16, vous ne pouvez spécifier qu'une version 16.x.y de Node.js. Vous pouvez utiliser les options de plage de versions prises en charge par npm pour offrir plus de flexibilité. Pour les versions Node.js valides pour chaque branche de plateforme, consultez Node.js dans le guide Plateformes AWS Elastic Beanstalk .

Note

Lorsque la prise en charge de la version de Node.js que vous utilisez est supprimée de la plateforme, vous devez modifier ou supprimer le paramètre de version de Node.js avant de procéder à une mise à jour de la plateforme. Cela peut se produire lorsqu'une faille de sécurité est identifiée pour une ou plusieurs versions de Node.js
Lorsque cela se produit, toute tentative de mise à jour vers une nouvelle version de la plateforme qui ne prend pas en charge la version Node.js configurée échoue. Pour éviter de créer un nouvel environnement, modifiez le paramètre de version de Node.js dans `package.json` sur une version de Node.js qui est prise en charge à la fois par l'ancienne

version de la plateforme et par la nouvelle. Vous avez la possibilité de spécifier une plage de versions Node.js qui inclut une version prise en charge, comme décrit plus haut dans cette rubrique. Vous pouvez également supprimer le paramètre, puis déployer le nouveau bundle source.

Verrouiller les dépendances avec npm shrinkwrap Elastic Beanstalk

La plateforme Node.js exécute `npm install` en tant qu'utilisateur webapp lors de chaque déploiement. Lorsque de nouvelles versions de vos dépendances sont disponibles, elles sont installées lorsque vous déployez l'application, ce qui peut ralentir de manière significative le déploiement.

Pour éviter la mise à jour des dépendances, il est possible de créer un fichier `npm-shrinkwrap.json` qui verrouille les dépendances de votre application sur la version actuelle.

```
$ npm install
$ npm shrinkwrap
wrote npm-shrinkwrap.json
```

Incluez ce fichier dans votre bundle de fichiers source afin que les dépendances ne soient installées qu'à une seule reprise.

Configuration du serveur proxy

Elastic Beanstalk utilise NGINX ou Apache HTTPD comme proxy inverse pour mapper votre application à votre équilibreur de charge Elastic Load Balancing sur le port 80. La valeur par défaut est NGINX. Elastic Beanstalk fournit une configuration de proxy par défaut que vous pouvez étendre ou remplacer totalement par votre propre configuration.

Par défaut, Elastic Beanstalk configure le serveur proxy pour transmettre les demandes à votre application sur le port 5 000. Vous pouvez remplacer le port par défaut en définissant la `PORT`propriété d'environnement sur le port écouté par votre application.

> **Note**
>
> Le port que votre application écoute n'affecte pas le port que le serveur NGINX écoute pour recevoir des demandes de l'équilibreur de charge.

Configuration du serveur proxy sur la version de votre plateforme

Toutes les AL2 plateformes AL2023/prennent en charge une fonctionnalité de configuration de proxy uniforme. Pour plus d'informations sur la configuration du serveur proxy sur les versions de votre plate-forme exécutant AL2023/AL2, consultezConfiguration du proxy inverse.

Configuration du proxy sur l'AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Node.js utilise une version de plateforme Amazon Linux AMI (antérieure à Amazon Linux 2), lisez les informations de cette section.

> **Remarques**
>
> - Les informations de cette rubrique s'appliquent uniquement aux branches de plate-forme basées sur l'AMI Amazon Linux (AL1). AL2023Les branches/AL2 platform sont incompatibles avec les versions précédentes de la plateforme Amazon Linux AMI (AL1) et nécessitent des paramètres de configuration différents.
> - Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Extension et remplacement de la configuration du proxy par défaut — Amazon Linux AMI () AL1

La plateforme Node.js utilise un proxy inverse pour acheminer les requêtes du port 80 de l'instance vers le port 8081 d'écoute de votre application. Elastic Beanstalk fournit une configuration de proxy par défaut que vous pouvez étendre ou remplacer totalement par votre propre configuration.

Pour étendre la configuration par défaut, ajoutez des fichiers `.conf` à `/etc/nginx/conf.d` avec un fichier de configuration. Pour un exemple spécifique, veuillez consulter Suspension des connexions HTTPS sur des instances EC2 exécutant Node.js.

La plateforme Node.js définit la variable d'environnement PORT sur le port vers lequel le serveur proxy transfère le trafic. Lisez cette variable dans votre code pour configurer le port de votre application.

```
    var port = process.env.PORT || 3000;
```

```
    var server = app.listen(port, function () {
        console.log('Server running at http://127.0.0.1:' + port + '/');
    });
```

La configuration NGINX par défaut achemine le trafic vers un serveur en amont nommé `nodejs` à l'adresse `127.0.0.1:8081`. Vous pouvez supprimer la configuration par défaut et fournir une configuration personnalisée dans un fichier de configuration.

Example.ebextensions/proxy.config

L'exemple suivant supprime la configuration par défaut et ajoute une configuration personnalisée qui achemine le trafic vers le port 5000 au lieu du port 8081.

```
files:
  /etc/nginx/conf.d/proxy.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      upstream nodejs {
        server 127.0.0.1:5000;
        keepalive 256;
      }

      server {
        listen 8080;

        if ($time_iso8601 ~ "^(\d{4})-(\d{2})-(\d{2})T(\d{2})") {
            set $year $1;
            set $month $2;
            set $day $3;
            set $hour $4;
        }
        access_log /var/log/nginx/healthd/application.log.$year-$month-$day-$hour
 healthd;
        access_log  /var/log/nginx/access.log  main;

        location / {
            proxy_pass  http://nodejs;
            proxy_set_header   Connection "";
            proxy_http_version 1.1;
            proxy_set_header        Host            $host;
```

```
            proxy_set_header        X-Real-IP       $remote_addr;
            proxy_set_header        X-Forwarded-For $proxy_add_x_forwarded_for;
        }

        gzip on;
        gzip_comp_level 4;
        gzip_types text/html text/plain text/css application/json application/x-
javascript text/xml application/xml application/xml+rss text/javascript;

        location /static {
            alias /var/app/current/static;
        }

      }

  /opt/elasticbeanstalk/hooks/configdeploy/post/99_kill_default_nginx.sh:
    mode: "000755"
    owner: root
    group: root
    content: |
      #!/bin/bash -xe
      rm -f /etc/nginx/conf.d/00_elastic_beanstalk_proxy.conf
      service nginx stop
      service nginx start

container_commands:
  removeconfig:
    command: "rm -f /tmp/deployment/config/
#etc#nginx#conf.d#00_elastic_beanstalk_proxy.conf /etc/nginx/
conf.d/00_elastic_beanstalk_proxy.conf"
```

L'exemple de configuration (`/etc/nginx/conf.d/proxy.conf`) utilise la configuration par défaut dans `/etc/nginx/conf.d/00_elastic_beanstalk_proxy.conf` comme base pour inclure le bloc de serveur par défaut avec des paramètres de compression et de journalisation et un mappage de fichiers statiques.

La commande `removeconfig` supprime la configuration par défaut du conteneur pour que le serveur proxy utilise bien la configuration personnalisée. Elastic Beanstalk recrée la configuration par défaut lorsque chaque configuration est déployée. Pour tenir compte de cela, dans l'exemple suivant, un post-configuration-deployment hook (`/opt/elasticbeanstalk/hooks/configdeploy/post/99_kill_default_nginx.sh`) est ajouté. Celui-ci supprime la configuration par défaut et redémarre le serveur proxy.

> **Note**
>
> Il est possible que la configuration par défaut change dans les prochaines versions de la plateforme Node.js. Utilisez la version la plus récente de la configuration comme base pour vos personnalisations afin de garantir leur compatibilité.

Si vous remplacez la configuration par défaut, vous devez définir les mappages de fichiers statiques et la compression GZIP. Cela s'explique par le fait que la plateforme ne peut pas appliquer les paramètres standard.

Autres exemples d'applications et de didacticiels Elastic Beanstalk pour Node.js

Cette section fournit des applications et des didacticiels supplémentaires. La QuickStart pour Node.js rubrique présentée précédemment dans cette rubrique vous explique comment lancer l'exemple d'application Node.js avec l'interface de ligne de commande EB.

Pour commencer à utiliser les applications Node.js AWS Elastic Beanstalk, il vous suffit d'un bundle de sources d'applications à télécharger en tant que première version de l'application et à déployer dans un environnement.

Lancement d'un environnement avec un exemple d'application Node.js

Elastic Beanstalk fournit des exemples d'applications d'une seule page pour chaque plate-forme, ainsi que des exemples plus complexes illustrant l'utilisation de ressources AWS supplémentaires telles qu'Amazon RDS et des fonctionnalités spécifiques au langage ou à la plate-forme et. APIs

> **Note**
>
> Suivez les étapes du fichier de la solution groupée source `README.md` pour le déployer.

Exemples

Type d'environnement	Solution groupée source	Description
Serveur web	nodejs.zip	Application avec page unique. Pour lancer l'exemple d'application avec l'interface de ligne de commande EB, voirQuickStart pour Node.js. Vous pouvez également utiliser la console Elastic Beanstalk pour lancer l'exemple d'application. Pour connaître les étapes détaillées, voir Création d'un exemple d'application dans le chapitre Mise en route de ce guide.
Serveur web avec Amazon RDS	nodejs-example-express-rds.zip	Application de journal de randonnée qui utilise le cadre Express et un service Amazon Relational Database Service (RDS). Didacticiel
Serveur Web avec Amazon ElastiCache	nodejs-example-express-elasticache.zip	Application Web Express qui utilise Amazon ElastiCache pour le clustering. La mise en cluster permet d'améliorer la haute disponibilité, les performances et la sécurité de votre application web. Didacticiel
Serveur web avec DynamoDB, Amazon SNS et Amazon SQS	nodejs-example-dynamo.zip	Site web Express qui collecte les informations de contact des utilisateurs pour la nouvelle campagne de marketing d'une entreprise. Utilise le AWS SDK pour JavaScript in Node.js pour écrire des entrées dans une table DynamoDB, et les fichiers de configuration Elastic Beanstalk pour créer des ressources dans DynamoDB, Amazon SNS et Amazon SQS.

Type d'environnement	Solution groupée source	Description
		Didacticiel

Étapes suivantes

Une fois que vous disposez d'un environnement exécutant une application, vous pouvez déployer une nouvelle version de l'application ou une application totalement différente à tout moment. Le déploiement d'une nouvelle version d'application est très rapide, car il n'est pas nécessaire de mettre en service ni de redémarrer les instances EC2. Pour plus d'informations sur le déploiement d'applications, voir Déploiement d'une nouvelle version de votre application.

Une fois que vous avez déployé un ou deux exemples d'applications et que vous êtes prêt à commencer à développer et à exécuter des applications Node.js localement, consultez la section relative Configuration de votre environnement de développement Node.js pour Elastic Beanstalk à la configuration d'un environnement de développement Node.js avec tous les outils dont vous aurez besoin.

Déploiement d'une application Node.js Express sur Elastic Beanstalk

Cette section explique comment déployer un exemple d'application sur Elastic Beanstalk à l'aide de l'interface de ligne de commande Elastic Beanstalk (EB CLI), puis comment mettre à jour l'application pour utiliser le framework Express.

Conditions préalables

Ce tutoriel nécessite les prérequis suivants :

- Les exécutions Node.js
- Le logiciel de gestion des packages Node.js par défaut, npm
- Le générateur de ligne de commande Express
- Interface de ligne de commande Elastic Beanstalk (EB)

Pour plus d'informations sur l'installation des trois premiers composants et la configuration de votre environnement de développement local, consultez Configuration de votre environnement de développement Node.js pour Elastic Beanstalk. Pour ce didacticiel, il n'est pas nécessaire d'installer le AWS SDK pour Node.js, qui est également mentionné dans la rubrique référencée.

Pour plus d'informations sur l'installation et la configuration de l'EB CLI, consultez Installation de la CLI EB avec un script de configuration (recommandé) et Configuration de l'interface de ligne de commande EB.

Créer un environnement Elastic Beanstalk

Votre répertoire d'application

Ce tutoriel utilise un répertoire appelé `nodejs-example-express-rds` pour la création de la solution groupée de l'application. Créez le répertoire `nodejs-example-express-rds` pour ce tutoriel.

```
~$ mkdir nodejs-example-express-rds
```

> **Note**
>
> Chaque tutoriel de ce chapitre utilise son propre répertoire pour la solution groupée de l'application. Le nom du répertoire correspond au nom de l'exemple d'application utilisé par le tutoriel.

Changez votre répertoire de travail actuel vers `nodejs-example-express-rds`.

```
~$ cd nodejs-example-express-rds
```

Configurons maintenant un environnement Elastic Beanstalk exécutant la plateforme Node.js et l'exemple d'application. Nous utiliserons l'interface de ligne de commande Elastic Beanstalk (EB CLI).

Pour configurer un référentiel de l'EB CLI pour votre application et créer un environnement Elastic Beanstalk qui exécute la plateforme Node.js

1. Créez un référentiel à l'aide de la commande eb init.

   ```
   ~/nodejs-example-express-rds$ eb init --platform node.js --region <region>
   ```

Cette commande crée un fichier de configuration dans un dossier nommé `.elasticbeanstalk` qui spécifie les paramètres de création d'environnements pour votre application et crée une application Elastic Beanstalk dont le nom est basé sur le dossier actif.

2. Créez un environnement qui exécute un exemple d'application à l'aide de la commande eb create.

```
~/nodejs-example-express-rds$ eb create --sample nodejs-example-express-rds
```

Cette commande crée un environnement à charge équilibrée avec les paramètres par défaut de la plateforme Node.js et les ressources suivantes :

- EC2 instance — Une machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications Web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.
- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Équilibreur de charge – Équilibreur de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité d'équilibrage de charge : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibreur de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.

- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et qui sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack : Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*. *region*.elasticbeanstalk.com.

> **Sécurité du domaine**
>
> Pour renforcer la sécurité de vos applications Elastic Beanstalk, le domaine elasticbeanstalk.com est enregistré dans la liste des suffixes publics (PSL).
> Si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Elastic Beanstalk, nous vous recommandons d'utiliser des cookies `__Host-` avec un préfixe pour une sécurité accrue. Cette pratique protège votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

3. Une fois l'environnement créé, utilisez la commande eb open pour ouvrir l'URL de l'environnement dans le navigateur par défaut.

```
~/nodejs-example-express-rds$ eb open
```

Vous avez maintenant créé un environnement Node.js Elastic Beanstalk avec un exemple d'application. Vous pouvez le mettre à jour avec votre propre application. Ensuite, nous mettons à jour l'exemple d'application afin d'utiliser l'infrastructure Express.

Mise à jour de l'application pour utiliser Express

Une fois que vous avez créé un environnement avec un exemple d'application, vous pouvez le mettre à jour à l'aide de votre propre application. Dans cette procédure, nous exécutons d'abord les commandes express et npm install pour configurer le cadre Express dans votre répertoire d'application. Nous utilisons ensuite la EB CLI pour mettre à jour votre environnement Elastic Beanstalk avec l'application mise à jour.

Pour mettre à jour votre application afin d'utiliser Express

1. Exécutez la commande `express`. Il en résulte la génération de `package.json`, `app.js`, et de quelques répertoires.

   ```
   ~/nodejs-example-express-rds$ express
   ```

   Lorsque vous êtes invité à continuer, tapez **y**.

   > **Note**
   >
   > Si la commande express ne fonctionne pas, il se peut que vous n'ayez pas installé le générateur de ligne de commande Express, comme décrit dans la section précédente sur les prérequis. Il se peut également que le chemin d'accès au répertoire de votre ordinateur local doive être configuré pour permettre l'exécution de la commande express. Consultez la section Prérequis pour connaître les étapes détaillées de la configuration de votre environnement de développement, afin de pouvoir poursuivre ce tutoriel.

2. Configurez les dépendances locales.

   ```
   ~/nodejs-example-express-rds$ npm install
   ```

3. (Facultatif) Vérifiez que le serveur de l'application web démarre.

   ```
   ~/nodejs-example-express-rds$ npm start
   ```

   Vous devez voir des résultats similaires à ce qui suit :

   ```
   > nodejs@0.0.0 start /home/local/user/node-express
   > node ./bin/www
   ```

   Par défaut, le serveur s'exécute sur le port 3000. Pour le tester, exécutez `curl http://localhost:3000` sur un autre terminal, ou ouvrez un navigateur sur l'ordinateur local et entrez l'adresse URL `http://localhost:3000`.

   Appuyez sur Ctrl+C afin d'arrêter le serveur.

4. Déployez les modifications dans votre environnement Elastic Beanstalk avec la commande eb deploy.

```
~/nodejs-example-express-rds$ eb deploy
```

5. Une fois que votre environnement est prêt (il apparaît en vert), actualisez l'URL pour vérifier si l'opération a fonctionné. Une page web intitulée Welcome to Express doit s'afficher.

Ensuite, nous devons mettre à jour l'application Express afin de traiter les fichiers statiques et d'ajouter une nouvelle page.

Pour configurer les fichiers statiques et ajouter une nouvelle page à votre application Express

1. Ajoutez un deuxième fichier de configuration dans le dossier `.ebextensions` avec le contenu suivant :

   `nodejs-example-express-rds/.ebextensions/staticfiles.config`

   ```
   option_settings:
       aws:elasticbeanstalk:environment:proxy:staticfiles:
           /stylesheets: public/stylesheets
   ```

   Ce paramètre configure le serveur proxy afin qu'il transmette les fichiers du dossier `public` au chemin `/public` de l'application. La distribution de fichiers de manière statique depuis le serveur proxy réduit la charge sur votre application. Pour plus d'informations, consultez Fichiers statiques plus haut dans ce chapitre.
2. (Facultatif) Pour confirmer que les mappages statiques sont correctement configurés, commentez la configuration du mappage statique dans `nodejs-example-express-rds/app.js`. Cela supprime le mappage de l'application node.

   ```
   //  app.use(express.static(path.join(__dirname, 'public')));
   ```

   Les mappages de fichiers statiques dans le fichier `staticfiles.config` de l'étape précédente devraient toujours charger la feuille de style avec succès, même après avoir commenté cette ligne. Pour vérifier que les mappages de fichiers statiques sont chargés via la configuration de fichiers statiques du proxy, plutôt que via l'application express, supprimez les valeurs qui suivent `option_settings:`. Une fois qu'elle a été supprimée de la configuration des fichiers statiques et de l'application node, la feuille de style ne se chargera pas.

N'oubliez pas de réinitialiser le contenu des fichiers `nodejs-example-express-rds/app.js` et `staticfiles.config` lorsque vous avez terminé vos tests.

3. Addition `nodejs-example-express-rds/routes/hike.js`. Saisissez les données ci-dessous :

```
exports.index = function(req, res) {
 res.render('hike', {title: 'My Hiking Log'});
};

exports.add_hike = function(req, res) {
};
```

4. Mettez à jour `nodejs-example-express-rds/app.js` en y incluant trois nouvelles lignes.

   Commencez par ajouter la ligne suivante afin d'ajouter un `require` pour cette route :

```
var hike = require('./routes/hike');
```

   Votre fichier doit être similaire à l'extrait suivant :

```
var express = require('express');
var path = require('path');
var hike = require('./routes/hike');
```

   Ajoutez ensuite les deux lignes suivantes au fichier `nodejs-example-express-rds/app.js` après `var app = express();`

```
app.get('/hikes', hike.index);
app.post('/add_hike', hike.add_hike);
```

   Votre fichier doit être similaire à l'extrait suivant :

```
var app = express();
app.get('/hikes', hike.index);
app.post('/add_hike', hike.add_hike);
```

5. Copiez `nodejs-example-express-rds/views/index.jade` dans `nodejs-example-express-rds/views/hike.jade`.

```
~/nodejs-example-express-rds$ cp views/index.jade views/hike.jade
```

6. Déployez les modifications avec la commande eb deploy.

```
~/nodejs-example-express-rds$ eb deploy
```

7. Votre environnement est mis à jour au bout de quelques minutes. Une fois que votre environnement est prêt (il apparaît en vert), assurez-vous qu'il fonctionne. Pour ce faire, actualisez la page de votre navigateur et ajoutez **`hikes`** à la fin de l'URL (par exemple, `http://node-express-env-syypntcz2q.elasticbeanstalk.com/hikes`).

   Vous devriez voir une page web intitulée My Hiking Log.

Vous avez maintenant créé une application web qui utilise le cadre Express. Dans la section suivante, nous allons modifier l'application pour qu'elle utilise un service de base de données relationnelle Amazon (RDS) afin de stocker un journal de randonnée.

Mise à jour de l'application pour utiliser Amazon RDS

Dans l'étape suivante, nous mettons à jour l'application pour qu'elle utilise Amazon RDS for MySQL.

Pour mettre à jour votre application afin d'utiliser RDS for MySQL

1. Pour créer une base de données RDS for MySQL associée à votre environnement Elastic Beanstalk, suivez les instructions de la rubrique Ajout d'une base de données incluse plus loin dans ce chapitre. L'ajout d'une instance de base de données prend environ 10 minutes.
2. Mettez à jour la section des dépendances dans `package.json` avec le contenu suivant :

```
"dependencies": {
    "async": "^3.2.4",
    "express": "4.18.2",
    "jade": "1.11.0",
    "mysql": "2.18.1",
    "node-uuid": "^1.4.8",
    "body-parser": "^1.20.1",
    "method-override": "^3.0.0",
    "morgan": "^1.10.0",
    "errorhandler": "^1.5.1"
  }
```

3. Exécutez npm install.

```
~/nodejs-example-express-rds$ npm install
```

4. Mettez à jour `app.js` pour vous connecter à la base de données, créer une table et insérer un journal de randonnée unique par défaut. Chaque fois que cette application sera déployée, elle supprimera la table des randonnées précédente et la recréera.

```
/**
 * Module dependencies.
 */

 const express = require('express')
 , routes = require('./routes')
 , hike = require('./routes/hike')
 , http = require('http')
 , path = require('path')
 , mysql = require('mysql')
 , async = require('async')
 , bodyParser = require('body-parser')
 , methodOverride = require('method-override')
 , morgan = require('morgan')
 , errorhandler = require('errorhandler');

const { connect } = require('http2');

const app = express()

app.set('views', __dirname + '/views')
app.set('view engine', 'jade')
app.use(methodOverride())
app.use(bodyParser.json())
app.use(bodyParser.urlencoded({ extended: true }))
app.use(express.static(path.join(__dirname, 'public')))


app.set('connection', mysql.createConnection({
host: process.env.RDS_HOSTNAME,
user: process.env.RDS_USERNAME,
password: process.env.RDS_PASSWORD,
port: process.env.RDS_PORT}));

function init() {
```

```
 app.get('/', routes.index);
 app.get('/hikes', hike.index);
 app.post('/add_hike', hike.add_hike);
}

const client = app.get('connection');
async.series([
 function connect(callback) {
   client.connect(callback);
   console.log('Connected!');
 },
 function clear(callback) {
   client.query('DROP DATABASE IF EXISTS mynode_db', callback);
 },
 function create_db(callback) {
   client.query('CREATE DATABASE mynode_db', callback);
 },
 function use_db(callback) {
   client.query('USE mynode_db', callback);
 },
 function create_table(callback) {
    client.query('CREATE TABLE HIKES (' +
                         'ID VARCHAR(40), ' +
                         'HIKE_DATE DATE, ' +
                         'NAME VARCHAR(40), ' +
                         'DISTANCE VARCHAR(40), ' +
                         'LOCATION VARCHAR(40), ' +
                         'WEATHER VARCHAR(40), ' +
                         'PRIMARY KEY(ID))', callback);
 },
 function insert_default(callback) {
   const hike = {HIKE_DATE: new Date(), NAME: 'Hazard Stevens',
         LOCATION: 'Mt Rainier', DISTANCE: '4,027m vertical', WEATHER:'Bad', ID:
 '12345'};
   client.query('INSERT INTO HIKES set ?', hike, callback);
 }
], function (err, results) {
 if (err) {
   console.log('Exception initializing database.');
   throw err;
 } else {
   console.log('Database initialization complete.');
   init();
 }
```

```
});

module.exports = app
```

5. Ajoutez le contenu suivant à `routes/hike.js`. Cela permettra aux acheminements d'insérer de nouveaux journaux de randonnée dans la base de données HIKES.

```
const uuid = require('node-uuid');
exports.index = function(req, res) {
  res.app.get('connection').query( 'SELECT * FROM HIKES', function(err,
rows) {
    if (err) {
      res.send(err);
    } else {
      console.log(JSON.stringify(rows));
      res.render('hike', {title: 'My Hiking Log', hikes: rows});
  }});
};
exports.add_hike = function(req, res){
  const input = req.body.hike;
  const hike = { HIKE_DATE: new Date(), ID: uuid.v4(), NAME: input.NAME,
  LOCATION: input.LOCATION, DISTANCE: input.DISTANCE, WEATHER: input.WEATHER};
  console.log('Request to log hike:' + JSON.stringify(hike));
  req.app.get('connection').query('INSERT INTO HIKES set ?', hike, function(err) {
      if (err) {
        res.send(err);
      } else {
        res.redirect('/hikes');
      }
   });
};
```

6. Remplacez le contenu de `routes/index.js` par ce qui suit :

```
/*
 * GET home page.
 */

exports.index = function(req, res){
  res.render('index', { title: 'Express' });
};
```

7. Ajoutez le modèle jade suivant à `views/hike.jade` pour fournir l'interface utilisateur permettant d'ajouter des journaux de randonnée.

```
extends layout

block content
  h1= title
  p Welcome to #{title}

  form(action="/add_hike", method="post")
    table(border="1")
      tr
        td Your Name
        td
          input(name="hike[NAME]", type="textbox")
      tr
        td Location
        td
          input(name="hike[LOCATION]", type="textbox")
      tr
        td Distance
        td
          input(name="hike[DISTANCE]", type="textbox")
      tr
        td Weather
        td
          input(name="hike[WEATHER]", type="radio", value="Good")
          | Good
          input(name="hike[WEATHER]", type="radio", value="Bad")
          | Bad
          input(name="hike[WEATHER]", type="radio", value="Seattle", checked)
          | Seattle
      tr
        td(colspan="2")
          input(type="submit", value="Record Hike")

  div
    h3 Hikes
    table(border="1")
      tr
        td Date
        td Name
        td Location
```

```
        td Distance
        td Weather
      each hike in hikes
        tr
          td #{hike.HIKE_DATE.toDateString()}
          td #{hike.NAME}
          td #{hike.LOCATION}
          td #{hike.DISTANCE}
          td #{hike.WEATHER}
```

8. Déployez les modifications avec la commande eb deploy.

```
~/nodejs-example-express-rds$ eb deploy
```

Nettoyage

Si vous avez terminé de travailler avec Elastic Beanstalk, vous pouvez résilier votre environnement.

Utilisez la commande eb terminate pour suspendre votre environnement et toutes les ressources qu'il contient.

```
~/nodejs-example-express-rds$ eb terminate
The environment "nodejs-example-express-rds-env" and all associated instances will be
 terminated.
To confirm, type the environment name: nodejs-example-express-rds-env
INFO: terminateEnvironment is starting.
...
```

Déploiement d'une application Node.js Express avec clustering sur Elastic Beanstalk

Ce didacticiel explique comment déployer un exemple d'application sur Elastic Beanstalk à l'aide de l'interface de ligne de commande Elastic Beanstalk (EB CLI), puis comment mettre à jour l'application pour utiliser le framework Express, Amazon et le clustering. ElastiCache La mise en cluster permet d'améliorer la haute disponibilité, les performances et la sécurité de votre application web. Pour en savoir plus sur Amazon ElastiCache, rendez-vous sur Qu'est-ce qu'Amazon ElastiCache (Memcached) ? dans le guide de l'utilisateur d'Amazon ElastiCache (Memcached).

> **Note**
>
> Cet exemple crée AWS des ressources qui peuvent vous être facturées. Pour plus d'informations sur la AWS tarification, consultezhttps://aws.amazon.com/pricing/. Certains services font partie du niveau d'utilisation AWS gratuit. Si vous êtes un nouveau client, vous pouvez essayer ces services gratuitement. Pour plus d'informations, consultez https://aws.amazon.com/free/.

Conditions préalables

Ce tutoriel nécessite les prérequis suivants :

- Les exécutions Node.js
- Le logiciel de gestion des packages Node.js par défaut, npm
- Le générateur de ligne de commande Express
- Interface de ligne de commande Elastic Beanstalk (EB)

Pour plus d'informations sur l'installation des trois premiers composants et la configuration de votre environnement de développement local, consultez Configuration de votre environnement de développement Node.js pour Elastic Beanstalk. Pour ce didacticiel, il n'est pas nécessaire d'installer le AWS SDK pour Node.js, qui est également mentionné dans la rubrique référencée.

Pour plus d'informations sur l'installation et la configuration de l'EB CLI, consultez Installation de la CLI EB avec un script de configuration (recommandé) et Configuration de l'interface de ligne de commande EB.

Créer un environnement Elastic Beanstalk

Votre répertoire d'application

Ce tutoriel utilise un répertoire appelé `nodejs-example-express-elasticache` pour la création de la solution groupée de l'application. Créez le répertoire `nodejs-example-express-elasticache` pour ce tutoriel.

```
~$ mkdir nodejs-example-express-elasticache
```

> **Note**
>
> Chaque tutoriel de ce chapitre utilise son propre répertoire pour la solution groupée de l'application. Le nom du répertoire correspond au nom de l'exemple d'application utilisé par le tutoriel.

Changez votre répertoire de travail actuel vers `nodejs-example-express-elasticache`.

```
~$ cd nodejs-example-express-elasticache
```

Configurons maintenant un environnement Elastic Beanstalk exécutant la plateforme Node.js et l'exemple d'application. Nous utiliserons l'interface de ligne de commande Elastic Beanstalk (EB CLI).

Pour configurer un référentiel de l'EB CLI pour votre application et créer un environnement Elastic Beanstalk qui exécute la plateforme Node.js

1. Créez un référentiel à l'aide de la commande eb init.

   ```
   ~/nodejs-example-express-elasticache$ eb init --platform node.js --region <region>
   ```

   Cette commande crée un fichier de configuration dans un dossier nommé `.elasticbeanstalk` qui spécifie les paramètres de création d'environnements pour votre application et crée une application Elastic Beanstalk dont le nom est basé sur le dossier actif.
2. Créez un environnement qui exécute un exemple d'application à l'aide de la commande eb create.

   ```
   ~/nodejs-example-express-elasticache$ eb create --sample nodejs-example-express-elasticache
   ```

   Cette commande crée un environnement à charge équilibrée avec les paramètres par défaut de la plateforme Node.js et les ressources suivantes :

   - EC2 instance — Une machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications Web sur la plateforme de votre choix.

     Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur

web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Équilibreur de charge – Équilibreur de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité d'équilibrage de charge : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibreur de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et qui sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack : Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *`subdomain`*.*`region`*.elasticbeanstalk.com.

> **Sécurité du domaine**
>
> Pour renforcer la sécurité de vos applications Elastic Beanstalk, le domaine elasticbeanstalk.com est enregistré dans la liste des suffixes publics (PSL).
> Si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Elastic Beanstalk, nous vous recommandons d'utiliser des cookies `__Host-` avec un préfixe pour une sécurité accrue. Cette pratique protège votre

domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

3. Une fois l'environnement créé, utilisez la commande eb open pour ouvrir l'URL de l'environnement dans le navigateur par défaut.

```
~/nodejs-example-express-elasticache$ eb open
```

Vous avez maintenant créé un environnement Node.js Elastic Beanstalk avec un exemple d'application. Vous pouvez le mettre à jour avec votre propre application. Ensuite, nous mettons à jour l'exemple d'application afin d'utiliser l'infrastructure Express.

Mise à jour de l'application pour utiliser Express

Mettez à jour l'exemple d'application dans l'environnement Elastic Beanstalk pour utiliser le cadre Express.

Vous pouvez télécharger le code source final depuis le nodejs-example-express-elasticachefichier .zip.

Pour mettre à jour votre application afin d'utiliser Express

Une fois que vous avez créé un environnement avec un exemple d'application, vous pouvez le mettre à jour à l'aide de votre propre application. Dans cette procédure, nous exécutons d'abord les commandes express et npm install pour configurer le cadre Express dans votre répertoire d'application.

1. Exécutez la commande `express`. Il en résulte la génération de `package.json`, `app.js`, et de quelques répertoires.

```
~/nodejs-example-express-elasticache$ express
```

Lorsque vous êtes invité à continuer, tapez **y**.

> **Note**
>
> Si la commande express ne fonctionne pas, il se peut que vous n'ayez pas installé le générateur de ligne de commande Express, comme décrit dans la section précédente sur les prérequis. Il se peut également que le chemin d'accès au répertoire de votre

ordinateur local doive être configuré pour permettre l'exécution de la commande express. Consultez la section Prérequis pour connaître les étapes détaillées de la configuration de votre environnement de développement, afin de pouvoir poursuivre ce tutoriel.

2. Configurez les dépendances locales.

```
~/nodejs-example-express-elasticache$ npm install
```

3. (Facultatif) Vérifiez que le serveur de l'application web démarre.

```
~/nodejs-example-express-elasticache$ npm start
```

Vous devez voir des résultats similaires à ce qui suit :

```
> nodejs@0.0.0 start /home/local/user/node-express
> node ./bin/www
```

Par défaut, le serveur s'exécute sur le port 3000. Pour le tester, exécutez `curl http://localhost:3000` sur un autre terminal, ou ouvrez un navigateur sur l'ordinateur local et entrez l'adresse URL `http://localhost:3000`.

Appuyez sur Ctrl+C afin d'arrêter le serveur.

4. Renommez `nodejs-example-express-elasticache/app.js` en `nodejs-example-express-elasticache/express-app.js`.

```
~/nodejs-example-express-elasticache$ mv app.js express-app.js
```

5. Mettez à jour la ligne `var app = express();` dans `nodejs-example-express-elasticache/express-app.js` avec les éléments suivants :

```
var app = module.exports = express();
```

6. Sur votre ordinateur local, créez un fichier nommé `nodejs-example-express-elasticache/app.js` avec le code suivant.

```
/**
 * Module dependencies.
 */
```

```
 const express = require('express'),
 session = require('express-session'),
 bodyParser = require('body-parser'),
 methodOverride = require('method-override'),
 cookieParser = require('cookie-parser'),
 fs = require('fs'),
 filename = '/var/nodelist',
 app = express();

let MemcachedStore = require('connect-memcached')(session);

function setup(cacheNodes) {
 app.use(bodyParser.raw());
 app.use(methodOverride());
 if (cacheNodes.length > 0) {
   app.use(cookieParser());

   console.log('Using memcached store nodes:');
   console.log(cacheNodes);

   app.use(session({
     secret: 'your secret here',
     resave: false,
     saveUninitialized: false,
     store: new MemcachedStore({ 'hosts': cacheNodes })
   }));
 } else {
   console.log('Not using memcached store.');
   app.use(session({
     resave: false,
     saveUninitialized: false, secret: 'your secret here'
   }));
 }

 app.get('/', function (req, resp) {
   if (req.session.views) {
     req.session.views++
     resp.setHeader('Content-Type', 'text/html')
     resp.send(`You are session: ${req.session.id}. Views: ${req.session.views}`)
   } else {
     req.session.views = 1
     resp.send(`You are session: ${req.session.id}. No views yet, refresh the page!
`)
   }
```

```
 });

 if (!module.parent) {
   console.log('Running express without cluster. Listening on port %d',
 process.env.PORT || 5000)
   app.listen(process.env.PORT || 5000)
 }
}

console.log("Reading elastic cache configuration")
// Load elasticache configuration.
fs.readFile(filename, 'UTF8', function (err, data) {
 if (err) throw err;

 let cacheNodes = []
 if (data) {
   let lines = data.split('\n');
   for (let i = 0; i < lines.length; i++) {
     if (lines[i].length > 0) {
       cacheNodes.push(lines[i])
     }
   }
 }

 setup(cacheNodes)
});

module.exports = app;
```

7. Remplacez le contenu du fichier `nodejs-example-express-elasticache/bin/www` par ce qui suit :

```
#!/usr/bin/env node

/**
 * Module dependencies.
 */

const app = require('../app');
const cluster = require('cluster');
const debug = require('debug')('nodejs-example-express-elasticache:server');
const http = require('http');
const workers = {},
```

```
  count = require('os').cpus().length;

function spawn() {
  const worker = cluster.fork();
  workers[worker.pid] = worker;
  return worker;
}


/**
 * Get port from environment and store in Express.
 */

const port = normalizePort(process.env.PORT || '3000');
app.set('port', port);

if (cluster.isMaster) {
  for (let i = 0; i < count; i++) {
    spawn();
  }

  // If a worker dies, log it to the console and start another worker.
  cluster.on('exit', function (worker, code, signal) {
    console.log('Worker ' + worker.process.pid + ' died.');
    cluster.fork();
  });

  // Log when a worker starts listening
  cluster.on('listening', function (worker, address) {
    console.log('Worker started with PID ' + worker.process.pid + '.');
  });

} else {
  /**
   * Create HTTP server.
   */

  let server = http.createServer(app);

  /**
   * Event listener for HTTP server "error" event.
   */

  function onError(error) {
```

```
    if (error.syscall !== 'listen') {
      throw error;
    }

    const bind = typeof port === 'string'
      ? 'Pipe ' + port
      : 'Port ' + port;

    // handle specific listen errors with friendly messages
    switch (error.code) {
      case 'EACCES':
        console.error(bind + ' requires elevated privileges');
        process.exit(1);
        break;
      case 'EADDRINUSE':
        console.error(bind + ' is already in use');
        process.exit(1);
        break;
      default:
        throw error;
    }
  }

  /**
   * Event listener for HTTP server "listening" event.
   */

  function onListening() {
    const addr = server.address();
    const bind = typeof addr === 'string'
      ? 'pipe ' + addr
      : 'port ' + addr.port;
    debug('Listening on ' + bind);
  }

  /**
   * Listen on provided port, on all network interfaces.
   */

  server.listen(port);
  server.on('error', onError);
  server.on('listening', onListening);
}
```

```
/**
 * Normalize a port into a number, string, or false.
 */

function normalizePort(val) {
  const port = parseInt(val, 10);

  if (isNaN(port)) {
    // named pipe
    return val;
  }

  if (port >= 0) {
    // port number
    return port;
  }

  return false;
}
```

8. Déployez les modifications dans votre environnement Elastic Beanstalk avec la commande eb deploy.

```
~/nodejs-example-express-elasticache$ eb deploy
```

9. Votre environnement est mis à jour au bout de quelques minutes. Une fois que votre environnement est prêt (il apparaît en vert), actualisez l'URL pour vérifier si l'opération a fonctionné. Vous devriez voir une page web indiquant « Welcome to Express ».

Vous pouvez accéder aux journaux des EC2 instances qui exécutent votre application. Pour plus d'informations sur l'accès à vos journaux, consultez Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk.

Ensuite, mettons à jour l'application Express pour utiliser Amazon ElastiCache.

Pour mettre à jour votre application Express afin d'utiliser Amazon ElastiCache

1. Sur votre ordinateur local, créez un répertoire `.ebextensions` dans le répertoire de niveau supérieur de votre groupe source. Dans cet exemple, nous utilisons `nodejs-example-express-elasticache/.ebextensions`.

2. Créez un fichier de configuration `nodejs-example-express-elasticache/.ebextensions/elasticache-iam-with-script.config` avec l'extrait suivant. Pour plus d'informations sur le fichier de configuration, consultez Espaces de noms de la configuration Node.js. Il en résulte la création d'un utilisateur IAM avec les autorisations requises pour découvrir les nœuds ElastiCache et l'écriture dans un fichier chaque fois que le cache change. Vous pouvez également copier le fichier depuis un fichier nodejs-example-express-elasticache.zip. Pour plus d'informations sur les ElastiCache propriétés, consultezExemple : ElastiCache.

> **Note**
>
> YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

```
Resources:
  MyCacheSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: "Lock cache down to webserver access only"
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort:
            Fn::GetOptionSetting:
              OptionName: CachePort
              DefaultValue: 11211
          ToPort:
            Fn::GetOptionSetting:
              OptionName: CachePort
              DefaultValue: 11211
          SourceSecurityGroupName:
            Ref: AWSEBSecurityGroup
  MyElastiCache:
    Type: 'AWS::ElastiCache::CacheCluster'
    Properties:
      CacheNodeType:
        Fn::GetOptionSetting:
          OptionName: CacheNodeType
```

```
          DefaultValue: cache.t2.micro
      NumCacheNodes:
        Fn::GetOptionSetting:
          OptionName: NumCacheNodes
          DefaultValue: 1
      Engine:
        Fn::GetOptionSetting:
          OptionName: Engine
          DefaultValue: redis
      VpcSecurityGroupIds:
        -
          Fn::GetAtt:
            - MyCacheSecurityGroup
            - GroupId
  AWSEBAutoScalingGroup :
    Metadata :
      ElastiCacheConfig :
        CacheName :
          Ref : MyElastiCache
        CacheSize :
           Fn::GetOptionSetting:
             OptionName : NumCacheNodes
             DefaultValue: 1
  WebServerUser :
    Type : AWS::IAM::User
    Properties :
      Path : "/"
      Policies:
        -
          PolicyName: root
          PolicyDocument :
            Statement :
              -
                Effect : Allow
                Action :
                  - cloudformation:DescribeStackResource
                  - cloudformation:ListStackResources
                  - elasticache:DescribeCacheClusters
                Resource : "*"
  WebServerKeys :
    Type : AWS::IAM::AccessKey
    Properties :
      UserName :
        Ref: WebServerUser
```

```
Outputs:
  WebsiteURL:
    Description: sample output only here to show inline string function parsing
    Value: |
      http://`{ "Fn::GetAtt" : [ "AWSEBLoadBalancer", "DNSName" ] }`
  MyElastiCacheName:
    Description: Name of the elasticache
    Value:
      Ref : MyElastiCache
  NumCacheNodes:
    Description: Number of cache nodes in MyElastiCache
    Value:
      Fn::GetOptionSetting:
        OptionName : NumCacheNodes
        DefaultValue: 1

files:
  "/etc/cfn/cfn-credentials" :
    content : |
      AWSAccessKeyId=`{ "Ref" : "WebServerKeys" }`
      AWSSecretKey=`{ "Fn::GetAtt" : ["WebServerKeys", "SecretAccessKey"] }`
    mode : "000400"
    owner : root
    group : root

  "/etc/cfn/get-cache-nodes" :
    content : |
      # Define environment variables for command line tools
      export AWS_ELASTICACHE_HOME="/home/ec2-user/elasticache/$(ls /home/ec2-user/
elasticache/)"
      export AWS_CLOUDFORMATION_HOME=/opt/aws/apitools/cfn
      export PATH=$AWS_CLOUDFORMATION_HOME/bin:$AWS_ELASTICACHE_HOME/bin:$PATH
      export AWS_CREDENTIAL_FILE=/etc/cfn/cfn-credentials
      export JAVA_HOME=/usr/lib/jvm/jre

      # Grab the Cache node names and configure the PHP page
      aws cloudformation list-stack-resources --stack `{ "Ref" :
 "AWS::StackName" }` --region `{ "Ref" : "AWS::Region" }` --output text | grep
 MyElastiCache | awk '{print $4}' | xargs -I {} aws elasticache describe-cache-
clusters --cache-cluster-id {} --region `{ "Ref" : "AWS::Region" }` --show-
cache-node-info --output text | grep '^ENDPOINT' | awk '{print $2 ":" $3}' >
 `{ "Fn::GetOptionSetting" : { "OptionName" : "NodeListPath", "DefaultValue" : "/
var/www/html/nodelist" } }`
```

```
    mode : "000500"
    owner : root
    group : root

  "/etc/cfn/hooks.d/cfn-cache-change.conf" :
    "content": |
      [cfn-cache-size-change]
      triggers=post.update
      path=Resources.AWSEBAutoScalingGroup.Metadata.ElastiCacheConfig
      action=/etc/cfn/get-cache-nodes
      runas=root

sources :
  "/home/ec2-user/elasticache" : "https://elasticache-downloads.s3.amazonaws.com/
AmazonElastiCacheCli-latest.zip"

commands:
  make-elasticache-executable:
    command: chmod -R ugo+x /home/ec2-user/elasticache/*/bin/*

packages :
  "yum" :
    "aws-apitools-cfn"  : []

container_commands:
  initial_cache_nodes:
    command: /etc/cfn/get-cache-nodes
```

3. Sur votre ordinateur local, créez un fichier de configuration `nodejs-example-express-elasticache/.ebextensions/elasticache_settings.config` avec l'extrait de code suivant à configurer. ElastiCache

```
option_settings:
  "aws:elasticbeanstalk:customoption":
     CacheNodeType: cache.t2.micro
     NumCacheNodes: 1
     Engine: memcached
     NodeListPath: /var/nodelist
```

4. Sur votre ordinateur local, remplacez `nodejs-example-express-elasticache/express-app.js` par l'extrait suivant. Ce fichier lit la liste des nœuds depuis le disque (`/var/nodelist`) et configure Express de façon à utiliser `memcached` comme magasin de sessions s'il existe des nœuds. Votre fichier doit se présenter comme suit :

```
/**
 * Module dependencies.
 */

var express = require('express'),
    session = require('express-session'),
    bodyParser = require('body-parser'),
    methodOverride = require('method-override'),
    cookieParser = require('cookie-parser'),
    fs = require('fs'),
    filename = '/var/nodelist',
    app = module.exports = express();

var MemcachedStore = require('connect-memcached')(session);

function setup(cacheNodes) {
  app.use(bodyParser.raw());
  app.use(methodOverride());
  if (cacheNodes) {
      app.use(cookieParser());

      console.log('Using memcached store nodes:');
      console.log(cacheNodes);

      app.use(session({
          secret: 'your secret here',
          resave: false,
          saveUninitialized: false,
          store: new MemcachedStore({'hosts': cacheNodes})
      }));
  } else {
    console.log('Not using memcached store.');
    app.use(cookieParser('your secret here'));
    app.use(session());
  }

  app.get('/', function(req, resp){
  if (req.session.views) {
      req.session.views++
      resp.setHeader('Content-Type', 'text/html')
      resp.write('Views: ' + req.session.views)
      resp.end()
   } else {
```

```
      req.session.views = 1
      resp.end('Refresh the page!')
    }
  });

  if (!module.parent) {
      console.log('Running express without cluster.');
      app.listen(process.env.PORT || 5000);
  }
}

// Load elasticache configuration.
fs.readFile(filename, 'UTF8', function(err, data) {
    if (err) throw err;

    var cacheNodes = [];
    if (data) {
        var lines = data.split('\n');
        for (var i = 0 ; i < lines.length ; i++) {
            if (lines[i].length > 0) {
                cacheNodes.push(lines[i]);
            }
        }
    }
    setup(cacheNodes);
});
```

5. Sur votre ordinateur local, mettez à jour `package.json` avec le contenu suivant :

```
  "dependencies": {
    "cookie-parser": "~1.4.4",
    "debug": "~2.6.9",
    "express": "~4.16.1",
    "http-errors": "~1.6.3",
    "jade": "~1.11.0",
    "morgan": "~1.9.1",
    "connect-memcached": "*",
    "express-session": "*",
    "body-parser": "*",
    "method-override": "*"
  }
```

6. Exécutez npm install.

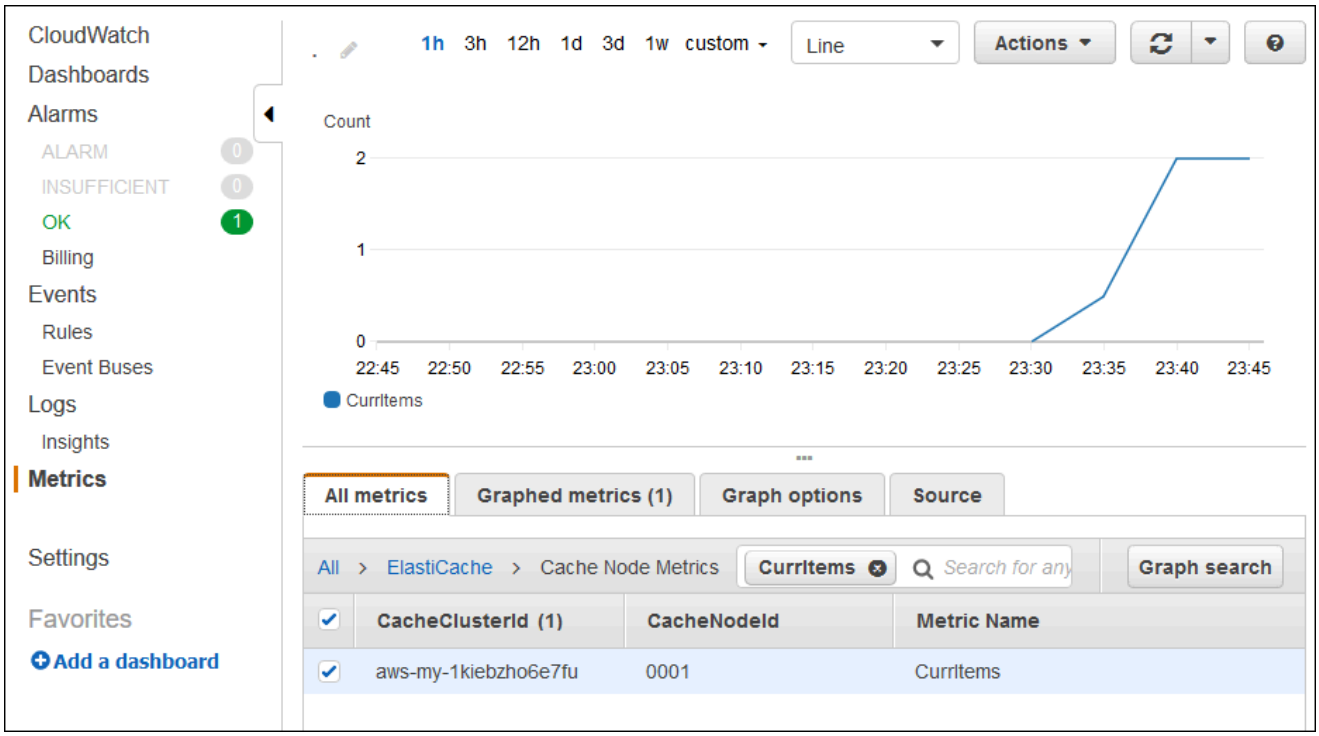
```
~/nodejs-example-express-elasticache$ npm install
```

7. Déployez l'application mise à jour.

```
~/nodejs-example-express-elasticache$ eb deploy
```

8. Votre environnement est mis à jour au bout de quelques minutes. Une fois que votre environnement est prêt (il apparaît en vert), assurez-vous que le code a fonctionné.

   a. Consultez la CloudWatch console Amazon pour consulter vos ElastiCache statistiques. Pour afficher vos ElastiCache statistiques, sélectionnez Métriques dans le volet de gauche, puis recherchez CurrItems. Sélectionnez ElastiCache > Mesures du nœud de cache, puis sélectionnez votre nœud de cache pour afficher le nombre d'éléments contenus dans le cache.



> **Note**
>
> Vérifiez que vous consultez bien la même région que celle dans laquelle vous avez déployé votre application.

Si vous copiez et collez l'URL de votre application dans un autre navigateur Web et que vous actualisez la page, vous devriez voir votre CurrItem compte augmenter au bout de 5 minutes.

b. Prenez un instantané de vos journaux. Pour de plus amples informations sur la récupération des journaux, veuillez consulter Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk.

c. Vérifiez le fichier `/var/log/nodejs/nodejs.log` dans le groupe des journaux. Le résultat devrait être similaire à ce qui suit :

```
Using memcached store nodes:
[ 'aws-my-1oys9co8zt1uo.1iwtrn.0001.use1.cache.amazonaws.com:11211' ]
```

Nettoyage

Si vous ne souhaitez plus exécuter votre application, vous pouvez effectuer un nettoyage en suspendant votre environnement et en supprimant votre application.

Utilisez la commande `eb terminate` pour mettre votre environnement hors service et la commande `eb delete` pour supprimer votre application.

Pour résilier votre environnement

Depuis le répertoire où vous avez créé votre référentiel local, exécutez `eb terminate`.

```
$ eb terminate
```

Ce processus peut prendre quelques minutes. Elastic Beanstalk affiche un message une fois que l'environnement est arrêté.

Déploiement d'une application Node.js avec DynamoDB vers Elastic Beanstalk

Ce didacticiel et son exemple d'application nodejs-example-dynamo.zip vous guident JavaScript dans le processus de déploiement d'une application Node.js qui utilise le AWS SDK de Node.js pour

interagir avec le service Amazon DynamoDB. Vous allez créer une table DynamoDB qui se trouve dans une base de données découplée, ou externe, de l'environnement. AWS Elastic Beanstalk Vous allez également configurer l'application pour qu'elle utilise une base de données dissociée. Dans un environnement de production, l'utilisation d'une base de données dissociée de l'environnement Elastic Beanstalk constitue une bonne pratique, afin qu'elle soit indépendante du cycle de vie de l'environnement. Cette pratique vous permet également d'effectuer des déploiements bleus/verts.

L'exemple d'application illustre ce qui suit :

- Une table DynamoDB qui stocke des données textuelles fournies par l'utilisateur.
- Les fichiers de configuration pour créer la table.
- Une rubrique Amazon Simple Notification Service.
- L'utilisation d'un fichier package.json pour installer des packages pendant le déploiement.

Sections

- Conditions préalables
- Créer un environnement Elastic Beanstalk
- Ajout d'autorisations aux instances de votre environnement
- Déploiement de l'exemple d'application
- Créez une table DynamoDB
- Mise à jour des fichiers de configuration de l'application
- Configuration de votre environnement pour une haute disponibilité
- Nettoyage
- Étapes suivantes

Conditions préalables

Ce tutoriel nécessite les prérequis suivants :

- Les exécutions Node.js
- Le logiciel de gestion des packages Node.js par défaut, npm
- Le générateur de ligne de commande Express
- Interface de ligne de commande Elastic Beanstalk (EB)

Pour plus d'informations sur l'installation des trois premiers composants et la configuration de votre environnement de développement local, consultez Configuration de votre environnement de développement Node.js pour Elastic Beanstalk. Pour ce didacticiel, il n'est pas nécessaire d'installer le AWS SDK pour Node.js, qui est également mentionné dans la rubrique référencée.

Pour plus d'informations sur l'installation et la configuration de l'EB CLI, consultez Installation de la CLI EB avec un script de configuration (recommandé) et Configuration de l'interface de ligne de commande EB.

Créer un environnement Elastic Beanstalk

Votre répertoire d'application

Ce tutoriel utilise un répertoire appelé `nodejs-example-dynamo` pour la création de la solution groupée de l'application. Créez le répertoire `nodejs-example-dynamo` pour ce tutoriel.

```
~$ mkdir nodejs-example-dynamo
```

> **Note**
>
> Chaque tutoriel de ce chapitre utilise son propre répertoire pour la solution groupée de l'application. Le nom du répertoire correspond au nom de l'exemple d'application utilisé par le tutoriel.

Changez votre répertoire de travail actuel vers `nodejs-example-dynamo`.

```
~$ cd nodejs-example-dynamo
```

Configurons maintenant un environnement Elastic Beanstalk exécutant la plateforme Node.js et l'exemple d'application. Nous utiliserons l'interface de ligne de commande Elastic Beanstalk (EB CLI).

Pour configurer un référentiel de l'EB CLI pour votre application et créer un environnement Elastic Beanstalk qui exécute la plateforme Node.js

1. Créez un référentiel à l'aide de la commande eb init.

   ```
   ~/nodejs-example-dynamo$ eb init --platform node.js --region <region>
   ```

Cette commande crée un fichier de configuration dans un dossier nommé `.elasticbeanstalk` qui spécifie les paramètres de création d'environnements pour votre application et crée une application Elastic Beanstalk dont le nom est basé sur le dossier actif.

2. Créez un environnement qui exécute un exemple d'application à l'aide de la commande eb create.

```
~/nodejs-example-dynamo$ eb create --sample nodejs-example-dynamo
```

Cette commande crée un environnement à charge équilibrée avec les paramètres par défaut de la plateforme Node.js et les ressources suivantes :

- EC2 instance — Une machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications Web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.
- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Équilibreur de charge – Équilibreur de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité d'équilibrage de charge : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibreur de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.

- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et qui sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

> **Sécurité du domaine**
>
> Pour renforcer la sécurité de vos applications Elastic Beanstalk, le domaine elasticbeanstalk.com est enregistré dans la liste des suffixes publics (PSL).
> Si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Elastic Beanstalk, nous vous recommandons d'utiliser des cookies `__Host-` avec un préfixe pour une sécurité accrue. Cette pratique protège votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

3. Une fois l'environnement créé, utilisez la commande eb open pour ouvrir l'URL de l'environnement dans le navigateur par défaut.

```
~/nodejs-example-dynamo$ eb open
```

Vous avez maintenant créé un environnement Node.js Elastic Beanstalk avec un exemple d'application. Vous pouvez le mettre à jour avec votre propre application. Ensuite, nous mettons à jour l'exemple d'application afin d'utiliser l'infrastructure Express.

Ajout d'autorisations aux instances de votre environnement

Votre application s'exécute sur une ou plusieurs EC2 instances derrière un équilibreur de charge, servant les requêtes HTTP provenant d'Internet. Lorsqu'elle reçoit une demande qui l'oblige à utiliser AWS des services, l'application utilise les autorisations de l'instance sur laquelle elle s'exécute pour accéder à ces services.

L'exemple d'application utilise les autorisations d'instance pour écrire des données dans une table DynamoDB et pour envoyer des notifications à une rubrique Amazon SNS à l'aide du SDK pour Node.js. JavaScript Ajoutez les politiques gérées suivantes au profil d'instance par défaut pour autoriser les EC2 instances de votre environnement à accéder à DynamoDB et à Amazon SNS :

- AmazonDynamoDBFullAccès
- Amazon SNSFull Access

Pour ajouter des stratégies au profil d'instance par défaut

1. Ouvrez la page Roles (Rôles) dans la console IAM.
2. Choisissez aws-elasticbeanstalk-ec2 rôles.
3. Sous l'onglet Permissions (Autorisations), choisissez Attach policies (Attacher des politiques).
4. Sélectionnez la politique gérée relative aux services supplémentaires utilisés par votre application. Dans le cadre de ce tutoriel, sélectionnez `AmazonSNSFullAccess` et `AmazonDynamoDBFullAccess`.
5. Choisissez Attach policy (Attacher la politique).

Pour de plus amples informations sur la gestion des profils d'instance, veuillez consulter Gestion des profils d'instance Elastic Beanstalk.

Déploiement de l'exemple d'application

Votre environnement est maintenant prêt pour que vous puissiez déployer et exécuter l'exemple d'application pour ce didacticiel : nodejs-example-dynamo.zip.

Pour déployer et exécuter l'exemple d'application du tutoriel

1. Changez votre répertoire de travail actuel vers le répertoire de l'application `nodejs-example-dynamo`.

   ```
   ~$ cd nodejs-example-dynamo
   ```

2. Téléchargez et extrayez le contenu de l'exemple de bundle source d'application nodejs-example-dynamo.zip dans le répertoire `nodejs-example-dynamo` de l'application.
3. Déployez l'exemple d'application dans votre environnement Elastic Beanstalk avec la commande eb deploy.

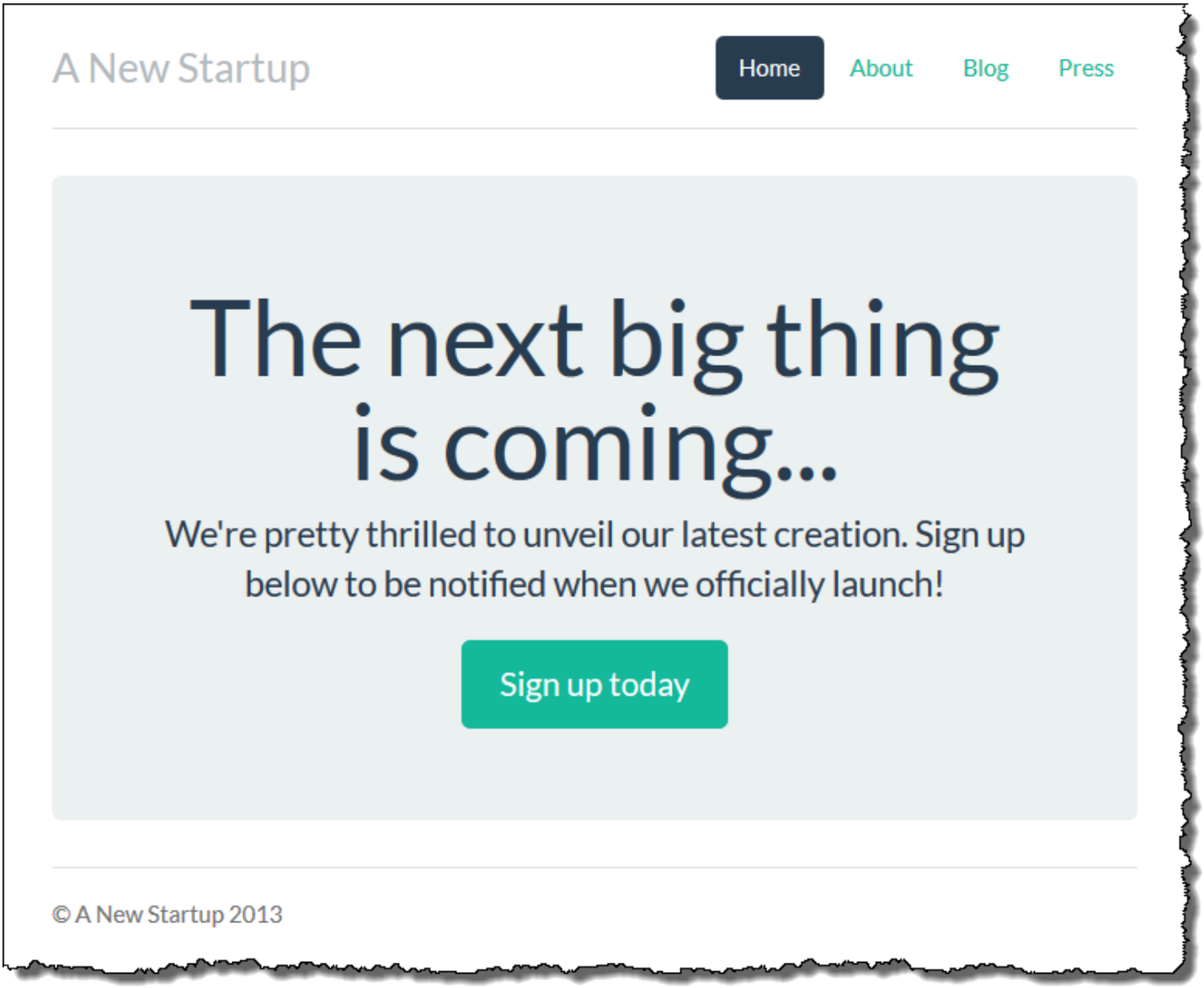
```
~/nodejs-example-dynamo$ eb deploy
```

> **Note**
>
> Par défaut, la `eb deploy` commande crée un fichier ZIP du dossier de votre projet. Vous pouvez configurer l'interface de ligne de commande (CLI) EB pour déployer un artefact de votre processus de construction au lieu de créer un fichier ZIP de votre dossier de projet. Pour de plus amples informations, veuillez consulter Déploiement d'un artefact à la place du dossier de projet.

4. Une fois l'environnement créé, utilisez la commande eb open pour ouvrir l'URL de l'environnement dans le navigateur par défaut.

```
~/nodejs-example-dynamo$ eb open
```

Le site collecte les informations de contact des utilisateurs et stocke les données dans une table DynamoDB. Pour ajouter une entrée, choisissez Sign up today (Inscrivez-vous aujourd'hui), entrez un nom et une adresse e-mail et choisissez Sign Up! (Inscrivez-vous). L'application web écrit le contenu du formulaire dans la table et déclenche une notification Amazon SNS par e-mail.



Pour le moment, la rubrique Amazon SNS est configurée avec un e-mail à espace réservé pour les notifications. Vous allez mettre à jour la configuration prochainement, mais en attendant, vous pouvez vérifier la table DynamoDB et la rubrique Amazon SNS dans AWS Management Console Management Console.

Pour afficher la table

1. Ouvrez la page Tables dans la console DynamoDB.
2. Recherchez la table créée par l'application. Le nom commence par awseb et contient. StartupSignupsTable

3. Sélectionnez la table, choisissez Items (Éléments), puis Start search (Commencer la recherche) afin d'afficher tous les éléments de la table.

La table contient une entrée pour chaque adresse e-mail fournie sur le site d'inscription. En plus d'écrire les données dans la table, l'application envoie un message à une rubrique Amazon SNS qui compte deux abonnements. Le premier abonnement est pour les notifications par e-mail qui vous sont adressées et l'autre pour une file d'attente Amazon Simple Queue Service qu'une application de travail peut lire afin de traiter les demandes et d'envoyer des e-mails aux clients intéressés.

Pour afficher la rubrique

1. Ouvrez la page Topics (Rubriques) dans la console Amazon SNS.
2. Recherchez la rubrique créée par l'application. Le nom commence par awseb et contient. NewSignupTopic
3. Choisissez la rubrique pour afficher ses abonnements.

L'application (`app.js`) définit deux routes. Le chemin racine (`/`) renvoie une page Web rendue à partir d'un modèle intégré JavaScript (EJS) avec un formulaire que l'utilisateur remplit pour enregistrer son nom et son adresse e-mail. La soumission du formulaire envoie une demande POST avec les données du formulaire vers l'acheminement `/signup`, qui écrit une entrée dans la table DynamoDB et publie un message dans la rubrique Amazon SNS pour informer le propriétaire de l'inscription.

L'exemple d'application inclut des fichiers de configuration qui créent la table DynamoDB, la rubrique Amazon SNS et la file d'attente Amazon SQS utilisées par l'application. Cela vous permet de créer un nouvel environnement et de tester la fonctionnalité immédiatement, mais en revanche la table DynamoDB est liée à l'environnement. Pour un environnement de production, vous devez créer la table DynamoDB en dehors de l'environnement afin d'éviter de la perdre lorsque vous arrêtez l'environnement ou que vous mettez sa configuration à jour.

Créez une table DynamoDB

Pour utiliser une table DynamoDB externe avec une application exécutée dans Elastic Beanstalk, créez d'abord une table dans DynamoDB. Lorsque vous créez une table en dehors d'Elastic Beanstalk, elle est complètement indépendante d'Elastic Beanstalk et de vos environnements Elastic Beanstalk et ne sera pas arrêtée par Elastic Beanstalk.

Créez une table avec les paramètres suivants :

- Nom de la table – **`nodejs-tutorial`**
- Clé primaire – **`email`**
- Type de clé primaire – Chaîne

Pour créer une table DynamoDB

1. Ouvrez la page Tables dans la console de gestion DynamoDB.
2. Choisissez Créer un tableau.
3. Entrez un nom de table et une clé primaire.
4. Choisissez le type de clé primaire.
5. Choisissez Créer.

Mise à jour des fichiers de configuration de l'application

Mettez à jour les fichiers de configuration dans la source de l'application afin d'utiliser la table nodejs-tutorial au lieu d'en créer une nouvelle.

Pour mettre à jour l'exemple d'application pour l'utilisation en production

1. Changez votre répertoire de travail actuel vers le répertoire de l'application `nodejs-example-dynamo`.

   ```
   ~$ cd nodejs-example-dynamo
   ```

2. Ouvrez `.ebextensions/options.config` et modifiez les valeurs des paramètres suivants :

   - NewSignupEmail— Votre adresse e-mail.
   - STARTUP_SIGNUP_TABLE – nodejs-tutorial

   Example.ebextensions/options.config

   ```
   option_settings:
     aws:elasticbeanstalk:customoption:
       NewSignupEmail: you@example.com
     aws:elasticbeanstalk:application:environment:
       THEME: "flatly"
   ```

```
    AWS_REGION: '`{"Ref" : "AWS::Region"}`'
    STARTUP_SIGNUP_TABLE: nodejs-tutorial
    NEW_SIGNUP_TOPIC: '`{"Ref" : "NewSignupTopic"}`'
  aws:elasticbeanstalk:container:nodejs:
    ProxyServer: nginx
  aws:elasticbeanstalk:container:nodejs:staticfiles:
    /static: /static
  aws:autoscaling:asg:
    Cooldown: "120"
  aws:autoscaling:trigger:
    Unit: "Percent"
    Period: "1"
    BreachDuration: "2"
    UpperThreshold: "75"
    LowerThreshold: "30"
    MeasureName: "CPUUtilization"
```

Les configurations suivantes sont appliquées à l'application :

- L'adresse e-mail que la rubrique Amazon SNS utilise pour les notifications est définie sur votre adresse, ou celle que vous saisissez dans le fichier `options.config`.
- La table nodejs-tutorial sera utilisée à la place de celle créée par `.ebextensions/create-dynamodb-table.config`.

3. Supprimez `.ebextensions/create-dynamodb-table.config`.

```
~/nodejs-tutorial$ rm .ebextensions/create-dynamodb-table.config
```

La prochaine fois que vous déployez l'application, la table créée par ce fichier de configuration sera supprimée.

4. Déployez l'application mise à jour dans votre environnement Elastic Beanstalk avec la commande eb deploy.

```
~/nodejs-example-dynamo$ eb deploy
```

5. Une fois l'environnement créé, utilisez la commande eb open pour ouvrir l'URL de l'environnement dans le navigateur par défaut.

```
~/nodejs-example-dynamo$ eb open
```

Lors du déploiement, Elastic Beanstalk met à jour la configuration de la rubrique Amazon SNS et supprime la table DynamoDB créée lorsque vous avez déployé la première version de l'application.

Désormais, lorsque vous résiliez l'environnement, la table nodejs-tutorial n'est pas supprimée. Cela vous permet d'effectuer des déploiements bleu/vert, de modifier les fichiers de configuration ou d'arrêter votre site web sans risque de perte de données.

Ouvrez votre site dans un navigateur et vérifiez que le formulaire fonctionne comme prévu. Créez quelques entrées et vérifiez la table dans la console DynamoDB.

Pour afficher la table

1. Ouvrez la page Tables dans la console DynamoDB.
2. Recherchez la table nodejs-tutorial.
3. Sélectionnez la table, choisissez Items (Éléments), puis Start search (Commencer la recherche) afin d'afficher tous les éléments de la table.

Vous pouvez également voir qu'Elastic Beanstalk a supprimé la table créée précédemment.

Configuration de votre environnement pour une haute disponibilité

Enfin, configurez le groupe Auto Scaling de votre environnement avec un nombre minimum d'instances plus élevé. Exécutez au moins deux instances en permanence afin d'empêcher que les serveurs web de votre environnement constituent un point de défaillance unique et pour vous permettre de déployer des modifications sans mettre votre site hors service.

Pour configurer le groupe Auto Scaling de votre environnement pour une haute disponibilité

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Auto Scaling group (Groupe Auto Scaling), définissez les Min instances (Instances min.) sur **2**.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Nettoyage

Une fois que vous avez fini d'utiliser le code de démonstration, vous pouvez mettre fin à votre environnement. Elastic Beanstalk supprime toutes les ressources AWS associées, telles que les instances EC2 Amazon, les instances de base de données, les équilibreurs de charge, lesgroupes de sécurité et les alarmes.

La suppression de ressources n'entraîne pas la suppression de l'application Elastic Beanstalk. Vous pouvez donc créer de nouveaux environnements pour votre application à tout moment.

Pour mettre fin à votre environnement Elastic Beanstalk depuis la console

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Terminate Environment (Résilier l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Vous pouvez également supprimer les tables DynamoDB externes que vous avez créées.

Pour supprimer une table DynamoDB

1. Ouvrez la page Tables dans la console DynamoDB.
2. Sélectionnez une table.
3. Choisissez Actions, puis Delete table (Supprimer la table).
4. Choisissez Supprimer.

Étapes suivantes

L'exemple d'application utilise des fichiers de configuration pour configurer les paramètres logiciels et créer AWS des ressources dans le cadre de votre environnement. Pour de plus amples informations sur les fichiers de configuration et leur utilisation, veuillez consulter Personnalisation d'environnement avancée avec fichiers de configuration (`.ebextensions`).

L'exemple d'application de ce tutoriel utilise l'infrastructure web Express pour Node.js. Pour plus d'informations sur Express, consultez la documentation officielle à l'adresse expressjs.com.

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, configurez un nom de domaine personnalisé pour votre environnement et activez HTTPS pour des connexions sécurisées.

Ajout d'une instance de base de données Amazon RDS à votre environnement Node.js Elastic Beanstalk

Cette rubrique fournit des instructions pour créer un Amazon RDS à l'aide de la console Elastic Beanstalk. Vous pouvez utiliser une instance de base de données Amazon Relational Database Service (Amazon RDS) pour stocker les données collectées et modifiées par votre application. La base de données peut être liée à votre environnement et gérée par Elastic Beanstalk, ou elle peut être créée comme déliée et gérée en externe par un autre service. Dans ces instructions, la base de données est couplée à votre environnement et gérée par Elastic Beanstalk. Pour plus d'informations sur l'intégration d'un Amazon RDS avec Elastic Beanstalk, consultez Ajout d'une base de données à votre environnement Elastic Beanstalk.

Sections

- Ajout d'une instance de base de données à votre environnement
- Téléchargement d'un pilote
- Connexion à une base de données

Ajout d'une instance de base de données à votre environnement

Pour ajouter une instance DB à votre environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

L'ajout d'une instance DB prend environ 10 minutes. Une fois la mise à jour de l'environnement terminée, le nom d'hôte de l'instance DB et les autres informations de connexion sont disponibles dans votre application, via les propriétés d'environnement suivantes :

Nom de la propriété	Description	Valeur de la propriété
`RDS_HOSTNAME`	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
`RDS_PORT`	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
`RDS_DB_NAME`	Nom de la base de données, **ebdb**.	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
`RDS_USERNAME`	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
`RDS_PASSWORD`	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

Pour plus d'informations sur la configuration d'une instance de base de données liée à un environnement Elastic Beanstalk, consultez Ajout d'une base de données à votre environnement Elastic Beanstalk.

Téléchargement d'un pilote

Ajoutez le pilote de base de données au fichier `package.json` de votre projet sous `dependencies`.

Example **`package.json`** – Express avec MySQL

```
{
  "name": "my-app",
  "version": "0.0.1",
  "private": true,
  "dependencies": {
    "ejs": "latest",
    "aws-sdk": "latest",
    "express": "latest",
    "body-parser": "latest",
    "mysql": "latest"
  },
  "scripts": {
    "start": "node app.js"
  }
}
```

Packages de pilotes courants pour Node.js

- MySQL – mysql
- PostgreSQL – node-postgres
- SQL Server – node-mssql
- Oracle – node-oracledb

Connexion à une base de données

Elastic Beanstalk fournit des informations de connexion pour les instances de base de données attachées dans les propriétés de l'environnement. Utilisez `process.env.`*`VARIABLE`* pour lire les propriétés et configurer une connexion de base de données.

Example app.js – Connexion à une base de données MySQL

```
var mysql = require('mysql');
```

```
var connection = mysql.createConnection({
  host     : process.env.RDS_HOSTNAME,
  user     : process.env.RDS_USERNAME,
  password : process.env.RDS_PASSWORD,
  port     : process.env.RDS_PORT
});

connection.connect(function(err) {
  if (err) {
    console.error('Database connection failed: ' + err.stack);
    return;
  }

  console.log('Connected to database.');
});

connection.end();
```

Pour plus d'informations sur la construction d'une chaîne de connexion à l'aide de node-mysql, consultez npmjs. org/package/mysql.

Outils et ressources Node.js

Voici plusieurs ressources vous permettant d'obtenir une aide supplémentaire lors du développement de vos applications Node.js :

Ressource	Description
GitHub	Installez le AWS SDK pour Node.js à l'aide GitHub de.
AWS SDK pour Node.js (version préliminaire pour les développeurs)	Guichet unique pour l'exemple de code, la documenta tion, les outils et les ressources supplémentaires.

Déploiement d'applications PHP avec Elastic Beanstalk

Vous pouvez déployer une application PHP en quelques minutes à l'aide de l'interface de ligne de commande Elastic Beanstalk (EB CLI) ou de la console Elastic Beanstalk. Pour savoir comment procéder, commencez par le QuickStart didacticiel, puis consultez les exemples avancés.

Ressources pour déployer des applications PHP sur Elastic Beanstalk

- QuickStart pour PHP— Step-by-step instructions pour déployer une application PHP Hello World à l'aide de l'interface de ligne de commande EB.
- Utilisation de la plateforme PHP Elastic Beanstalk— Comment utiliser les fonctionnalités de la plateforme spécifiquement pour PHP.
- Exemples avancés— Tutoriels pratiques pour les frameworks et applications PHP courants, et comment ajouter une base de données Amazon RDS à votre environnement.

Pour plus d'informations sur le développement avec PHP dans PHP AWS, consultez les ressources suivantes :

- GitHub— Installez le AWS SDK pour PHP GitHub à l'aide de.
- Centre de développement PHP — Outils, documentation et exemples de code pour développer des applications PHP sur AWS.
- AWS SDK for FAQs PHP : obtenez des réponses aux questions fréquemment posées.

QuickStart: Déployer une application PHP sur Elastic Beanstalk

Dans le didacticiel suivant, vous allez apprendre à créer et à déployer un exemple d'application PHP dans un AWS Elastic Beanstalk environnement à l'aide de la CLI EB.

> ⚠ Non destiné à une utilisation en production
>
> Les exemples sont destinés uniquement à des fins de démonstration. N'utilisez pas d'exemples d'applications en production.

Sections

- Votre AWS compte

- Conditions préalables
- Étape 1 : Création d'une application PHP
- Étape 2 : Exécutez votre application localement
- Étape 3 : Initialisation et déploiement de votre application PHP
- Étape 4 : Parcourez votre application cloud
- Étape 5 : Mettez à jour et redéployez votre application
- Nettoyage
- Étapes suivantes

Votre AWS compte

Si vous n'êtes pas encore AWS client, vous devez créer un AWS compte pour utiliser Elastic Beanstalk.

Créez un AWS compte

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez l'https://portal.aws.amazon.com/billing/inscription.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d’inscription, vous recevrez un appel téléphonique ou un SMS et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

   Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l’utilisateur racine a accès à l’ensemble des Services AWS et des ressources de ce compte. La meilleure pratique de sécurité consiste à attribuer un accès administratif à un utilisateur, et à utiliser uniquement l’utilisateur racine pour effectuer les tâches nécessitant un accès utilisateur racine.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. À tout moment, vous pouvez consulter l'activité actuelle de votre compte et gérer votre compte en accédant à https://aws.amazon.com/et en choisissant Mon compte.

Création d’un utilisateur doté d’un accès administratif

Après vous être inscrit à un Compte AWS, sécurisez Utilisateur racine d'un compte AWS AWS IAM Identity Center, activez et créez un utilisateur administratif afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l’aide pour vous connecter en utilisant l’utilisateur racine, consultez Connexion en tant qu’utilisateur racine dans le Guide de l’utilisateur Connexion à AWS .
2. Activez l’authentification multifactorielle (MFA) pour votre utilisateur racine.

   Pour obtenir des instructions, voir Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d’un utilisateur doté d’un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d’ AWS IAM Identity Center dans le Guide de l’utilisateur AWS IAM Identity Center .
2. Dans IAM Identity Center, octroyez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connexion en tant qu’utilisateur doté d’un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l’URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l’utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribution d'un accès à d'autres utilisateurs

1. Dans IAM Identity Center, créez un ensemble d'autorisations qui respecte la bonne pratique consistant à appliquer les autorisations de moindre privilège.

   Pour obtenir des instructions, consultez Création d'un ensemble d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .

2. Attribuez des utilisateurs à un groupe, puis attribuez un accès par authentification unique au groupe.

   Pour obtenir des instructions, consultez Ajout de groupes dans le Guide de l'utilisateur AWS IAM Identity Center .

Conditions préalables

- Interface de ligne de commande Elastic Beanstalk - Pour l'installation, voir. Installation de la CLI EB avec un script de configuration (recommandé)
- PHP - Installez PHP sur votre machine locale en suivant les instructions d'installation et de configuration sur le site Web de PHP.

Étape 1 : Création d'une application PHP

Pour ce démarrage rapide, vous allez créer une application PHP Hello World.

Créez un répertoire de projet.

```
~$ mkdir eb-php
~$ cd eb-php
```

Créez ensuite un `index.php` fichier dans le répertoire du projet et ajoutez le code suivant.

Example **`index.php`**

```
<?php
  echo "Hello from a PHP application running in Elastic Beanstalk!";
?>
```

Étape 2 : Exécutez votre application localement

Utilisez la commande suivante pour exécuter votre application localement.

```
~$ php -S localhost:5000
```

Ouvrez un navigateur pour http://localhost:5000.

Vous devriez voir votre message d'accueil dans le navigateur et les messages de journal dans votre terminal.

Arrêtez le serveur local en entrant`Control+c`, afin de pouvoir déployer Elastic Beanstalk.

Étape 3 : Initialisation et déploiement de votre application PHP

Vous allez ensuite déployer votre application dans un environnement à l'aide de la console Elastic Beanstalk ou de l'EB CLI. Dans ce didacticiel, vous allez utiliser l'interface de ligne de commande EB avec l'option interactive pour initialiser un environnement.

Pour initialiser votre environnement et créer un environnement

1. Exécutez la commande suivante init.

   ```
   ~$ eb init -i
   ```

   La commande init crée une application de manière interactive. Le nom de l'application sera par défaut le dossier local qui est`eb-php`.

   Pour toutes les demandes, à l'exception de l'accès SSH, acceptez les valeurs par défaut pour créer un environnement avec la dernière version de la plateforme PHP. Pour le dépannage des instances, vous pouvez configurer l'accès SSH en réexécutant la `eb init -i` commande ultérieurement ou en vous connectant à l'aide d'Amazon EC2 Instance Connect ou de Session Manager.
2. Créez un environnement et déployez votre application

   Exécutez la commande suivante pour créer un environnement nommé`blue-env`.

   ```
   ~$ eb create blue-env
   ```

Lorsque vous exécutez la eb create commande pour la première fois, Elastic Beanstalk crée automatiquement un fichier zip de votre application, appelé bundle source. Elastic Beanstalk crée ensuite un environnement avec une ou plusieurs instances Amazon EC2, puis déploie l'application dans l'environnement.

Le déploiement de votre application sur Elastic Beanstalk peut prendre jusqu'à cinq minutes.

Étape 4 : Parcourez votre application cloud

Lorsque le processus de création de votre environnement est terminé, votre application devrait être en cours d'exécution et écouter les demandes sur le port 5000. Connectez-vous à votre application à l'aide de la commande suivante :

```
~$ eb open
```

La `eb open` commande ouvre un onglet de navigateur vers un sous-domaine personnalisé créé pour votre application.

Étape 5 : Mettez à jour et redéployez votre application

Après avoir créé une application et l'avoir déployée dans un environnement, vous pouvez déployer une nouvelle version de l'application ou une autre application à tout moment. Le déploiement d'une nouvelle version d'application est plus rapide car il ne nécessite ni le provisionnement ni le redémarrage des instances Amazon EC2.

Mettez à jour votre code PHP pour inclure la valeur REQUEST_TIME provenant de l'environnement du serveur :

```
<?php
  echo "Hello from a PHP application running in Elastic Beanstalk!";

  $timestamp = $_SERVER['REQUEST_TIME'];
  echo '<br/>Request time: ' . date('Y/m/d H:i:s', $timestamp);
?>
```

Redéployez votre code PHP sur Elastic Beanstalk à l'aide de la commande suivante :

```
~$ eb deploy
```

Lorsque vous exécutez eb deploy, l'interface de ligne de commande (CLI) EB crée un bundle avec le contenu de votre répertoire de projet et le déploie dans votre environnement.

Une fois le déploiement terminé, actualisez la page ou reconnectez-vous à votre application aveceb open. Vous devriez voir vos mises à jour. Si ce n'est pas le cas, réexécutez votre serveur local pour vérifier vos modifications.

> Félicitations !
>
> Vous avez créé, déployé et mis à jour une application PHP avec Elastic Beanstalk !

Nettoyage

Une fois que vous avez fini d'utiliser le code de démonstration, vous pouvez mettre fin à votre environnement. Elastic Beanstalk supprime AWS toutes les ressources associées, telles que les instances Amazon EC2, les instances de base de données, les équilibreursde charge, les groupes de sécurité et les alarmes.

La suppression de ressources n'entraîne pas la suppression de l'application Elastic Beanstalk. Vous pouvez donc créer de nouveaux environnements pour votre application à tout moment.

Pour mettre fin à votre environnement Elastic Beanstalk depuis la console

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Terminate Environment (Résilier l’environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Vous pouvez également mettre fin à votre environnement à l'aide de la CLI EB à l'aide de la commande suivante :

```
~$ eb terminate
```

Étapes suivantes

Vous pouvez explorer l'environnement de votre application à l'aide de la console Elastic Beanstalk. Pour plus d'informations, consultez la section Explorez votre environnement.

Pour des exemples avancés utilisant PHP, consultezExemples avancés de PHP dans Elastic Beanstalk.

Utilisation de la plateforme PHP Elastic Beanstalk

AWS Elastic Beanstalk fournit et prend en charge diverses branches de plate-forme pour différentes versions de PHP. Les plateformes prennent en charge les applications Web PHP qui s'exécutent de manière autonome ou sous Composer. Voir PHP dans le document AWS Elastic Beanstalk Platforms pour une liste complète des branches de plateforme prises en charge.

Elastic Beanstalk propose des options de configuration que vous pouvez utiliser pour personnaliser le logiciel qui s'exécute sur les instances Amazon EC2 de votre environnement Elastic Beanstalk. Vous pouvez configurer les variables d'environnement requises par votre application, activer la rotation des journaux sur Amazon S3, mapper les dossiers de la source de votre application contenant les fichiers statiques avec les chemins desservis par le serveur proxy, et définir les paramètres d'initialisation PHP communs.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour modifier la configuration d'un environnement en cours d'exécution. Pour éviter de perdre la configuration de votre environnement en le résiliant, vous pouvez utiliser des configurations enregistrées pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des fichiers de configuration. Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

Si vous utilisez Composer, vous pouvez inclure un fichier`composer.json` dans votre bundle source afin d'installer des packages pendant le déploiement.

Pour des paramètres avancés PHP et de configuration PHP non fournis en tant qu'options de configuration, vous pouvez utiliser des fichiers de configuration pour fournir un fichier `INI` qui peut étendre et remplacer les paramètres par défaut appliqués par Elastic Beanstalk, ou installer des extensions supplémentaires.

Les paramètres appliqués dans la console Elastic Beanstalk remplacent les mêmes paramètres des fichiers de configuration, s'ils existent. Cela vous permet d'utiliser les paramètres par défaut dans les fichiers de configuration et de les remplacer par des paramètres spécifiques à l'environnement dans la console. Pour plus d'informations sur la priorité et les autres méthodes de modification des paramètres, consultez Options de configuration.

Pour plus d'informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, consultez the section called “Extension des plateformes Linux”.

Sujets relatifs à la plateforme PHP

- Installation du AWS SDK pour PHP
- Considérations relatives à PHP 8.1 sur Amazon Linux 2
- Configuration de votre environnement PHP
- Espaces de noms pour la configuration
- Installation des dépendances de votre application PHP Elastic Beanstalk
- Mettre à jour Composer sur Elastic Beanstalk
- Extension du fichier php.ini dans votre configuration Elastic Beanstalk

Installation du AWS SDK pour PHP

Si vous devez gérer AWS des ressources depuis votre application, installez le AWS SDK pour PHP. Par exemple, avec le kit SDK pour PHP, vous pouvez utiliser Amazon DynamoDB (DynamoDB) pour stocker les informations relatives à l'utilisateur et à la session sans créer de base de données relationnelle.

Pour installer le SDK pour PHP avec Composer

```
$ composer require aws/aws-sdk-php
```

Pour plus d'informations, consultez la AWS SDK for PHPpage d'accueil. Pour obtenir des instructions, voir Installer le AWS SDK for PHP.

Considérations relatives à PHP 8.1 sur Amazon Linux 2

Veuillez lire cette section si vous utilisez la branche de plateforme PHP 8.1 sur Amazon Linux 2.

Considérations relatives à PHP 8.1 sur Amazon Linux 2

> **Note**
>
> Les informations de cette rubrique s'appliquent uniquement à la branche de plateforme PHP 8.1 sur Amazon Linux 2. Cela ne s'applique pas aux branches de la plateforme PHP basées sur AL2023. Elles ne s'appliquent pas non plus à la branche de plateforme PHP 8.0 Amazon Linux 2.

Elastic Beanstalk stocke les packages RPM associés à PHP 8.1 pour la branche de plateforme PHP 8.1 sur Amazon Linux 2 sur les instances EC2 dans un répertoire local, au lieu du référentiel Amazon Linux. Vous pouvez utiliser rpm -i afin d'installer des packages. À partir de la version 3.5.0 de la plateforme PHP 8.1, Elastic Beanstalk stocke les packages RPM associés à PHP 8.1 dans le répertoire EC2 local suivant.

`/opt/elasticbeanstalk/RPMS`

L'exemple suivant permet d'installer le package php-debuginfo.

```
$rpm -i /opt/elasticbeanstalk/RPMS/php-debuginfo-8.1.8-1.amzn2.x86_64.rpm
```

La version dans le nom du package varie en fonction de celle répertoriée dans le répertoire local EC2 `/opt/elasticbeanstalk/RPMS`. Utilisez la même syntaxe afin d'installer d'autres packages RPM PHP 8.1.

Développez la section suivante afin d'afficher une liste des packages RPM que nous fournissons.

Packages RPM

La liste suivante présente les packages RMP fournis par la plateforme PHP 8.1 Elastic Beanstalk sur Amazon Linux 2. Ils se situent dans le répertoire local `/opt/elasticbeanstalk/RPMS`.

Les numéros de version 8.1.8-1 et 3.7.0-1 figurant dans les noms de packages listés ne sont qu'un exemple.

- `php-8.1.8-1.amzn2.x86_64.rpm`
- `php-bcmath-8.1.8-1.amzn2.x86_64.rpm`

- php-cli-8.1.8-1.amzn2.x86_64.rpm
- php-common-8.1.8-1.amzn2.x86_64.rpm
- php-dba-8.1.8-1.amzn2.x86_64.rpm
- php-dbg-8.1.8-1.amzn2.x86_64.rpm
- php-debuginfo-8.1.8-1.amzn2.x86_64.rpm
- php-devel-8.1.8-1.amzn2.x86_64.rpm
- php-embedded-8.1.8-1.amzn2.x86_64.rpm
- php-enchant-8.1.8-1.amzn2.x86_64.rpm
- php-fpm-8.1.8-1.amzn2.x86_64.rpm
- php-gd-8.1.8-1.amzn2.x86_64.rpm
- php-gmp-8.1.8-1.amzn2.x86_64.rpm
- php-intl-8.1.8-1.amzn2.x86_64.rpm
- php-ldap-8.1.8-1.amzn2.x86_64.rpm
- php-mbstring-8.1.8-1.amzn2.x86_64.rpm
- php-mysqlnd-8.1.8-1.amzn2.x86_64.rpm
- php-odbc-8.1.8-1.amzn2.x86_64.rpm
- php-opcache-8.1.8-1.amzn2.x86_64.rpm
- php-pdo-8.1.8-1.amzn2.x86_64.rpm
- php-pear-1.10.13-1.amzn2.noarch.rpm
- php-pgsql-8.1.8-1.amzn2.x86_64.rpm
- php-process-8.1.8-1.amzn2.x86_64.rpm
- php-pspell-8.1.8-1.amzn2.x86_64.rpm
- php-snmp-8.1.8-1.amzn2.x86_64.rpm
- php-soap-8.1.8-1.amzn2.x86_64.rpm
- php-sodium-8.1.8-1.amzn2.x86_64.rpm
- php-xml-8.1.8-1.amzn2.x86_64.rpm
- php-pecl-imagick-3.7.0-1.amzn2.x86_64.rpm
- php-pecl-imagick-debuginfo-3.7.0-1.amzn2.x86_64.rpm

- `php-pecl-imagick-devel-3.7.0-1.amzn2.noarch.rpm`

Vous pouvez utiliser les packages PEAR et PECL afin d'installer des extensions courantes. Pour plus d'informations sur PEAR, veuillez consulter le site Web PEAR - PHP Extension and Application Repository (PEAR : référentiel d'applications et extension PHP) (français non garanti). Pour plus d'informations sur PECL, veuillez consulter le site Web de PECL extension (L'extension PECL) (français non garanti).

Les exemples de commandes suivants permettent d'installer les extensions Memcached.

```
$pecl install memcache
```

Vous pouvez également utiliser ce qui suit :

```
$pear install pecl/memcache
```

Les exemples de commandes suivants permettent d'installer les extensions Redis.

```
$pecl install redis
```

Vous pouvez également utiliser ce qui suit :

```
$pear install pecl/redis
```

Configuration de votre environnement PHP

Vous pouvez utiliser la console Elastic Beanstalk pour activer la rotation de journal sur Amazon S3, configurer des variables que votre application peut lire depuis l'environnement et modifier des paramètres PHP.

Pour configurer votre environnement PHP dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.

4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.

Paramètres PHP

- Proxy server (Serveur proxy) – Serveur proxy à utiliser sur vos instances d'environnement. Le serveur nginx est utilisé par défaut.
- Racine du document – Dossier qui contient la page par défaut de votre site. Si votre page d'accueil n'est pas à la racine de votre groupe source, spécifiez le dossier qui la contient par rapport aux chemin d'accès racine. Par exemple, `/public` si la page d'accueil est dans un dossier nommé `public`.
- Limite de mémoire – Volume maximum de mémoire qu'un script est autorisé à allouer. Par exemple, `512M`.
- Compression de sortie Zlib – Définissez sur `On` pour compresser les réponses.
- Permettre de faire un fopen d'une URL – Définissez cette option sur `Off` pour empêcher les scripts de télécharger des fichiers à partir d'emplacements distants.
- Afficher les erreurs – Définissez cette option sur `On` pour afficher les messages d'erreur internes à des fins de débogage.
- Durée d'exécution maximum – La durée maximale en secondes pendant laquelle un script est autorisé à s'exécuter avant que l'environnement ne l'arrête.

Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Enable log file rotation to Amazon S3 (Permettre la rotation du fichier journal sur Amazon S3) – Indique si les fichiers journaux des instances Amazon EC2 de votre application doivent être copiés dans le compartiment Amazon S3 associé à votre application.

Fichiers statiques

Pour améliorer les performances, la section des Fichiers statiques vous permet de configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un

ensemble de répertoires dans votre application web. Pour chaque répertoire, vous définissez le chemin virtuel sur le mappage de répertoires. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Pour en savoir plus sur la configuration des fichiers statiques à l'aide des fichiers de configuration ou de la console Elastic Beanstalk, consultez the section called “Fichiers statiques”.

Propriétés de l'environnement

La section Propriétés de l'environnement vous permet de spécifier des paramètres de configuration de l'environnement sur les instances Amazon EC2 exécutant votre application. Ces paramètres sont passés en tant que paires clé-valeur à l'application.

Votre code d'application peut accéder aux propriétés de l'environnement à l'aide de `$_SERVER` ou à la fonction `get_cfg_var`.

```
$endpoint = $_SERVER['API_ENDPOINT'];
```

Pour plus d’informations, consultez Variables d'environnement et autres paramètres du logiciel.

Espaces de noms pour la configuration

Vous pouvez utiliser un fichier de configuration pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être spécifiques à la plate-forme ou s'appliquer à toutes les plateformes du service Elastic Beanstalk dans son ensemble. Les options de configuration sont organisées en espaces de noms.

Les espaces de noms suivants configurent à la fois votre service proxy et les options spécifiques à PHP :

- `aws:elasticbeanstalk:environment:proxy:staticfiles`— configurez le proxy d'environnement pour servir les fichiers statiques. Vous définissez les mappages des chemins virtuels vers les répertoires d'applications.
- `aws:elasticbeanstalk:environment:proxy`— spécifiez le serveur proxy de l'environnement.
- `aws:elasticbeanstalk:container:php:phpini`— configure les options spécifiques à PHP. Cet espace de noms inclut `composer_options` ce qui n'est pas disponible sur la console Elastic Beanstalk. Cette option définit les options personnalisées à utiliser lors de l'installation

de dépendances à l'aide de Composer via la `composer.phar install` commande. Pour plus d'informations sur cette commande, y compris les options disponibles, consultez la section installation sur le site Web getcomposer.org.

L'exemple de fichier de configuration suivant spécifie une option de fichiers statiques qui mappe un répertoire nommé `staticimages` au chemin `/images`, d'accès et affiche les paramètres de chacune des options disponibles dans l'espace de noms `aws:elasticbeanstalk:container:php:phpini` :

Example.ebextensions/php-settings.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache
  aws:elasticbeanstalk:environment:proxy:staticfiles:
    /images: staticimages
  aws:elasticbeanstalk:container:php:phpini:
    document_root: /public
    memory_limit: 128M
    zlib.output_compression: "Off"
    allow_url_fopen: "On"
    display_errors: "Off"
    max_execution_time: 60
    composer_options: vendor/package
```

> **Note**
>
> L'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles` n'est pas défini sur les branches PHP de l'AMI Amazon Linux (précédant Amazon Linux 2).

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d' AWS CLI. Pour plus d’informations, consultez Options de configuration.

Installation des dépendances de votre application PHP Elastic Beanstalk

Cette rubrique décrit comment configurer votre application pour installer les autres packages PHP dont elle a besoin. Votre application peut avoir des dépendances sur d'autres packages PHP. Vous

pouvez configurer votre application pour installer ces dépendances sur les instances Amazon Elastic Compute Cloud (Amazon EC2) de l'environnement. Vous pouvez également inclure les dépendances de votre application dans le bundle source et les déployer avec l'application. La section ci-après décrit chacune de ces méthodes.

Utilisation d'un fichier Composer pour installer des dépendances sur les instances

Utilisez un fichier `composer.json` à la racine de votre source de projet afin d'utiliser Composer pour installer les packages dont votre application a besoin sur les instances Amazon EC2 de votre environnement.

Example composer.json

```
{
    "require": {
        "monolog/monolog": "1.0.*"
    }
}
```

Lorsqu'un fichier `composer.json` est présent, Elastic Beanstalk exécute `composer.phar install` pour installer les dépendances. Vous pouvez ajouter des options à joindre à la commande en définissant l'option `composer_options` dans l'espace de noms `aws:elasticbeanstalk:container:php:phpini`.

Inclusion des dépendances dans la solution groupée source

Si votre application possède un grand nombre de dépendances, leur installation peut prendre beaucoup de temps. Il peut en résulter un nombre accru d'opérations de déploiement et de mise à l'échelle, car les dépendances sont installées sur chaque nouvelle instance.

Pour éviter d'accroître le temps de déploiement, utilisez Composer dans votre environnement de développement afin de résoudre les dépendances et de les installer dans le dossier `vendor`.

Pour inclure des dépendances dans le bundle source de votre application

1. Exécutez la commande suivante :

   ```
   % composer install
   ```

2. Incluez le dossier `vendor` généré dans la racine du bundle source de votre application.

Lorsque Elastic Beanstalk trouve un dossier `vendor` sur l'instance, il ignore le fichier `composer.json` (même s'il existe). Votre application utilise alors les dépendances issues du dossier `vendor`.

Mettre à jour Composer sur Elastic Beanstalk

Cette rubrique décrit comment configurer Elastic Beanstalk pour maintenir Composer à jour. Vous devrez peut-être mettre à jour Composer si un message d'erreur s'affiche lorsque vous essayez d'installer des packages avec un fichier Composer, ou si vous ne parvenez pas à utiliser la dernière version de la plateforme. Entre les mises à jour de la plateforme, vous pouvez mettre à jour Composer dans les instances de votre environnement en utilisant les fichiers de configuration de votre `.ebextensions`dossier.

Vous pouvez mettre à jour automatiquement Composer avec la configuration suivante.

```
commands:
  01updateComposer:
    command: /usr/bin/composer.phar self-update 2.7.0
```

Le paramètre d'option suivant définit la variable d'`COMPOSER_HOME`environnement, qui configure l'emplacement du cache Composer.

```
option_settings:
  - namespace: aws:elasticbeanstalk:application:environment
    option_name: COMPOSER_HOME
    value: /home/webapp/composer-home
```

Vous pouvez combiner les deux dans le même fichier de configuration de votre `.ebextensions` dossier.

Example.ebextensions/composer.config

```
commands:
  01updateComposer:
    command: /usr/bin/composer.phar self-update 2.7.0

option_settings:
  - namespace: aws:elasticbeanstalk:application:environment
    option_name: COMPOSER_HOME
    value: /home/webapp/composer-home
```

> **Note**
>
> En raison des mises à jour apportées à l'installation de Composer dans les versions de AL2023 plateforme du 22 février 2024 et du 28 février 2024, AL2 la mise à jour automatique de Composer peut échouer si elle `COMPOSER_HOME` est définie au moment de l'exécution de la mise à jour automatique.
> Les commandes combinées suivantes ne pourront pas être exécutées : `export COMPOSER_HOME=/home/webapp/composer-home && /usr/bin/composer.phar self-update 2.7.0`
> Cependant, l'exemple précédent fonctionnera. Dans l'exemple précédent, le paramètre d'option pour ne `COMPOSER_HOME` sera pas transmis à l'`01updateComposer`exécution, et il ne sera pas défini lors de l'exécution de la commande de mise à jour automatique.

> **Important**
>
> Si vous n'indiquez pas le numéro de version dans la commande `composer.phar self-update`, Composer met à jour vers la dernière version disponible chaque fois que vous déployez votre code source et que de nouvelles instances sont provisionnées par Auto Scaling. Si une version de Composer est disponible qui est incompatible avec votre application, elle peut provoquer l'échec des opérations de dimensionnement et des déploiements.

Pour plus d'informations sur les plateformes PHP Elastic Beanstalk, ainsi que sur la version de Composer, consultez la section Versions des plateformes PHP du document Plateformes AWS Elastic Beanstalk .

Extension du fichier php.ini dans votre configuration Elastic Beanstalk

Utilisez un fichier de configuration avec un bloc `files` pour ajouter un fichier `.ini` à `/etc/php.d/` sur les instances de votre environnement. Le fichier de configuration principal, `php.ini`, recherche des paramètres dans des fichiers dans ce dossier dans l'ordre alphabétique. De nombreuses extensions sont activées par défaut par les fichiers dans ce dossier.

Example.ebextensions/mongo.config

```
files:
```

```
  "/etc/php.d/99mongo.ini":
    mode: "000755"
    owner: root
    group: root
    content: |
      extension=mongo.so
```

Exemples avancés de PHP dans Elastic Beanstalk

Pour commencer à utiliser les applications PHP AWS Elastic Beanstalk, vous avez besoin d'un bundle de sources d'applications à télécharger en tant que première version d'application à déployer dans un environnement.

Nous vous recommandons QuickStart pour PHP de démarrer avec une simple application PHP déployée avec l'EB CLI.

Exemples PHP avancés

- Ajouter une instance de base de données Amazon RDS à votre environnement PHP Elastic Beanstalk
- Déploiement d'une application Laravel sur Elastic Beanstalk
- Déploiement d'une application CakePHP sur Elastic Beanstalk
- Déploiement d'une application Symfony sur Elastic Beanstalk
- Déploiement d'une application PHP haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk
- Déploiement d'un WordPress site Web à haute disponibilité avec une base de données Amazon RDS externe sur Elastic Beanstalk
- Déploiement d'un site web Drupal haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk

Ajouter une instance de base de données Amazon RDS à votre environnement PHP Elastic Beanstalk

Cette rubrique fournit des instructions pour créer un Amazon RDS à l'aide de la console Elastic Beanstalk. Vous pouvez utiliser une instance de base de données Amazon Relational Database Service (Amazon RDS) pour stocker les données collectées et modifiées par votre application. La

base de données peut être liée à votre environnement et gérée par Elastic Beanstalk, ou elle peut être créée comme déliée et gérée en externe par un autre service. Dans ces instructions, la base de données est couplée à votre environnement et gérée par Elastic Beanstalk. Pour plus d'informations sur l'intégration d'un Amazon RDS avec Elastic Beanstalk, consultez Ajout d'une base de données à votre environnement Elastic Beanstalk.

Sections

- Ajout d'une instance de base de données à votre environnement
- Téléchargement d'un pilote
- Connexion à une base de données à l'aide d'un PDO ou d'un My SQLi
- Connexion à une base de données avec Symfony

Ajout d'une instance de base de données à votre environnement

Pour ajouter une instance DB à votre environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

L'ajout d'une instance DB prend environ 10 minutes. Une fois la mise à jour de l'environnement terminée, le nom d'hôte de l'instance DB et les autres informations de connexion sont disponibles dans votre application, via les propriétés d'environnement suivantes :

Nom de la propriété	Description	Valeur de la propriété
`RDS_HOSTNAME`	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console

Nom de la propriété	Description	Valeur de la propriété
		Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, **ebdb**.	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

Pour plus d'informations sur la configuration d'une instance de base de données liée à un environnement Elastic Beanstalk, consultez Ajout d'une base de données à votre environnement Elastic Beanstalk.

Téléchargement d'un pilote

Pour utiliser des objets de données PHP (PDO) pour la connexion à la base de données, installez le pilote correspondant au moteur de base de données que vous avez choisi.

- MySQL – PDO_MYSQL
- PostgreSQL – PDO_PGSQL
- Oracle – PDO_OCI
- SQL Server – PDO_SQLSRV

Pour de plus amples informations, veuillez consulter http://php.net/manual/en/pdo.installation.php.

Connexion à une base de données à l'aide d'un PDO ou d'un My SQLi

Vous pouvez utiliser `$_SERVER[`VARIABLE`]` afin de lire les informations de connexion depuis l'environnement.

Pour un PDO, créez un nom de source de données (DSN) à partir de l'hôte, du port et du nom. Transmettez le DSN au constructeur du PDO avec le nom d'utilisateur de base de données et le mot de passe.

Example Connexion à une base de données RDS avec PDO - MySQL

```
<?php
$dbhost = $_SERVER['RDS_HOSTNAME'];
$dbport = $_SERVER['RDS_PORT'];
$dbname = $_SERVER['RDS_DB_NAME'];
$charset = 'utf8' ;

$dsn = "mysql:host={$dbhost};port={$dbport};dbname={$dbname};charset={$charset}";
$username = $_SERVER['RDS_USERNAME'];
$password = $_SERVER['RDS_PASSWORD'];

$pdo = new PDO($dsn, $username, $password);
?>
```

Pour les autres pilotes, remplacez `mysql` par le nom de votre pilote – `pgsql`, `oci` ou `sqlsrv`.

Pour MySQLi, transmettez le nom d'hôte, le nom d'utilisateur, le mot de passe, le nom de base de données et le port au `mysqli` constructeur.

Example Connexion à une base de données RDS avec mysqli_connect()

```
$link = new mysqli($_SERVER['RDS_HOSTNAME'], $_SERVER['RDS_USERNAME'],
 $_SERVER['RDS_PASSWORD'], $_SERVER['RDS_DB_NAME'], $_SERVER['RDS_PORT']);
```

Connexion à une base de données avec Symfony

Pour Symfony version 3.2 et versions ultérieures, vous pouvez utiliser `%env(`*PROPERTY_NAME*`)%` pour définir les paramètres de base de données dans un fichier de configuration en fonction des propriétés d'environnement définies par Elastic Beanstalk.

Example app/config/parameters.yml

```
parameters:
    database_driver:   pdo_mysql
    database_host:     '%env(RDS_HOSTNAME)%'
    database_port:     '%env(RDS_PORT)%'
    database_name:     '%env(RDS_DB_NAME)%'
    database_user:     '%env(RDS_USERNAME)%'
    database_password: '%env(RDS_PASSWORD)%'
```

Pour plus d'informations, consultez External Parameters (Symfony 3.4).

Pour les versions antérieures de Symfony, les variables d'environnement sont uniquement accessibles si elles commencent par `SYMFONY__`. Cela signifie que les propriétés d'environnement définies par Elastic Beanstalk ne sont pas accessibles, et vous devez définir vos propres propriétés d'environnement pour transmettre les informations de connexion à Symfony.

Pour vous connecter à une base de données avec Symfony 2, créez une propriété d'environnement pour chaque paramètre. Ensuite, utilisez `%property.name%` pour accéder à la variable transformée par Symfony dans un fichier de configuration. Par exemple, une propriété d'environnement nommée `SYMFONY__DATABASE__USER` est accessible en tant que `database.user`.

```
    database_user:     "%database.user%"
```

Pour plus d'informations, consultez External Parameters (Symfony 2.8).

Déploiement d'une application Laravel sur Elastic Beanstalk

Laravel est un framework open source model-view-controller (MVC) pour PHP. Ce didacticiel vous explique le processus de génération d'une application Laravel, de son déploiement dans un AWS Elastic Beanstalk environnement et de sa configuration pour se connecter à une instance de base de données Amazon Relational Database Service (Amazon RDS).

Sections

- Conditions préalables
- Lancer un environnement Elastic Beanstalk
- Installation de Laravel et génération d'un site web
- Déploiement de votre application

- Configuration des paramètres Composer
- Ajout d'une base de données à votre environnement
- Nettoyage
- Étapes suivantes

Conditions préalables

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans Découvrez comment démarrer avec Elastic Beanstalk pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

Laravel 6 nécessite PHP 7.2 ou version ultérieure. Il requiert également les extensions PHP répertoriées dans la rubrique Prérequis du serveur de la documentation officielle de Laravel. Suivez les instructions pour installer PHP et Composer.

Pour plus d'informations sur la prise en charge et la maintenance de Laravel, consultez la rubrique Stratégie de prise en charge de la documentation officielle de Laravel.

Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Choisissez la plateforme PHP et acceptez les paramètres par défaut et l'exemple de code.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk à l'aide de ce lien préconfiguré : console.aws.amazon.com/elasticbeanstalk/home#/newApplication? Nom de l'application = Tutoriels et type d'environnement = LoadBalanced

2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Create app (Créer une application).

La création d'un environnement prend environ 5 minutes et crée les ressources suivantes :

- EC2 instance — Une machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications Web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.
- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Équilibreur de charge – Équilibreur de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité d'équilibrage de charge : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibreur de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et qui sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.

- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

> **Sécurité du domaine**
>
> Pour renforcer la sécurité de vos applications Elastic Beanstalk, le domaine elasticbeanstalk.com est enregistré dans la liste des suffixes publics (PSL).
> Si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Elastic Beanstalk, nous vous recommandons d'utiliser des cookies `__Host-` avec un préfixe pour une sécurité accrue. Cette pratique protège votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

> **Note**
>
> Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon S3.

Installation de Laravel et génération d'un site web

Composer peut installer Laravel et créer un projet de travail avec une commande :

```
~$ composer create-project --prefer-dist laravel/laravel eb-laravel
```

Composer installe Laravel et ses dépendances et génère un projet par défaut.

Si vous rencontrez des problèmes lors de l'installation de Laravel, consultez le sujet d'installation dans la documentation officielle : https://laravel.com/docs/6.x.

Déploiement de votre application

Créez un bundle source contenant les fichiers créés par Composer. La commande suivante permet de créer une solution groupée source nommée `laravel-default.zip`. Elle exclut les fichiers du dossier `vendor`, lesquels prennent beaucoup de place et ne sont pas nécessaires pour le déploiement de votre application dans Elastic Beanstalk.

```
~/eb-laravel$ zip ../laravel-default.zip -r * .[^.]* -x "vendor/*"
```

Téléchargez l'offre groupée source sur Elastic Beanstalk pour déployer Laravel dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

> **Note**
>
> En vue d'optimiser davantage le bundle source, initialisez un référentiel Git et utilisez la commande `git archive` pour créer le bundle source. Le projet Laravel par défaut inclut un fichier `.gitignore` qui indique à Git d'exclure le dossier `vendor` et d'autres fichiers qui ne sont pas nécessaires pour le déploiement.

Configuration des paramètres Composer

Lorsque le déploiement est terminé, cliquez sur l'URL pour ouvrir votre application Laravel dans le navigateur :

De quoi s'agit-il ? Par défaut, Elastic Beanstalk constitue la racine de votre projet sur le chemin d'accès racine du site Web. Dans ce cas, cependant, la page par défaut (`index.php`) est un niveau en-dessous dans le dossier `public`. Vous pouvez vérifier cela en ajoutant `/public` à l'URL. Par exemple, `http://laravel.us-east-2.elasticbeanstalk.com/public`.

Pour servir l'application Laravel à la racine, utilisez la console Elastic Beanstalk afin de configurer la racine du document du site Web.

Pour configurer la racine du document de votre site Web

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Pour Racine du document, entrez **`/public`**.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.
7. Lorsque la mise à jour est terminée, cliquez sur l'URL pour rouvrir votre site dans le navigateur.

Jusqu'ici, tout va bien. Vous ajoutez ensuite une base de données à votre environnement et vous configurez Laravel pour s'y connecter.

Ajout d'une base de données à votre environnement

Lancez une instance de base de données RDS dans votre environnement Elastic Beanstalk. Vous pouvez utiliser des bases de données MySQL ou PostgreSQL avec Laravel sur Elastic Beanstalk. SQLServer Pour cet exemple, nous utiliserons MySQL.

Pour ajouter une instance de base de données RDS à votre environnement Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Pour Moteur, choisissez mysql.
6. Saisissez un nom d'utilisateur principal et un mot de passe. Elastic Beanstalk fournit ces valeurs à votre application en utilisant les propriétés d'environnement.
7. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

La création d'une instance de base de données prend environ 10 minutes. Pour plus d'informations sur les bases de données couplées à un environnement Elastic Beanstalk, consultez Ajout d'une base de données à votre environnement Elastic Beanstalk.

En attendant, vous pouvez mettre à jour votre code source afin de lire les informations de connexion depuis l'environnement. Elastic Beanstalk fournit des détails de connexion en utilisant les variables d'environnement telles que `RDS_HOSTNAME` auxquelles vous pouvez accéder depuis votre application.

La configuration de la base de données Laravel est stockée dans un fichier nommé `database.php` dans le dossier `config` dans votre code de projet. Recherchez l'entrée `mysql` et modifiez les variables `host`, `database`, `username` et `and password` pour lire les valeurs correspondantes à partir d'Elastic Beanstalk :

Example~/Eb- .php laravel/config/database

```
...
    'connections' => [

        'sqlite' => [
            'driver' => 'sqlite',
            'database' => env('DB_DATABASE', database_path('database.sqlite')),
            'prefix' => '',
        ],

        'mysql' => [
            'driver' => 'mysql',
            'host' => env('RDS_HOSTNAME', '127.0.0.1'),
            'port' => env('RDS_PORT', '3306'),
            'database' => env('RDS_DB_NAME', 'forge'),
            'username' => env('RDS_USERNAME', 'forge'),
            'password' => env('RDS_PASSWORD', ''),
            'unix_socket' => env('DB_SOCKET', ''),
            'charset' => 'utf8mb4',
            'collation' => 'utf8mb4_unicode_ci',
            'prefix' => '',
            'strict' => true,
            'engine' => null,
        ],
...
```

Pour vérifier que la connexion de base de données est configurée correctement, ajoutez le code à `index.php` pour vous connecter à la base de données et ajoutez du code à la réponse par défaut :

Example~/Eb- .php laravel/public/index

```
...
if(DB::connection()->getDatabaseName())
{
   echo "Connected to database ".DB::connection()->getDatabaseName();
}
$response->send();
...
```

Lorsque l'instance DB a terminé son lancement, regroupez et déployez l'application mise à jour dans votre environnement.

Pour mettre à jour votre environnement Elastic Beanstalk

1. Créez un nouveau groupe source :

```
~/eb-laravel$ zip ../laravel-v2-rds.zip -r * .[^.]* -x "vendor/*"
```

2. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
3. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
4. Choisissez Upload and Deploy (Charger et déployer).
5. Choisissez Browse (Parcourir) et chargez `laravel-v2-rds.zip`.
6. Choisissez Deploy (Déployer).

Le déploiement d'une nouvelle version de votre application prend moins d'une minute. Lorsque le déploiement est terminé, actualisez la page web à nouveau pour vérifier que la connexion de base de données a abouti :

Nettoyage

Une fois que vous avez fini d'utiliser le code de démonstration, vous pouvez mettre fin à votre environnement. Elastic Beanstalk supprime toutes les ressources AWS associées, telles que les instances EC2 Amazon, les instances de base de données, les équilibreurs de charge, lesgroupes de sécurité et les alarmes.

La suppression de ressources n'entraîne pas la suppression de l'application Elastic Beanstalk. Vous pouvez donc créer de nouveaux environnements pour votre application à tout moment.

Pour mettre fin à votre environnement Elastic Beanstalk depuis la console

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Terminate Environment (Résilier l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Vous pouvez également arrêter les ressources de base de données que vous avez créées hors de votre environnement Elastic Beanstalk. Lorsque vous résiliez une instance de base de données Amazon RDS, vous pouvez prendre un instantané et restaurer les données dans une autre instance ultérieurement.

Pour résilier votre instance DB RDS

1. Ouvrez la console Amazon RDS.
2. Choisissez Databases (Bases de données).
3. Sélectionnez votre instance DB.
4. Choisissez Actions, puis Supprimer.
5. Choisissez si vous souhaitez créer un instantané, puis choisissez Delete (Supprimer).

Étapes suivantes

Pour plus d'informations sur Laravel, consultez le site officiel à l'adresse laravel.com.

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le charger manuellement sur la console Elastic Beanstalk. L'interface de ligne de commande Elastic Beanstalk (EB CLI easy-to-use) fournit des commandes pour créer, configurer et déployer des applications dans les environnements Elastic Beanstalk à partir de la ligne de commande.

Dans ce tutoriel, vous avez utilisé la console Elastic Beanstalk pour configurer les options du composeur. Pour que cette configuration fasse partie de votre source d'application, vous pouvez utiliser un fichier de configuration comme celui qui suit.

Example.ebextensions/composer.config

```
option_settings:
  aws:elasticbeanstalk:container:php:phpini:
    document_root: /public
```

Pour de plus amples informations, veuillez consulter Personnalisation d'environnement avancée avec fichiers de configuration (`.ebextensions`).

L'exécution d'une instance de base de données Amazon RDS dans votre environnement Elastic Beanstalk est excellente pour le développement et les tests, mais elle lie le cycle de vie de votre base de données à votre environnement. Pour de plus amples informations sur la connexion à une base de données s'exécutant en dehors de votre environnement, consultez Ajouter une instance de base de données Amazon RDS à votre environnement PHP Elastic Beanstalk.

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez configurer un nom de domaine personnalisé pour votre environnement et activer HTTPS pour des connexions sécurisées.

Déploiement d'une application CakePHP sur Elastic Beanstalk

CakePHP est une infrastructure MVC, open source pour PHP. Ce tutoriel vous guide à travers le processus permettant la génération d'un projet CakePHP, sont déploiement dans un environnement Elastic Beanstalk et sa configuration pour vous connecter à une instance de base de données Amazon RDS.

Sections

- Conditions préalables
- Lancer un environnement Elastic Beanstalk
- Installation de CakePHP et génération d'un site web
- Déploiement de votre application
- Ajout d'une base de données à votre environnement
- Nettoyage
- Étapes suivantes

Conditions préalables

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans Découvrez comment démarrer avec Elastic Beanstalk pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

CakePHP 4 nécessite PHP 7.4 ou version ultérieure. Il requiert également les extensions PHP répertoriées dans la documentation d'installation officielle de CakePHP. Vous devez installer PHP et Composer.

Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Choisissez la plateforme PHP et acceptez les paramètres par défaut et l'exemple de code.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk à l'aide de ce lien préconfiguré : console.aws.amazon.com/elasticbeanstalk/home#/newApplication? Nom de l'application = Tutoriels et type d'environnement = LoadBalanced
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Create app (Créer une application).

La création d'un environnement prend environ 5 minutes et crée les ressources suivantes :

- EC2 instance — Une machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications Web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.
- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Équilibreur de charge – Équilibreur de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité d'équilibrage de charge : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibreur de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et qui sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

> **Sécurité du domaine**
>
> Pour renforcer la sécurité de vos applications Elastic Beanstalk, le domaine elasticbeanstalk.com est enregistré dans la liste des suffixes publics (PSL).
> Si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Elastic Beanstalk, nous vous recommandons d'utiliser des cookies `__Host-` avec un préfixe pour une sécurité accrue. Cette pratique protège votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

> **Note**
>
> Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon S3.

Installation de CakePHP et génération d'un site web

Composer peut installer CakePHP et créer un projet de travail avec une commande :

```
~$ composer create-project --prefer-dist cakephp/app eb-cake
```

Composer installe CakePHP et environ 20 dépendances et génère un projet par défaut.

Si vous rencontrez des problèmes lors de l'installation de CakePHP, consultez la rubrique consacrée à l'installation dans la documentation officielle : http://book.cakephp.org/4.0/en/installation.html

Déploiement de votre application

Créez un bundle source contenant les fichiers créés par Composer. La commande suivante permet de créer une solution groupée source nommée `cake-default.zip`. Elle exclut les fichiers du dossier `vendor`, lesquels prennent beaucoup de place et ne sont pas nécessaires pour le déploiement de votre application dans Elastic Beanstalk.

```
eb-cake zip ../cake-default.zip -r * .[^.]* -x "vendor/*"
```

Téléchargez le bundle source sur Elastic Beanstalk pour déployer CakePHP dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

> **Note**
>
> En vue d'optimiser davantage le bundle source, initialisez un référentiel Git et utilisez la commande `git archive` pour créer le bundle source. Le projet Symfony par défaut inclut un fichier `.gitignore` qui indique à Git d'exclure le dossier `vendor` et d'autres fichiers qui ne sont pas nécessaires pour le déploiement.

Au terme du processus, cliquez sur l'URL pour ouvrir votre application CakePHP dans le navigateur.

Jusqu'ici, tout va bien. Vous ajoutez ensuite une base de données à votre environnement et vous configurez CakePHP pour s'y connecter.

Ajout d'une base de données à votre environnement

Lancez une instance de base de données Amazon RDS dans votre environnement Elastic Beanstalk. Vous pouvez utiliser des bases de données MySQL ou PostgreSQL avec CakePHP sur Elastic Beanstalk. SQLServer Pour cet exemple, nous utiliserons PostgreSQL.

Pour ajouter une instance de base de données Amazon RDS à votre environnement Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Sous Database (Base de données), choisissez Edit (Modifier).
5. Pour Moteur de base de données, choisissez postgres.
6. Saisissez un nom d'utilisateur principal et un mot de passe. Elastic Beanstalk fournit ces valeurs à votre application en utilisant les propriétés d'environnement.
7. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

La création d'une instance de base de données prend environ 10 minutes. En attendant, vous pouvez mettre à jour votre code source afin de lire les informations de connexion depuis l'environnement. Elastic Beanstalk fournit des détails de connexion en utilisant les variables d'environnement telles que `RDS_HOSTNAME` auxquelles vous pouvez accéder depuis votre application.

La configuration de base de données de CakePHP est dans un fichier nommé `app.php` dans le dossier `config` dans votre code de projet. Ouvrez ce fichier et ajoutez du code qui lit les variables d'environnement de `$_SERVER` et les attribue à des variables locales. Insérez les lignes en surbrillance dans l'exemple ci-dessous, après la première ligne (`<?php`) :

Example~/Eb- .php cake/config/app

```
<?php
if (!defined('RDS_HOSTNAME')) {
  define('RDS_HOSTNAME', $_SERVER['RDS_HOSTNAME']);
  define('RDS_USERNAME', $_SERVER['RDS_USERNAME']);
  define('RDS_PASSWORD', $_SERVER['RDS_PASSWORD']);
  define('RDS_DB_NAME', $_SERVER['RDS_DB_NAME']);
}
return [
...
```

La connexion de base de données est configurée plus bas dans `app.php`. Trouvez la section suivante et modifiez la configuration des sources de données par défaut avec le nom du pilote qui

correspond à votre moteur de base de données (`Mysql`, `Sqlserver` ou `Postgres`) et configurez les variables `host`, `username`, `password` et `database` pour lire les valeurs correspondantes à partir d'Elastic Beanstalk :

Example~/Eb- .php cake/config/app

```
...
     /**
     * Connection information used by the ORM to connect
     * to your application's datastores.
     * Drivers include Mysql Postgres Sqlite Sqlserver
     * See vendor\cakephp\cakephp\src\Database\Driver for complete list
     */
    'Datasources' => [
        'default' => [
            'className' => 'Cake\Database\Connection',
            'driver' => 'Cake\Database\Driver\Postgres',
            'persistent' => false,
            'host' => RDS_HOSTNAME,
            /*
             * CakePHP will use the default DB port based on the driver selected
             * MySQL on MAMP uses port 8889, MAMP users will want to uncomment
             * the following line and set the port accordingly
             */
            //'port' => 'non_standard_port_number',
            'username' => RDS_USERNAME,
            'password' => RDS_PASSWORD,
            'database' => RDS_DB_NAME,
            /*
             * You do not need to set this flag to use full utf-8 encoding (internal
 default since CakePHP 3.6).
             */
            //'encoding' => 'utf8mb4',
            'timezone' => 'UTC',
            'flags' => [],
            'cacheMetadata' => true,
            'log' => false,
...
```

Lorsque l'instance DB a terminé son lancement, regroupez et déployez l'application mise à jour dans votre environnement :

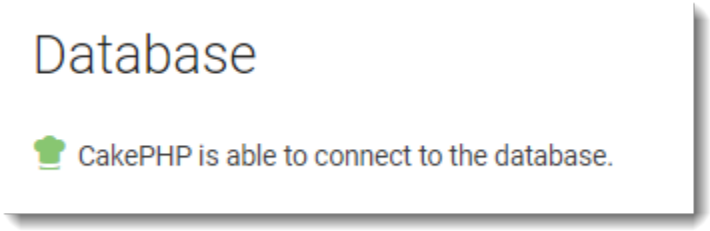
Pour mettre à jour votre environnement Elastic Beanstalk

1. Créez un nouveau groupe source :

```
~/eb-cake$ zip ../cake-v2-rds.zip -r * .[^.]* -x "vendor/*"
```

2. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
3. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
4. Choisissez Upload and Deploy (Charger et déployer).
5. Choisissez Browse (Parcourir) et chargez `cake-v2-rds.zip`.
6. Choisissez Deploy (Déployer).

Le déploiement d'une nouvelle version de votre application prend moins d'une minute. Lorsque le déploiement est terminé, actualisez la page web à nouveau pour vérifier que la connexion de base de données a abouti :



Nettoyage

Une fois que vous avez fini d'utiliser le code de démonstration, vous pouvez mettre fin à votre environnement. Elastic Beanstalk supprime toutes les ressources AWS associées, telles que les instances EC2 Amazon, les instances de base de données, les équilibreurs de charge, lesgroupes de sécurité et les alarmes.

La suppression de ressources n'entraîne pas la suppression de l'application Elastic Beanstalk. Vous pouvez donc créer de nouveaux environnements pour votre application à tout moment.

Pour mettre fin à votre environnement Elastic Beanstalk depuis la console

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Terminate Environment (Résilier l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Vous pouvez également arrêter les ressources de base de données que vous avez créées hors de votre environnement Elastic Beanstalk. Lorsque vous résiliez une instance de base de données Amazon RDS, vous pouvez prendre un instantané et restaurer les données dans une autre instance ultérieurement.

Pour résilier votre instance DB RDS

1. Ouvrez la console Amazon RDS.
2. Choisissez Databases (Bases de données).
3. Sélectionnez votre instance DB.
4. Choisissez Actions, puis Supprimer.
5. Choisissez si vous souhaitez créer un instantané, puis choisissez Delete (Supprimer).

Étapes suivantes

Pour plus d'informations sur CakePHP, lisez l'ouvrage à l'adresse book.cakephp.org.

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le charger manuellement sur la console Elastic Beanstalk. L'interface de ligne de commande Elastic Beanstalk (EB CLI easy-to-use) fournit des commandes pour créer, configurer et déployer des applications dans les environnements Elastic Beanstalk à partir de la ligne de commande.

L'exécution d'une instance de base de données Amazon RDS dans votre environnement Elastic Beanstalk est excellente pour le développement et les tests, mais elle lie le cycle de vie de votre base de données à votre environnement. Pour de plus amples informations sur la connexion à une base de données s'exécutant en dehors de votre environnement, consultez Ajouter une instance de base de données Amazon RDS à votre environnement PHP Elastic Beanstalk.

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez configurer un nom de domaine personnalisé pour votre environnement et activer HTTPS pour des connexions sécurisées.

Déploiement d'une application Symfony sur Elastic Beanstalk

Symfony est une infrastructure open source destinée au développement d'applications web PHP dynamiques. Ce tutoriel explique le processus de génération d'une application Symfony et de son déploiement dans un AWS Elastic Beanstalk environnement.

Sections

- Conditions préalables
- Lancer un environnement Elastic Beanstalk
- Installation de Symfony et génération d'un site web
- Déploiement de votre application
- Configuration des paramètres Composer
- Nettoyage
- Étapes suivantes

Conditions préalables

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans Découvrez comment démarrer avec Elastic Beanstalk pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

Symfony 4.4.9 nécessite PHP 7.1.3 ou version ultérieure. Il requiert également les extensions PHP répertoriées dans la rubrique Exigences techniques de la documentation d'installation officielle de Symfony. Dans le cadre de ce tutoriel, nous utilisons PHP 7.2 et la version correspondante de la plateforme Elastic Beanstalk. Avant de continuer, vous devez installer PHP et Composer.

Pour plus d'informations sur la prise en charge et la maintenance de Symfony, consultez la rubrique Versions de Symfony sur le site Web de Symfony. Pour plus d'informations sur les mises à jour liées à la prise en charge des versions de PHP pour Symfony 4.4.9, consultez la rubrique Notes de mise à jour de Symfony 4.4.9 sur le site Web de Symfony.

Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Choisissez la plateforme PHP et acceptez les paramètres par défaut et l'exemple de code.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk à l'aide de ce lien préconfiguré : console.aws.amazon.com/elasticbeanstalk/home#/newApplication? Nom de l'application = Tutoriels et type d'environnement = LoadBalanced
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Create app (Créer une application).

La création d'un environnement prend environ 5 minutes et crée les ressources suivantes :

- EC2 instance — Une machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications Web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Équilibreur de charge – Équilibreur de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité d'équilibrage de charge : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibreur de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et qui sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *`subdomain`*.*`region`*.elasticbeanstalk.com.

> **Sécurité du domaine**
>
> Pour renforcer la sécurité de vos applications Elastic Beanstalk, le domaine elasticbeanstalk.com est enregistré dans la liste des suffixes publics (PSL).
> Si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Elastic Beanstalk, nous vous recommandons d'utiliser des cookies `__Host-` avec un préfixe pour une sécurité accrue. Cette pratique protège votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

> **Note**
>
> Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon S3.

Installation de Symfony et génération d'un site web

Composer peut installer Symfony et créer un projet de travail avec une commande :

```
~$ composer create-project symfony/website-skeleton eb-symfony
```

Composer installe Symfony et ses dépendances et génère un projet par défaut.

Si vous rencontrez des problèmes lors de l'installation de Symfony, accédez à la rubrique consacrée à l'installation dans la documentation officielle de Symfony.

Déploiement de votre application

Accédez au répertoire du projet.

```
~$ cd eb-symfony
```

Créez un bundle source contenant les fichiers créés par Composer. La commande suivante permet de créer une solution groupée source nommée `symfony-default.zip`. Elle exclut les fichiers du dossier `vendor`, lesquels prennent beaucoup de place et ne sont pas nécessaires pour le déploiement de votre application dans Elastic Beanstalk.

```
eb-symfony$ zip ../symfony-default.zip -r * .[^.]* -x "vendor/*"
```

Chargez le bundle source sur Elastic Beanstalk pour déployer Symfony dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

> **Note**
>
> En vue d'optimiser davantage le bundle source, initialisez un référentiel Git et utilisez la commande `git archive` pour créer le bundle source. Le projet Symfony par défaut inclut un fichier `.gitignore` qui indique à Git d'exclure le dossier `vendor` et d'autres fichiers qui ne sont pas nécessaires pour le déploiement.

Configuration des paramètres Composer

Au terme du déploiement, cliquez sur l'URL pour ouvrir votre application Symfony dans le navigateur.

De quoi s'agit-il ? Par défaut, Elastic Beanstalk constitue la racine de votre projet sur le chemin d'accès racine du site web. Dans ce cas, cependant, la page par défaut (`app.php`) est un niveau en-dessous dans le dossier `web`. Vous pouvez vérifier cela en ajoutant `/public` à l'URL. Par exemple, `http://symfony.us-east-2.elasticbeanstalk.com/public`.

Pour servir l'application Symfony dans le chemin d'accès racine, utilisez la console Elastic Beanstalk pour configurer le document racine du site web.

Pour configurer la racine du document de votre site web

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.

4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Pour Racine du document, entrez **`/public`**.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.
7. Lorsque la mise à jour est terminée, cliquez sur l'URL pour rouvrir votre site dans le navigateur.

Nettoyage

Une fois que vous avez fini d'utiliser le code de démonstration, vous pouvez mettre fin à votre environnement. Elastic Beanstalk supprime toutes les ressources AWS associées, telles que les instances EC2 Amazon, les instances de base de données, les équilibreurs de charge, lesgroupes de sécurité et les alarmes.

La suppression de ressources n'entraîne pas la suppression de l'application Elastic Beanstalk. Vous pouvez donc créer de nouveaux environnements pour votre application à tout moment.

Pour mettre fin à votre environnement Elastic Beanstalk depuis la console

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Terminate Environment (Résilier l’environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Étapes suivantes

Pour plus d'informations sur Symfony, consultez la rubrique Qu'est-ce que Symfony ? sur le site symfony.com.

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le charger manuellement sur la console Elastic Beanstalk. L'interface de ligne de commande Elastic Beanstalk (EB CLI easy-to-use) fournit des commandes pour créer, configurer et déployer des applications dans les environnements Elastic Beanstalk à partir de la ligne de commande.

Dans ce tutoriel, vous avez utilisé la console Elastic Beanstalk pour configurer les options du composeur. Pour que cette configuration fasse partie de votre source d'application, vous pouvez utiliser un fichier de configuration comme celui qui suit.

Example.ebextensions/composer.config

```
option_settings:
  aws:elasticbeanstalk:container:php:phpini:
    document_root: /public
```

Pour de plus amples informations, veuillez consulter Personnalisation d'environnement avancée avec fichiers de configuration (`.ebextensions`).

Symfony utilise ses propres fichiers de configuration pour configurer les connexions de base de données. Pour de plus amples informations sur la connexion à une base de données avec Symfony, veuillez consulter Connexion à une base de données avec Symfony.

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez configurer un nom de domaine personnalisé pour votre environnement et activer HTTPS pour des connexions sécurisées.

Déploiement d'une application PHP haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk

Ce didacticiel explique le processus de lancement d'une instance de base de données RDS externe à AWS Elastic Beanstalk et de configuration d'un environnement de haute disponibilité exécutant une application PHP pour s'y connecter. L'exécution d'une instance de base de données externe à Elastic Beanstalk découple la base de données du cycle de vie de votre environnement. Cela vous permet de connecter la même base de données depuis plusieurs environnements, de remplacer une base de données par une autre ou d'effectuer un déploiement bleu/vert sans affecter votre base de données.

Ce didacticiel utilise un exemple d'application PHP qui utilise une base de données MySQL pour stocker les données de texte fournies par l'utilisateur. L'exemple d'application utilise des fichiers de configuration en vue de configurer les paramètres PHP et de créer une table dans la base de données pour l'application à utiliser. Il montre également comment utiliser un fichier Composer pour installer les packages au cours du déploiement.

Sections

- Conditions préalables
- Lancement d'une instance de base de données dans Amazon RDS
- Créer un environnement Elastic Beanstalk
- Configuration des groupes de sécurité, des propriétés de l'environnement et de la mise à l'échelle
- Déploiement de l'exemple d'application
- Nettoyage
- Étapes suivantes

Conditions préalables

Avant de commencer, téléchargez l'exemple de bundle source de l'application depuis GitHub : eb-demo-php-simple-app-1.3.zip

Les procédures décrites dans ce tutoriel pour les tâches Amazon Relational Database Service (Amazon RDS) supposent que vous lanciez des ressources dans un Amazon Virtual Private Cloud (Amazon VPC) par défaut. Tous les nouveaux comptes incluent un VPC par défaut dans chaque région. Si vous n'avez pas de VPC par défaut, les procédures seront différentes. Consultez Utilisation d'Elastic Beanstalk avec Amazon RDS les instructions relatives aux EC2 plateformes VPC classiques et personnalisées.

Lancement d'une instance de base de données dans Amazon RDS

Pour utiliser une base de données externe avec une application exécutée dans Elastic Beanstalk, lancez d'abord une instance de base de données avec Amazon RDS. Lorsque vous lancez une instance avec Amazon RDS, elle est totalement indépendante d'Elastic Beanstalk et de vos environnements Elastic Beanstalk et ne sera pas résiliée ou surveillée par Elastic Beanstalk.

Utilisez la console Amazon RDS pour lancer une instance de base de données DB MySQL Multi-AZ. Le choix d'un déploiement multi-AZ garantit que votre base de données basculera et qu'elle sera toujours disponible si l'instance de base de données source se retrouve suspendue.

Pour lancer une instance DB RDS dans un VPC par défaut

1. Ouvrez la console RDS.
2. Dans le panneau de navigation, choisissez Databases (Bases de données).
3. Choisissez Create database (Créer une base de données).

4. Choisissez Standard Create (Création standard).

> **Important**
>
> Ne choisissez pas Easy Create (Création facile). Si vous choisissez cette option, vous ne pouvez pas configurer les paramètres nécessaires pour lancer cette base de données RDS.

5. Sous Additional Configuration (Configuration supplémentaire), pour Initial database name (Nom initial de la base de données), tapez **ebdb**.
6. Vérifiez les paramètres par défaut et ajustez ces paramètres en fonction de vos exigences spécifiques. Prêtez attention aux options suivantes :
   - DB instance class (Classe d'instance de base de données) : choisissez une taille d'instance avec un niveau approprié de puissance d'UC et de mémoire pour votre charge de travail.
   - Déploiement multi-AZ : pour une haute disponibilité, définissez ce paramètre sur Créer un Replica/Reader nœud Aurora dans une autre zone AZ.
   - Master username (Identifiant principal) et Master password (Mot de passe principal) : nom d'utilisateur et mot de passe de la base de données. Notez les valeurs de ces paramètres, car vous en aurez besoin par la suite.
7. Vérifiez les paramètres par défaut pour les autres options, puis cliquez sur Create database (Créer une base de données).

Ensuite, modifiez le groupe de sécurité associé à votre instance DB pour autoriser le trafic entrant sur le port approprié. Il s'agit du même groupe de sécurité que celui que vous associerez plus tard à votre environnement Elastic Beanstalk. Par conséquent, la règle que vous ajoutez accordera une autorisation d'entrée aux autres ressources de ce même groupe de sécurité.

Pour modifier les règles de trafic entrant sur le groupe de sécurité associé à votre instance RDS

1. Ouvrez la console Amazon RDS.
2. Choisissez Databases (Bases de données).
3. Choisissez le nom de votre instance de base de données pour en afficher les détails.
4. Dans la section Connectivity (Connectivité), prenez note des Subnets (Sous-réseaux), des Security groups (Groupes de sécurité) et du Endpoint (Point de terminaison) affichés sur cette page. Ainsi, vous pourrez utiliser ces informations ultérieurement.

5. Sous Security (Sécurité), vous voyez le groupe de sécurité associé à l'instance de base de données. Ouvrez le lien pour afficher le groupe de sécurité dans la EC2 console Amazon.
6. Dans les détails du groupe de sécurité, choisissez l'onglet Inbound (Entrant).
7. Choisissez Edit (Modifier).
8. Choisissez Add Rule (Ajouter une règle).
9. Pour Type, choisissez le moteur de base de données utilisé par votre application.
10. Pour Source, entrez **`sg-`** pour afficher la liste des groupes de sécurité disponibles. Choisissez le groupe de sécurité associé au groupe Auto Scaling utilisé avec votre environnement Elastic Beanstalk. Cela permet aux EC2 instances Amazon présentes dans l'environnement d'accéder à la base de données.

11. Choisissez Enregistrer.

La création d'une instance DB prend environ 10 minutes. En attendant, créez votre environnement Elastic Beanstalk.

Créer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Choisissez la plateforme PHP et acceptez les paramètres par défaut et l'exemple de code. Après avoir lancé l'environnement, vous pouvez le configurer pour vous connecter à la base de données, puis déployer l'exemple d'application à partir duquel vous avez effectué le téléchargement GitHub.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk à l'aide de ce lien préconfiguré : console.aws.amazon.com/elasticbeanstalk/home#/newApplication? Nom de l'application = Tutoriels et type d'environnement = LoadBalanced

2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Create app (Créer une application).

La création d'un environnement prend environ 5 minutes et crée les ressources suivantes :

- EC2 instance — Une machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications Web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.
- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Équilibreur de charge – Équilibreur de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité d'équilibrage de charge : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibreur de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et qui sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.

- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

> **Sécurité du domaine**
>
> Pour renforcer la sécurité de vos applications Elastic Beanstalk, le domaine elasticbeanstalk.com est enregistré dans la liste des suffixes publics (PSL).
> Si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Elastic Beanstalk, nous vous recommandons d'utiliser des cookies `__Host-` avec un préfixe pour une sécurité accrue. Cette pratique protège votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient. L'instance DB RDS que vous avez lancée se trouvant en dehors de votre environnement, vous êtes chargé de la gestion de son cycle de vie.

> **Note**
>
> Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour plus d'informations, consultez Utilisation d'Elastic Beanstalk avec Amazon S3.

Configuration des groupes de sécurité, des propriétés de l'environnement et de la mise à l'échelle

Ajoutez le groupe de sécurité de votre instance DB à votre environnement en cours d'exécution. Cette procédure conduit Elastic Beanstalk à remettre en service toutes les instances de votre environnement avec le groupe de sécurité supplémentaire associé.

Pour ajouter un groupe de sécurité à votre environnement

- Effectuez l'une des actions suivantes :
  - Pour ajouter un groupe de sécurité à l'aide de la console Elastic Beanstalk
    a. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
    b. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
    c. Dans le panneau de navigation, choisissez Configuration.
    d. Dans la catégorie de configuration Instances, choisissez Edit (Modifier).
    e. Sous groupes EC2 de sécurité, choisissez le groupe de sécurité à associer aux instances, en plus du groupe de sécurité d'instance créé par Elastic Beanstalk.
    f. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.
    g. Lisez l'avertissement, puis choisissez Confirm (Confirmer).
  - Pour ajouter un groupe de sécurité à l'aide d'un fichier de configuration, utilisez le fichier exemple `securitygroup-addexisting.config`.

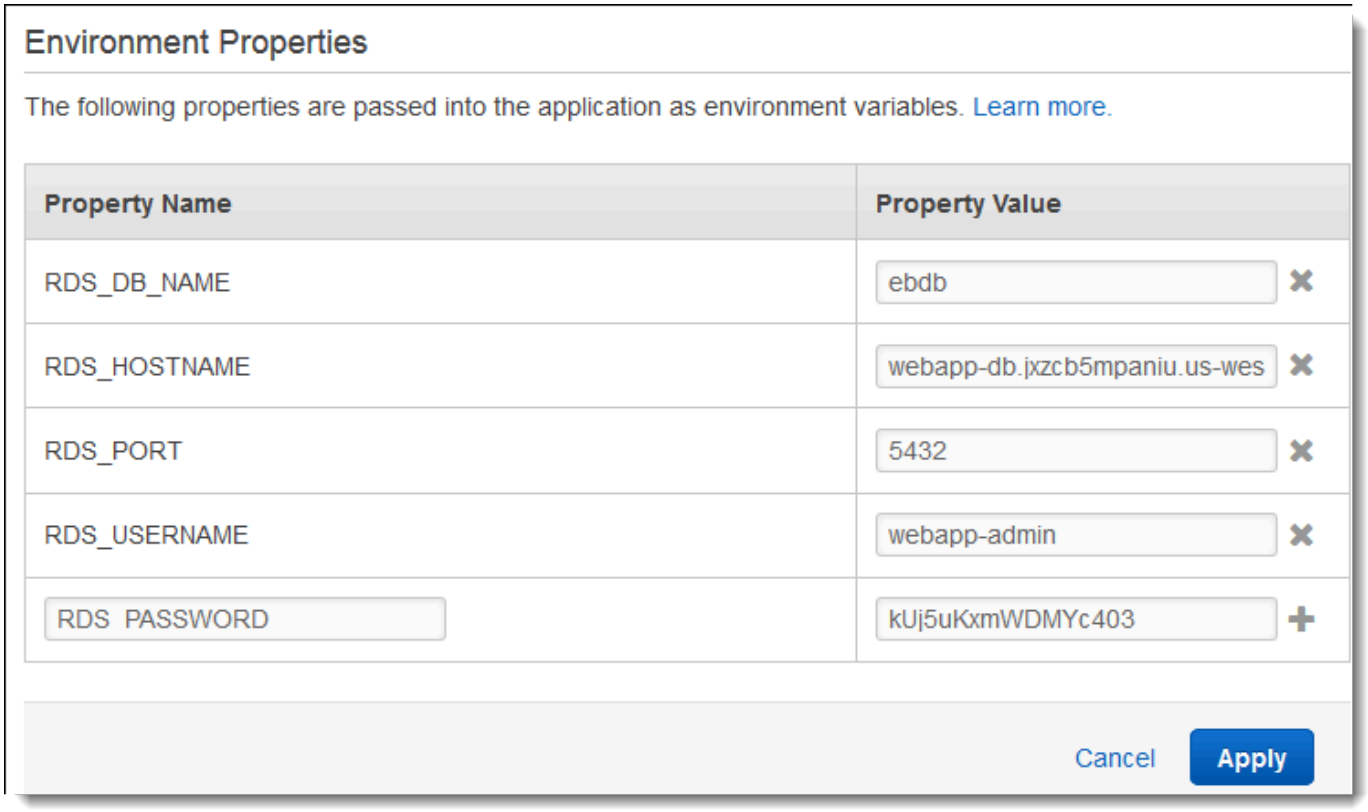
Ensuite, utilisez les propriétés de l'environnement pour transmettre les informations de connexion à votre environnement. L'exemple d'application utilise un ensemble de propriétés par défaut qui correspondent à celles qui sont configurées par Elastic Beanstalk lorsque vous mettez en service une base de données dans votre environnement.

Pour configurer les propriétés d'environnement pour une instance de base de données Amazon RDS

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Dans la section Environment properties (Propriétés de l'environnement), définissez les variables lues par votre application pour créer une chaîne de connexion. Pour assurer la compatibilité avec les environnements disposant d'une instance DB RDS intégrée, utilisez les noms et valeurs

suivants : Vous pouvez trouver toutes les valeurs, à l'exception de votre mot de passe, dans la console RDS.

Nom de la propriété	Description	Valeur de la propriété
RDS_HOSTNAME	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, **ebdb**.	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.



6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Enfin, configurez le groupe Auto Scaling de votre environnement avec un nombre minimum d'instances plus élevé. Exécutez au moins deux instances en permanence afin d'empêcher que les serveurs web de votre environnement constituent un point de défaillance unique et pour vous permettre de déployer des modifications sans mettre votre site hors service.

Pour configurer le groupe Auto Scaling de votre environnement pour une haute disponibilité

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Auto Scaling group (Groupe Auto Scaling), définissez les Min instances (Instances min.) sur **2**.

6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Déploiement de l'exemple d'application

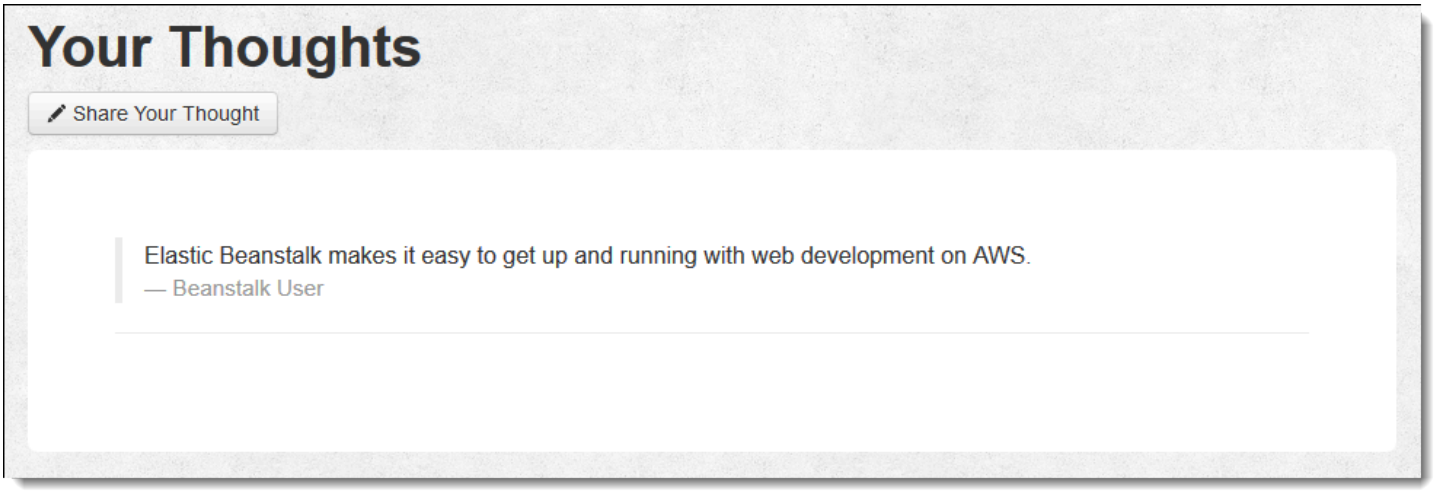
Votre environnement est maintenant prêt à exécuter l'exemple d'application et à se connecter à Amazon RDS. Déployez l'exemple d'application dans votre environnement.

> **Note**
>
> Téléchargez le bundle source depuis GitHub, si ce n'est pas déjà fait : eb-demo-php-simple-app-1.3.zip

Pour déployer un groupe source

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

Le site collecte les commentaires des utilisateurs et tire parti d'une base de données MySQL pour stocker les données. Pour ajouter un commentaire, cliquez sur Share Your Thought, saisissez un commentaire, puis sélectionnez Submit Your Thought. L'application web écrit le commentaire dans la base de données pour que n'importe quelle instance de l'environnement puisse le lire et qu'il ne soit pas perdu si les instances se retrouvent suspendues.



Nettoyage

Une fois que vous avez fini d'utiliser le code de démonstration, vous pouvez mettre fin à votre environnement. Elastic Beanstalk supprime toutes les ressources AWS associées, telles que les instances EC2 Amazon, les instances de base de données, les équilibreurs de charge, lesgroupes de sécurité et les alarmes.

La suppression de ressources n'entraîne pas la suppression de l'application Elastic Beanstalk. Vous pouvez donc créer de nouveaux environnements pour votre application à tout moment.

Pour mettre fin à votre environnement Elastic Beanstalk depuis la console

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Terminate Environment (Résilier l’environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Vous pouvez également arrêter les ressources de base de données que vous avez créées hors de votre environnement Elastic Beanstalk. Lorsque vous résiliez une instance de base de données Amazon RDS, vous pouvez prendre un instantané et restaurer les données dans une autre instance ultérieurement.

Pour résilier votre instance DB RDS

1. Ouvrez la console Amazon RDS.
2. Choisissez Databases (Bases de données).
3. Sélectionnez votre instance DB.
4. Choisissez Actions, puis Supprimer.
5. Choisissez si vous souhaitez créer un instantané, puis choisissez Delete (Supprimer).

Étapes suivantes

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le charger manuellement sur la console Elastic Beanstalk. L'interface de ligne de commande Elastic Beanstalk (EB CLI easy-to-use) fournit des commandes pour créer, configurer et déployer des applications dans les environnements Elastic Beanstalk à partir de la ligne de commande.

L'exemple d'application utilise des fichiers de configuration pour configurer les paramètres PHP et pour créer une table dans la base de données si elle n'existe pas déjà. Vous pouvez également utiliser un fichier de configuration pour configurer les paramètres de groupe de sécurité de vos instances lors de la création de l'environnement afin d'éviter les mises à jour de configuration, qui prennent du temps. Pour plus d'informations, consultez Personnalisation d'environnement avancée avec fichiers de configuration (`.ebextensions`).

Pour le développement et les tests, vous pouvez utiliser la fonctionnalité Elastic Beanstalk permettant d'ajouter directement une instance de base de données gérée à votre environnement. Pour savoir comment configurer une base de données dans votre environnement, consultez Ajout d'une base de données à votre environnement Elastic Beanstalk.

Si vous avez besoin d'une base de données hautes performances, envisagez d'utiliser Amazon Aurora. Amazon Aurora est un moteur de base de données compatible MySQL qui offre des fonctionnalités de base de données commerciales à faible coût. Pour connecter votre application à une autre base de données, répétez la procédure de configuration du groupe de sécurité et mettez à jour les propriétés d'environnement associées à RDS.

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez configurer un nom de domaine personnalisé pour votre environnement et activer HTTPS pour des connexions sécurisées.

Déploiement d'un WordPress site Web à haute disponibilité avec une base de données Amazon RDS externe sur Elastic Beanstalk

Ce didacticiel explique comment lancer une instance de base de données Amazon RDS externe à AWS Elastic Beanstalk, puis comment configurer un environnement de haute disponibilité exécutant un WordPress site Web pour s'y connecter. Le site web utilise Amazon Elastic File System (Amazon EFS) comme stockage partagé pour les fichiers téléchargés.

L'exécution d'une instance de base de données externe à Elastic Beanstalk découple la base de données du cycle de vie de votre environnement. Cela vous permet de connecter la même base de données depuis plusieurs environnements, de remplacer une base de données par une autre ou d'effectuer un déploiement bleu/vert sans affecter votre base de données.

> **Note**
>
> Pour les informations actuelles sur la compatibilité des versions de PHP avec WordPress les versions, consultez la section Compatibilité et WordPress versions de PHP sur le WordPress site Web. Vous devez vous référer à ces informations avant de passer à une nouvelle version de PHP pour vos WordPress implémentations.

Rubriques

- Conditions préalables
- Lancement d'une instance de base de données dans Amazon RDS
- Télécharger WordPress
- Lancer un environnement Elastic Beanstalk
- Configuration des groupes de sécurité et des propriétés de l'environnement
- Configuration et déploiement de votre application
- Installer WordPress
- Mise à jour des clés et des valeurs salt
- Retrait des restrictions d'accès
- Configuration de votre groupe Auto Scaling
- Mise à niveau WordPress
- Nettoyage
- Étapes suivantes

Conditions préalables

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans Découvrez comment démarrer avec Elastic Beanstalk pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

VPC par défaut

Les procédures Amazon Relational Database Service (Amazon RDS) décrites dans ce tutoriel supposent que vous lancez des ressources dans un Amazon Virtual Private Cloud (Amazon VPC) par défaut. Tous les nouveaux comptes incluent un VPC par défaut dans chaque AWS région. Si vous n'avez pas de VPC par défaut, les procédures seront différentes. Consultez Utilisation d'Elastic Beanstalk avec Amazon RDS les instructions relatives aux EC2 plateformes VPC classiques et personnalisées.

AWS Régions

L'exemple d'application utilise Amazon EFS, qui ne fonctionne que dans AWS les régions compatibles avec Amazon EFS. Pour en savoir plus sur AWS les régions prises en charge, consultez Amazon Elastic File System Endpoints and Quotas dans le Références générales AWS.

Lancement d'une instance de base de données dans Amazon RDS

Lorsque vous lancez une instance avec Amazon RDS, elle est complètement indépendante d'Elastic Beanstalk et de vos environnements Elastic Beanstalk et ne sera pas résiliée ou surveillée par Elastic Beanstalk.

Dans les étapes suivantes, vous allez utiliser la console Amazon RDS pour :

- Lancez une base de données avec le moteur MySQL .
- Activez un déploiement Multi-AZ. Cela crée une veille dans une autre zone de disponibilité (AZ) afin d'assurer la redondance des données, d'éliminer les I/O blocages et de minimiser les pics de latence lors des sauvegardes du système.

Pour lancer une instance DB RDS dans un VPC par défaut

1. Ouvrez la console RDS.
2. Dans le panneau de navigation, choisissez Databases (Bases de données).
3. Choisissez Create database (Créer une base de données).
4. Choisissez Standard Create (Création standard).

   > **Important**
   >
   > Ne choisissez pas Easy Create (Création facile). Si vous choisissez cette option, vous ne pouvez pas configurer les paramètres nécessaires pour lancer cette base de données RDS.

5. Sous Additional Configuration (Configuration supplémentaire), pour Initial database name (Nom initial de la base de données), tapez **ebdb**.
6. Vérifiez les paramètres par défaut et ajustez ces paramètres en fonction de vos exigences spécifiques. Prêtez attention aux options suivantes :

   - DB instance class (Classe d'instance de base de données) : choisissez une taille d'instance avec un niveau approprié de puissance d'UC et de mémoire pour votre charge de travail.
   - Déploiement multi-AZ : pour une haute disponibilité, définissez ce paramètre sur Créer un Replica/Reader nœud Aurora dans une autre zone AZ.
   - Master username (Identifiant principal) et Master password (Mot de passe principal) : nom d'utilisateur et mot de passe de la base de données. Notez les valeurs de ces paramètres, car vous en aurez besoin par la suite.

7. Vérifiez les paramètres par défaut pour les autres options, puis cliquez sur Create database (Créer une base de données).

Une fois votre instance de base de données créée, modifiez le groupe de sécurité qui lui est associé afin d'autoriser le trafic entrant sur le port approprié.

> **Note**
>
> Il s'agit du même groupe de sécurité que celui que vous associerez plus tard à votre environnement Elastic Beanstalk. Par conséquent, la règle que vous ajoutez maintenant accordera une autorisation d'entrée aux autres ressources de ce même groupe de sécurité.

Pour modifier les règles de trafic entrant sur le groupe de sécurité associé à votre instance RDS

1. Ouvrez la console Amazon RDS.
2. Choisissez Databases (Bases de données).
3. Choisissez le nom de votre instance de base de données pour en afficher les détails.
4. Dans la section Connectivity (Connectivité), prenez note des Subnets (Sous-réseaux), des Security groups (Groupes de sécurité) et du Endpoint (Point de terminaison) affichés sur cette page. Ainsi, vous pourrez utiliser ces informations ultérieurement.
5. Sous Security (Sécurité), vous voyez le groupe de sécurité associé à l'instance de base de données. Ouvrez le lien pour afficher le groupe de sécurité dans la EC2 console Amazon.
6. Dans les détails du groupe de sécurité, choisissez l'onglet Inbound (Entrant).
7. Choisissez Edit (Modifier).
8. Choisissez Add Rule (Ajouter une règle).
9. Pour Type, choisissez le moteur de base de données utilisé par votre application.
10. Pour Source, entrez **`sg-`** pour afficher la liste des groupes de sécurité disponibles. Choisissez le groupe de sécurité associé au groupe Auto Scaling utilisé avec votre environnement Elastic Beanstalk. Cela permet aux EC2 instances Amazon présentes dans l'environnement d'accéder à la base de données.

11. Choisissez Enregistrer.

La création d'une instance DB prend environ 10 minutes. En attendant, téléchargez WordPress et créez votre environnement Elastic Beanstalk.

Télécharger WordPress

Pour préparer le déploiement WordPress en utilisant AWS Elastic Beanstalk, vous devez copier les WordPress fichiers sur votre ordinateur et fournir les informations de configuration correctes.

Pour créer un WordPress projet

1. Téléchargez WordPress sur wordpress.org.

```
~$curl https://wordpress.org/wordpress-6.2.tar.gz -o wordpress.tar.gz
```

2. Téléchargez les fichiers de configuration à partir de l'exemple de référentiel.

```
~$ wget https://github.com/aws-samples/eb-php-wordpress/releases/download/v1.1/eb-
php-wordpress-v1.zip
```

3. Extrayez WordPress et modifiez le nom du dossier.

```
~$ tar -xvf wordpress.tar.gz
~$ mv wordpress wordpress-beanstalk
~$ cd wordpress-beanstalk
```

4. Extrayez les fichiers de configuration au cours de l' WordPress installation.

```
~/wordpress-beanstalk$ unzip ../eb-php-wordpress-v1.zip
 creating: .ebextensions/
inflating: .ebextensions/dev.config
inflating: .ebextensions/efs-create.config
inflating: .ebextensions/efs-mount.config
inflating: .ebextensions/loadbalancer-sg.config
inflating: .ebextensions/wordpress.config
inflating: LICENSE
inflating: README.md
inflating: wp-config.php
```

Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Après avoir lancé l'environnement, vous pouvez le configurer pour qu'il se connecte à la base de données, puis y déployer le WordPress code.

Dans les étapes suivantes, vous allez utiliser la console Elastic Beanstalk pour :

- Créez une application Elastic Beanstalk à l'aide de la plateforme PHP gérée.
- Acceptez les paramètres par défaut et l'exemple de code.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk à l'aide de ce lien préconfiguré : console.aws.amazon.com/elasticbeanstalk/home#/newApplication? Nom de l'application = Tutoriels et type d'environnement = LoadBalanced
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Create app (Créer une application).

La création d'un environnement; prend environ 5 minutes et crée les ressources suivantes.

Elastic Beanstalk a créé des ressources

- EC2 instance — Une machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications Web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.
- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge

d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.

- Équilibreur de charge – Équilibreur de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité d'équilibrage de charge : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibreur de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et qui sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack : Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui mène à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

> **Sécurité du domaine**
>
> Pour renforcer la sécurité de vos applications Elastic Beanstalk, le domaine elasticbeanstalk.com est enregistré dans la liste des suffixes publics (PSL).
> Si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Elastic Beanstalk, nous vous recommandons d'utiliser des cookies `__Host-` avec un préfixe pour une sécurité accrue. Cette pratique protège votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

L'instance Amazon RDS que vous avez lancée se trouvant en dehors de votre environnement, vous êtes chargé de la gestion de son cycle de vie.

> **Note**
>
> Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour plus d'informations, consultez Utilisation d'Elastic Beanstalk avec Amazon S3.

Configuration des groupes de sécurité et des propriétés de l'environnement

Ajoutez le groupe de sécurité de votre instance DB à votre environnement en cours d'exécution. Cette procédure conduit Elastic Beanstalk à remettre en service toutes les instances de votre environnement avec le groupe de sécurité supplémentaire associé.

Pour ajouter un groupe de sécurité à votre environnement

- Effectuez l'une des actions suivantes :
  - Pour ajouter un groupe de sécurité à l'aide de la console Elastic Beanstalk
    a. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
    b. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
    c. Dans le panneau de navigation, choisissez Configuration.
    d. Dans la catégorie de configuration Instances, choisissez Edit (Modifier).
    e. Sous groupes EC2 de sécurité, choisissez le groupe de sécurité à associer aux instances, en plus du groupe de sécurité d'instance créé par Elastic Beanstalk.
    f. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.
    g. Lisez l'avertissement, puis choisissez Confirm (Confirmer).
  - Pour ajouter un groupe de sécurité à l'aide d'un fichier de configuration, utilisez le fichier exemple `securitygroup-addexisting.config`.

Ensuite, utilisez les propriétés de l'environnement pour transmettre les informations de connexion à votre environnement.

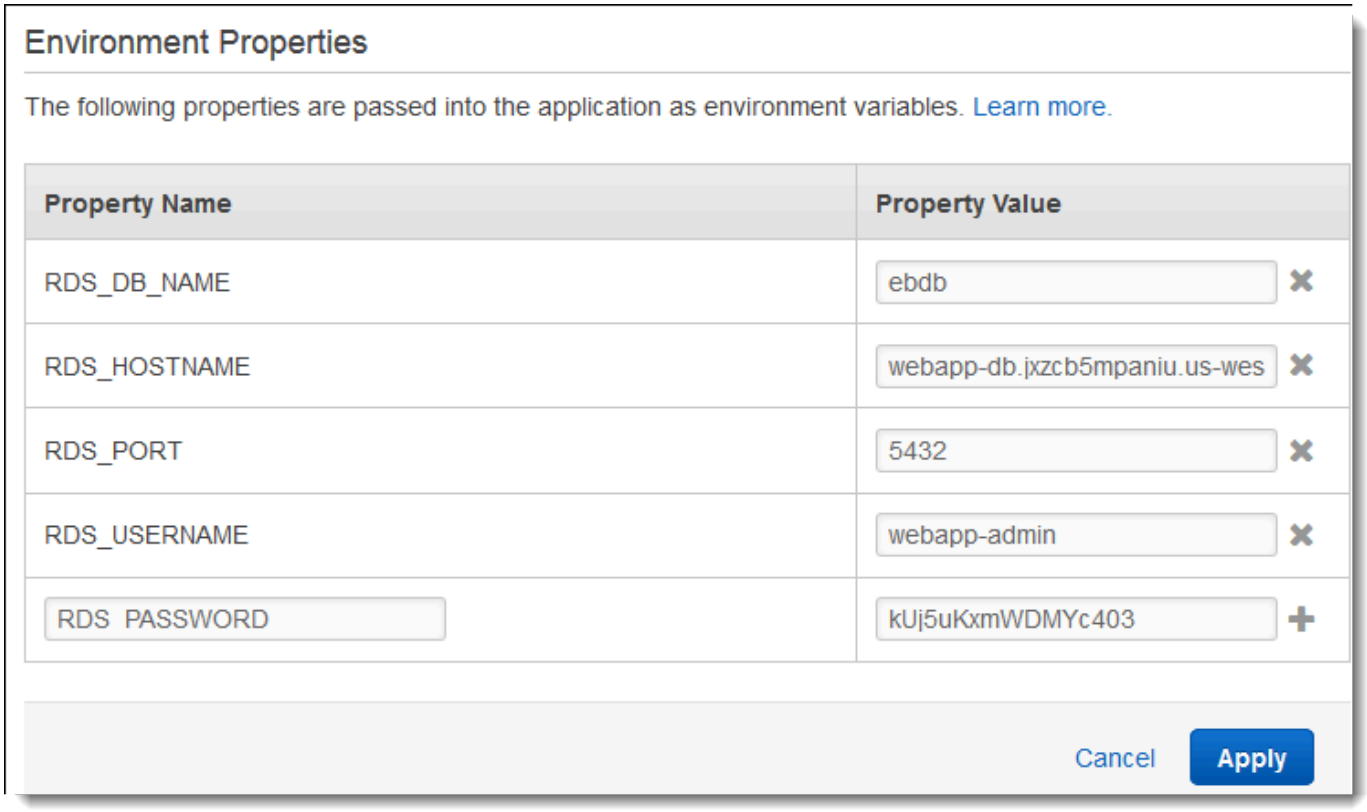
L' WordPress application utilise un ensemble de propriétés par défaut qui correspondent à celles configurées par Elastic Beanstalk lorsque vous approvisionnez une base de données au sein de votre environnement.

Pour configurer les propriétés d'environnement pour une instance de base de données Amazon RDS

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Dans la section Environment properties (Propriétés de l'environnement), définissez les variables lues par votre application pour créer une chaîne de connexion. Pour assurer la compatibilité avec les environnements disposant d'une instance DB RDS intégrée, utilisez les noms et valeurs suivants : Vous pouvez trouver toutes les valeurs, à l'exception de votre mot de passe, dans la console RDS.

Nom de la propriété	Description	Valeur de la propriété
`RDS_HOSTNAME`	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
`RDS_PORT`	Port sur lequel l'instanc e de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
`RDS_DB_NAME`	Nom de la base de données, **ebdb**.	Sous l'onglet Configuration de la console Amazon RDS :

Nom de la propriété	Description	Valeur de la propriété
		DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.



6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Configuration et déploiement de votre application

Vérifiez que la structure de votre dossier `wordpress-beanstalk` est correcte, comme illustré.

```
wordpress-beanstalk$ tree -aL 1
.
### .ebextensions
### index.php
### LICENSE
### license.txt
### readme.html
### README.md
### wp-activate.php
### wp-admin
### wp-blog-header.php
### wp-comments-post.php
### wp-config.php
### wp-config-sample.php
### wp-content
### wp-cron.php
### wp-includes
### wp-links-opml.php
### wp-load.php
### wp-login.php
### wp-mail.php
### wp-settings.php
### wp-signup.php
### wp-trackback.php
### xmlrpc.php
```

Le fichier `wp-config.php` personnalisé du référentiel de projet utilise les variables d'environnement que vous avez définies à l'étape précédente pour configurer la connexion de base de données. Le dossier `.ebextensions` contient des fichiers de configuration qui créent des ressources supplémentaires au sein de votre environnement Elastic Beanstalk.

Les fichiers de configuration doivent être modifiés pour être compatibles avec votre compte. Remplacez les valeurs d'espace réservé dans les fichiers par les valeurs appropriées IDs et créez un ensemble de sources.

Pour mettre à jour les fichiers de configuration et créer une solution groupée source

1. Modifiez les fichiers de configuration comme suit.

   - `.ebextensions/dev.config`— Restreint l'accès à votre environnement pour le protéger pendant le processus WordPress d'installation. Remplacez l'adresse IP fictive située en haut

du fichier par l'adresse IP publique de l'ordinateur que vous utiliserez pour accéder au site Web de votre environnement afin de terminer votre WordPress installation.

> **Note**
>
> En fonction de votre réseau, il se peut que vous ayez besoin d'utiliser un bloc d'adresses IP.

- `.ebextensions/efs-create.config`— Crée un système de fichiers EFS et des points de montage dans chaque disponibilité Zone/subnet de votre VPC. Identifiez votre VPC et votre sous-réseau par défaut dans IDs la console Amazon VPC.

2. Créez un bundle source contenant les fichiers dans votre dossier de projet. La commande suivante permet de créer une solution groupée source nommée `wordpress-beanstalk.zip`.

```
~/eb-wordpress$ zip ../wordpress-beanstalk.zip -r * .[^.]*
```

Téléchargez le bundle source dans Elastic Beanstalk WordPress pour le déployer dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

Installer WordPress

Pour terminer votre WordPress installation

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez l'URL d'environnement pour ouvrir votre site dans un navigateur. Vous êtes redirigé vers un assistant WordPress d'installation car vous n'avez pas encore configuré le site.
4. Effectuez une installation standard. Le fichier `wp-config.php` est déjà présent dans le code source et configuré pour lire les informations de connexion à la base de données à partir de l'environnement. Vous ne devriez donc pas être invité à configurer la connexion.

L'installation prend environ une minute.

Mise à jour des clés et des valeurs salt

Le fichier WordPress de configuration lit `wp-config.php` également les valeurs des clés et des sels à partir des propriétés de l'environnement. Actuellement, ces propriétés sont toutes définies sur `test` par le fichier `wordpress.config` dans le dossier `.ebextensions`.

La valeur salt de hachage peut être n'importe quelle valeur qui respecte les exigences de propriété d'environnement, mais vous ne devez pas la stocker dans le contrôle de code source. Utilisez la console Elastic Beanstalk pour définir ces propriétés directement sur l'environnement.

Pour mettre à jour des propriétés d'environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Sous Software (Logiciel), choisissez Edit (Modifier).
5. Pour `Environment properties`, modifiez les propriétés suivantes :
   - AUTH_KEY – Valeur choisie pour AUTH_KEY.

- SECURE_AUTH_KEY – Valeur choisie pour SECURE_AUTH_KEY.
- LOGGED_IN_KEY – Valeur choisie pour LOGGED_IN_KEY.
- NONCE_KEY – Valeur choisie pour NONCE_KEY.
- AUTH_SALT – Valeur choisie pour AUTH_SALT.
- SECURE_AUTH_SALT – Valeur choisie pour SECURE_AUTH_SALT.
- LOGGED_IN_SALT – Valeur choisie pour LOGGED_IN_SALT.
- NONCE_SALT – Valeur choisie pour NONCE_SALT.

6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

> **Note**
>
> La définition des propriétés sur l'environnement remplace directement les valeurs dans `wordpress.config`.

Retrait des restrictions d'accès

L'exemple de projet inclut le fichier de configuration `loadbalancer-sg.config`. Il crée un groupe de sécurité et l'affecte à l'équilibreur de charge de l'environnement, à l'aide de l'adresse IP que vous avez configurée dans `dev.config`. Il restreint l'accès HTTP sur le port 80 aux connexions provenant de votre réseau. Dans le cas contraire, un tiers pourrait se connecter à votre site avant que vous n'ayez installé WordPress et configuré votre compte administrateur.

Maintenant que vous l'avez installé WordPress, supprimez le fichier de configuration pour ouvrir le site au monde entier.

Pour supprimer la restriction et mettre à jour votre environnement

1. Supprimez le fichier `.ebextensions/loadbalancer-sg.config` de votre répertoire de projet.

```
~/wordpress-beanstalk$ rm .ebextensions/loadbalancer-sg.config
```

2. Créez une solution groupée source.

```
~/eb-wordpress$ zip ../wordpress-beanstalk-v2.zip -r * .[^.]*
```

Téléchargez le bundle source dans Elastic Beanstalk WordPress pour le déployer dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

Configuration de votre groupe Auto Scaling

Enfin, configurez le groupe Auto Scaling de votre environnement avec un nombre minimum d'instances plus élevé. Exécutez au moins deux instances à tout moment pour éviter que les serveurs Web de votre environnement constituent un point de défaillance unique. Cela vous permet également de déployer des modifications sans mettre votre site hors service.

Pour configurer le groupe Auto Scaling de votre environnement pour une haute disponibilité

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Auto Scaling group (Groupe Auto Scaling), définissez les Min instances (Instances min.) sur **2**.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Pour la prise en charge des chargements de contenu sur plusieurs instances, l'exemple de projet utilise Amazon EFS pour créer un système de fichiers partagé. Créez une publication sur le site et chargez le contenu afin de le stocker sur le système de fichiers partagé. Affichez la publication et actualisez la page plusieurs fois pour atteindre les deux instances et vérifier que le système de fichiers partagé fonctionne.

Mise à niveau WordPress

Pour effectuer une mise à niveau vers une nouvelle version de WordPress, sauvegardez votre site et déployez-le dans un nouvel environnement.

> **Important**
>
> N'utilisez pas la fonctionnalité de mise à jour intégrée WordPress ou ne mettez pas à jour vos fichiers source pour utiliser une nouvelle version. Ces deux actions peuvent faire en sorte que votre publication URLs renvoie des erreurs 404 même si elles se trouvent toujours dans la base de données et le système de fichiers.

Pour mettre à niveau WordPress

1. Dans la console WordPress d'administration, utilisez l'outil d'exportation pour exporter vos publications vers un fichier XML.
2. Déployez et installez la nouvelle version WordPress de sur Elastic Beanstalk en suivant les mêmes étapes que pour installer la version précédente. Pour éviter les temps d'arrêt, vous pouvez créer un environnement avec la nouvelle version.
3. Dans la nouvelle version, installez l'outil d' WordPress importation dans la console d'administration et utilisez-le pour importer le fichier XML contenant vos publications. Si les publications ont été créées par l'utilisateur administrateur dans l'ancienne version, affectez-les à l'utilisateur administrateur sur le nouveau site au lieu d'essayer d'importer l'utilisateur admin.
4. Si vous avez déployé la nouvelle version dans un autre environnement, effectuez un échange CNAME pour rediriger les utilisateurs de l'ancien site vers le nouveau site.

Nettoyage

Une fois que vous avez fini d'utiliser le code de démonstration, vous pouvez mettre fin à votre environnement. Elastic Beanstalk supprime toutes les ressources AWS associées, telles que les

instances EC2 Amazon, les instances de base de données, les équilibreurs de charge, lesgroupes de sécurité et les alarmes.

La suppression de ressources ne supprime pas l'application Elastic Beanstalk. Vous pouvez donc créer de nouveaux environnements pour votre application à tout moment.

Pour mettre fin à votre environnement Elastic Beanstalk depuis la console

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Terminate Environment (Résilier l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Vous pouvez également arrêter les ressources de base de données que vous avez créées hors de votre environnement Elastic Beanstalk. Lorsque vous résiliez une instance de base de données Amazon RDS, vous pouvez prendre un instantané et restaurer les données dans une autre instance ultérieurement.

Pour résilier votre instance DB RDS

1. Ouvrez la console Amazon RDS.
2. Choisissez Databases (Bases de données).
3. Sélectionnez votre instance DB.
4. Choisissez Actions, puis Supprimer.
5. Choisissez si vous souhaitez créer un instantané, puis choisissez Delete (Supprimer).

Étapes suivantes

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le charger manuellement sur la console Elastic Beanstalk. L'interface de ligne de commande Elastic Beanstalk (EB CLI easy-to-use) fournit des commandes pour créer, configurer et déployer des applications dans les environnements Elastic Beanstalk à partir de la ligne de commande.

L'exemple d'application utilise des fichiers de configuration pour configurer les paramètres PHP et pour créer une table dans la base de données, si elle n'existe pas déjà. Vous pouvez également utiliser un fichier de configuration pour configurer les paramètres de groupe de sécurité de vos instances lors de la création de l'environnement afin d'éviter les mises à jour de configuration, qui prennent du temps. Pour plus d'informations, consultez Personnalisation d'environnement avancée avec fichiers de configuration (`.ebextensions`).

Pour le développement et les tests, vous pouvez utiliser la fonctionnalité Elastic Beanstalk permettant d'ajouter directement une instance de base de données gérée à votre environnement. Pour savoir comment configurer une base de données dans votre environnement, consultez Ajout d'une base de données à votre environnement Elastic Beanstalk.

Si vous avez besoin d'une base de données hautes performances, envisagez d'utiliser Amazon Aurora. Amazon Aurora est un moteur de base de données compatible MySQL qui offre des fonctionnalités de base de données commerciales à faible coût. Pour connecter votre application à une autre base de données, répétez la procédure de configuration du groupe de sécurité et mettez à jour les propriétés d'environnement associées à RDS.

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez configurer un nom de domaine personnalisé pour votre environnement et activer HTTPS pour des connexions sécurisées.

Déploiement d'un site web Drupal haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk

Ce didacticiel explique le processus de lancement d'une instance de base de données RDS externe à AWS Elastic Beanstalk. Il décrit ensuite comment configurer un environnement à haute disponibilité en exécutant un site web Drupal pour s'y connecter. Le site web utilise Amazon Elastic File System (Amazon EFS) comme stockage partagé pour les fichiers téléchargés. L'exécution d'une instance de base de données externe à Elastic Beanstalk dissocie la base de données du cycle de vie de votre environnement et vous permet de vous connecter à la même base de données depuis plusieurs environnements, de remplacer une base de données par une autre ou blue/green d'effectuer un déploiement sans affecter votre base de données.

Sections

- Conditions préalables
- Lancement d'une instance de base de données dans Amazon RDS

- Lancer un environnement Elastic Beanstalk
- Configuration des paramètres de sécurité et des propriétés de l'environnement
- Configuration et déploiement de votre application
- Installation de Drupal
- Mise à jour de la configuration de Drupal et retrait de restrictions d'accès
- Configuration de votre groupe Auto Scaling
- Nettoyage
- Étapes suivantes

Conditions préalables

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans Découvrez comment démarrer avec Elastic Beanstalk pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

Les procédures décrites dans ce tutoriel pour les tâches Amazon Relational Database Service (Amazon RDS) supposent que vous lanciez des ressources dans un Amazon Virtual Private Cloud (Amazon VPC) par défaut. Tous les nouveaux comptes incluent un VPC par défaut dans chaque région. Si vous n'avez pas de VPC par défaut, les procédures seront différentes. Consultez Utilisation d'Elastic Beanstalk avec Amazon RDS les instructions relatives aux EC2 plateformes VPC classiques et personnalisées.

L'exemple d'application utilise Amazon EFS. Cela ne fonctionne que dans AWS les régions qui prennent en charge Amazon EFS. Pour en savoir plus sur la prise en charge AWS des régions, consultez Amazon Elastic File System Endpoints and Quotas dans le Références générales AWS.

Si la plateforme de votre environnement Elastic Beanstalk utilise PHP 7.4 ou version antérieure, nous vous recommandons d'utiliser la version 8.9.13 de Drupal pour ce tutoriel. Pour les plateformes installées avec PHP 8.0 ou version ultérieure, nous vous recommandons d'utiliser Drupal 9.1.5.

Pour plus d'informations sur les versions de Drupal et les versions de PHP qu'elles prennent en charge, consultez Exigences relatives à PHP sur le site Web de Drupal. Les versions principales recommandées par Drupal sont répertoriées sur le site Web https://www.drupal. org/project/drupal.

Lancement d'une instance de base de données dans Amazon RDS

Pour utiliser une base de données externe avec une application exécutée dans Elastic Beanstalk, lancez d'abord une instance de base de données avec Amazon RDS. Lorsque vous lancez une instance avec Amazon RDS, elle est totalement indépendante d'Elastic Beanstalk et de vos environnements Elastic Beanstalk et ne sera pas résiliée ou surveillée par Elastic Beanstalk.

Utilisez la console Amazon RDS pour lancer une instance de base de données DB MySQL Multi-AZ. Le choix d'un déploiement multi-AZ garantit que votre base de données basculera et qu'elle sera toujours disponible si l'instance de base de données source se retrouve hors service.

Pour lancer une instance DB RDS dans un VPC par défaut

1. Ouvrez la console RDS.
2. Dans le panneau de navigation, choisissez Databases (Bases de données).
3. Choisissez Create database (Créer une base de données).
4. Choisissez Standard Create (Création standard).

   > **Important**
   >
   > Ne choisissez pas Easy Create (Création facile). Si vous choisissez cette option, vous ne pouvez pas configurer les paramètres nécessaires pour lancer cette base de données RDS.

5. Sous Additional Configuration (Configuration supplémentaire), pour Initial database name (Nom initial de la base de données), tapez **ebdb**.
6. Vérifiez les paramètres par défaut et ajustez ces paramètres en fonction de vos exigences spécifiques. Prêtez attention aux options suivantes :

   - DB instance class (Classe d'instance de base de données) : choisissez une taille d'instance avec un niveau approprié de puissance d'UC et de mémoire pour votre charge de travail.

- Déploiement multi-AZ : pour une haute disponibilité, définissez ce paramètre sur Créer un Replica/Reader nœud Aurora dans une autre zone AZ.
- Master username (Identifiant principal) et Master password (Mot de passe principal) : nom d'utilisateur et mot de passe de la base de données. Notez les valeurs de ces paramètres, car vous en aurez besoin par la suite.

7. Vérifiez les paramètres par défaut pour les autres options, puis cliquez sur Create database (Créer une base de données).

Ensuite, modifiez le groupe de sécurité associé à votre instance DB pour autoriser le trafic entrant sur le port approprié. Il s'agit du même groupe de sécurité que celui que vous associerez plus tard à votre environnement Elastic Beanstalk. Par conséquent, la règle que vous ajoutez accordera une autorisation d'entrée aux autres ressources de ce même groupe de sécurité.

Pour modifier les règles de trafic entrant sur le groupe de sécurité associé à votre instance RDS

1. Ouvrez la console Amazon RDS.
2. Choisissez Databases (Bases de données).
3. Choisissez le nom de votre instance de base de données pour en afficher les détails.
4. Dans la section Connectivity (Connectivité), prenez note des Subnets (Sous-réseaux), des Security groups (Groupes de sécurité) et du Endpoint (Point de terminaison) affichés sur cette page. Ainsi, vous pourrez utiliser ces informations ultérieurement.
5. Sous Security (Sécurité), vous voyez le groupe de sécurité associé à l'instance de base de données. Ouvrez le lien pour afficher le groupe de sécurité dans la EC2 console Amazon.
6. Dans les détails du groupe de sécurité, choisissez l'onglet Inbound (Entrant).
7. Choisissez Edit (Modifier).
8. Choisissez Add Rule (Ajouter une règle).
9. Pour Type, choisissez le moteur de base de données utilisé par votre application.
10. Pour Source, entrez **`sg-`** pour afficher la liste des groupes de sécurité disponibles. Choisissez le groupe de sécurité associé au groupe Auto Scaling utilisé avec votre environnement Elastic Beanstalk. Cela permet aux EC2 instances Amazon présentes dans l'environnement d'accéder à la base de données.

11. Choisissez Enregistrer.

La création d'une instance DB prend environ 10 minutes. En attendant, lancez votre environnement Elastic Beanstalk.

Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Choisissez la plateforme PHP et acceptez les paramètres par défaut et l'exemple de code. Une fois l'environnement lancé, vous pouvez le configurer pour vous connecter à la base de données, puis déployer le code Drupal dans l'environnement.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk à l'aide de ce lien préconfiguré : console.aws.amazon.com/elasticbeanstalk/home#/newApplication? Nom de l'application = Tutoriels et type d'environnement = LoadBalanced
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Create app (Créer une application).

La création d'un environnement prend environ 5 minutes et crée les ressources suivantes :

- EC2 instance — Une machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications Web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.
- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Équilibreur de charge – Équilibreur de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité d'équilibrage de charge : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibreur de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et qui sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack : Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

> **Sécurité du domaine**
>
> Pour renforcer la sécurité de vos applications Elastic Beanstalk, le domaine elasticbeanstalk.com est enregistré dans la liste des suffixes publics (PSL).

Si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Elastic Beanstalk, nous vous recommandons d'utiliser des cookies `__Host-` avec un préfixe pour une sécurité accrue. Cette pratique protège votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient. L'instance DB RDS que vous avez lancée se trouvant en dehors de votre environnement, vous êtes chargé de la gestion de son cycle de vie.

Note

Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour plus d'informations, consultez Utilisation d'Elastic Beanstalk avec Amazon S3.

Configuration des paramètres de sécurité et des propriétés de l'environnement

Ajoutez le groupe de sécurité de votre instance DB à votre environnement en cours d'exécution. Cette procédure conduit Elastic Beanstalk à remettre en service toutes les instances de votre environnement avec le groupe de sécurité supplémentaire associé.

Pour ajouter un groupe de sécurité à votre environnement

- Effectuez l'une des actions suivantes :
  - Pour ajouter un groupe de sécurité à l'aide de la console Elastic Beanstalk
    a. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
    b. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
    c. Dans le panneau de navigation, choisissez Configuration.
    d. Dans la catégorie de configuration Instances, choisissez Edit (Modifier).

e. Sous groupes EC2 de sécurité, choisissez le groupe de sécurité à associer aux instances, en plus du groupe de sécurité d'instance créé par Elastic Beanstalk.

f. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

g. Lisez l'avertissement, puis choisissez Confirm (Confirmer).

- Pour ajouter un groupe de sécurité à l'aide d'un fichier de configuration, utilisez le fichier exemple `securitygroup-addexisting.config`.

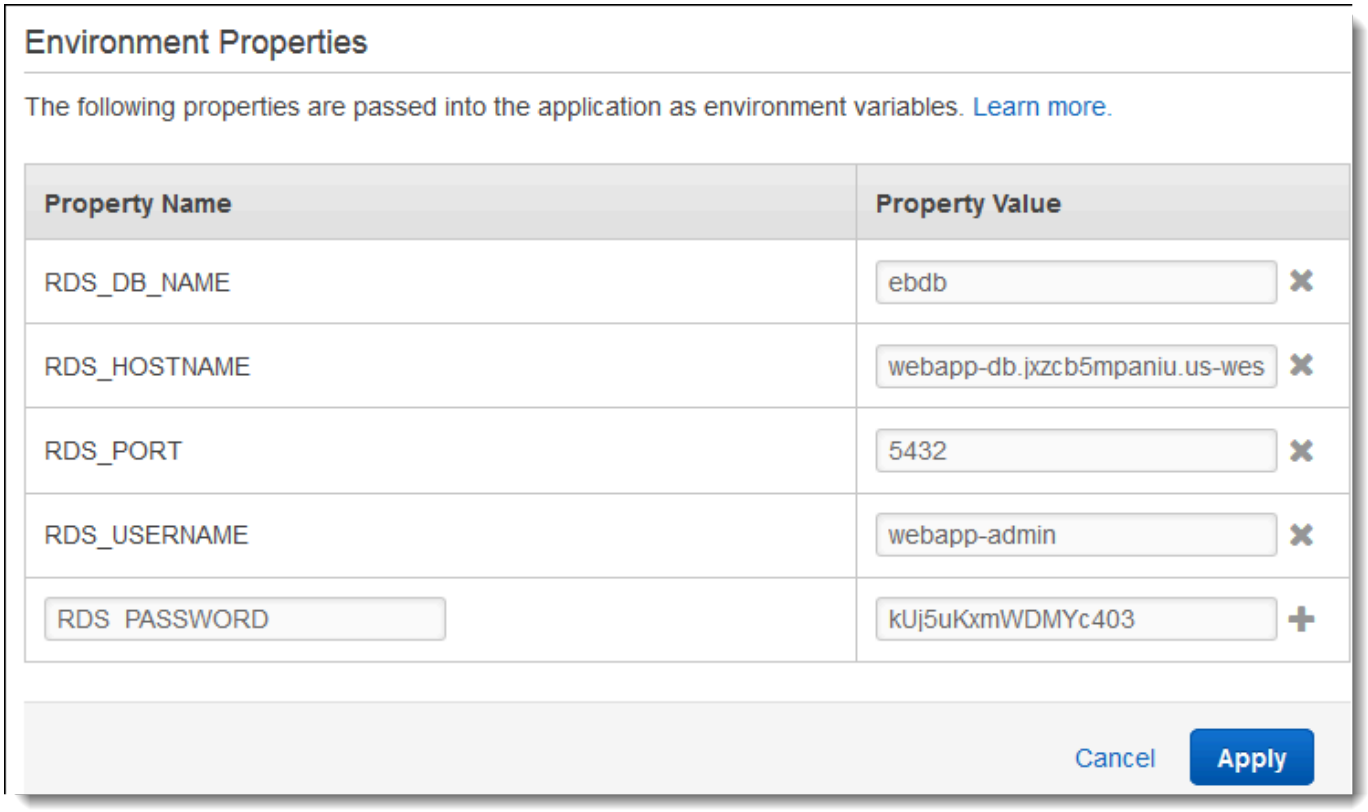
Ensuite, utilisez les propriétés de l'environnement pour transmettre les informations de connexion à votre environnement. L'exemple d'application utilise un ensemble de propriétés par défaut qui correspondent à celles qui sont configurées par Elastic Beanstalk lorsque vous mettez en service une base de données dans votre environnement.

Pour configurer les propriétés d'environnement pour une instance de base de données Amazon RDS

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Dans la section Environment properties (Propriétés de l'environnement), définissez les variables lues par votre application pour créer une chaîne de connexion. Pour assurer la compatibilité avec les environnements disposant d'une instance DB RDS intégrée, utilisez les noms et valeurs suivants : Vous pouvez trouver toutes les valeurs, à l'exception de votre mot de passe, dans la console RDS.

Nom de la propriété	Description	Valeur de la propriété
`RDS_HOSTNAME`	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).

Nom de la propriété	Description	Valeur de la propriété
RDS_PORT	Port sur lequel l'instanc e de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, **ebdb**.	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.



6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Une fois Drupal installé, vous devez vous connecter à l'instance avec SSH pour récupérer certains détails de configuration. Attribuez une clé SSH aux instances de votre environnement.

Pour configurer SSH

1. Si vous n'avez pas encore créé de paire de clés, ouvrez la page des paires de clés de la EC2 console Amazon et suivez les instructions pour en créer une.
2. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
3. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
4. Dans le panneau de navigation, choisissez Configuration.
5. Sous Security (Sécurité), choisissez Edit (Modifier).
6. Pour une paire de EC2 clés, choisissez votre paire de clés.
7. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Configuration et déploiement de votre application

Pour créer un projet Drupal pour Elastic Beanstalk, téléchargez le code source de Drupal et combinez-le avec les fichiers du référentiel aws-samples/ sur. eb-php-drupal GitHub

Pour créer un projet Drupal

1. Exécutez la commande suivante pour télécharger Drupal à partir de www.drupal.org/download. Pour en savoir plus sur les téléchargements, consultez le site Web de Drupal.

   Si la plateforme de votre environnement Elastic Beanstalk utilise PHP 7.4 ou version antérieure, nous vous recommandons de télécharger la version 8.9.13 de Drupal pour ce tutoriel. Pour ce faire, exécutez la commande suivante.

   ```
   ~$ curl https://ftp.drupal.org/files/projects/drupal-8.9.13.tar.gz -o drupal.tar.gz
   ```

   Si votre plateforme utilise PHP 8.0 ou version ultérieure, nous vous recommandons de télécharger Drupal 9.1.5. Pour ce faire, exécutez la commande suivante.

   ```
   ~$ curl https://ftp.drupal.org/files/projects/drupal-9.1.5.tar.gz -o drupal.tar.gz
   ```

   Pour plus d'informations sur les versions de Drupal et les versions de PHP qu'elles prennent en charge, consultez Exigences relatives à PHP dans la documentation officielle de Drupal. Les versions principales recommandées par Drupal sont répertoriées sur le site Web de Drupal.
2. Utilisez la commande suivante pour télécharger les fichiers de configuration à partir de l'exemple de référentiel :

   ```
   ~$ wget https://github.com/aws-samples/eb-php-drupal/releases/download/v1.1/eb-php-drupal-v1.zip
   ```
3. Extrayez Drupal et modifiez le nom du dossier.

   Si vous avez téléchargé Drupal 8.9.13 :

   ```
   ~$ tar -xvf drupal.tar.gz
   ~$ mv drupal-8.9.13 drupal-beanstalk
   ~$ cd drupal-beanstalk
   ```

   Si vous avez téléchargé Drupal 9.1.5 :

```
~$ tar -xvf drupal.tar.gz
~$ mv drupal-9.1.5 drupal-beanstalk
~$ cd drupal-beanstalk
```

4. Extrayez les fichiers de configuration sur l'installation de Drupal.

```
~/drupal-beanstalk$ unzip ../eb-php-drupal-v1.zip
 creating: .ebextensions/
 inflating: .ebextensions/dev.config
 inflating: .ebextensions/drupal.config
 inflating: .ebextensions/efs-create.config
 inflating: .ebextensions/efs-filesystem.template
 inflating: .ebextensions/efs-mount.config
 inflating: .ebextensions/loadbalancer-sg.config
 inflating: LICENSE
 inflating: README.md
 inflating: beanstalk-settings.php
```

Vérifiez que la structure de votre dossier `drupal-beanstalk` est correcte, comme illustré.

```
drupal-beanstalk$ tree -aL 1
.
### autoload.php
### beanstalk-settings.php
### composer.json
### composer.lock
### core
### .csslintrc
### .ebextensions
### .ebextensions
### .editorconfig
### .eslintignore
### .eslintrc.json
### example.gitignore
### .gitattributes
### .htaccess
### .ht.router.php
### index.php
### LICENSE
### LICENSE.txt
### modules
```

```
### profiles
### README.md
### README.txt
### robots.txt
### sites
### themes
### update.php
### vendor
### web.config
```

Le fichier `beanstalk-settings.php` du référentiel de projet utilise les variables d'environnement que vous avez définies à l'étape précédente pour configurer la connexion de base de données. Le dossier `.ebextensions` contient des fichiers de configuration qui créent des ressources supplémentaires au sein de votre environnement Elastic Beanstalk.

Les fichiers de configuration doivent être modifiés pour être compatibles avec votre compte. Remplacez les valeurs d'espace réservé dans les fichiers par les valeurs appropriées IDs et créez un ensemble de sources.

Pour mettre à jour les fichiers de configuration et créer une solution groupée source.

1. Modifiez les fichiers de configuration comme suit.

   - `.ebextensions/dev.config` – Restreint l'accès à votre environnement à votre adresse IP afin de le protéger lors du processus d'installation de Drupal. Remplacez l'adresse IP de l'espace réservé vers le haut du fichier par votre adresse IP publique.
   - `.ebextensions/efs-create.config` – Crée un système de fichiers EFS et des points de montage dans chaque zone de disponibilité/sous-réseau de votre VPC. Identifiez votre VPC et votre sous-réseau par défaut dans IDs la console Amazon VPC.

2. Créez un bundle source contenant les fichiers dans votre dossier de projet. La commande suivante permet de créer une solution groupée source nommée `drupal-beanstalk.zip`. Elle exclut les fichiers du dossier `vendor`, lesquels prennent beaucoup de place et ne sont pas nécessaires pour le déploiement de votre application dans Elastic Beanstalk.

   ```
   ~/eb-drupal$ zip ../drupal-beanstalk.zip -r * .[^.]* -x "vendor/*"
   ```

Chargez le bundle source sur Elastic Beanstalk pour déployer Drupal dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

Installation de Drupal

Pour terminer l'installation de Drupal

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez l'URL d'environnement pour ouvrir votre site dans un navigateur. Vous êtes redirigé vers un assistant d'installation Drupal car le site n'a pas encore été configuré.
4. Effectuez une installation standard avec les paramètres suivants pour la base de données :

   - Nom de la base de données – Nom de la base de données qui apparaît sur la console Amazon RDS.
   - Nom d'utilisateur de base de données et mot de passe – Valeurs des champs Identifiant principal et Mot de passe principal que vous avez saisies lors de la création de votre base de données.
   - Options avancées > Hôte – Valeur du champ Point de terminaison correspondant à l'instance de base de données, affichée dans la console Amazon RDS.

L'installation prend environ une minute.

Mise à jour de la configuration de Drupal et retrait de restrictions d'accès

Le processus d'installation de Drupal a créé un fichier nommé `settings.php` dans le dossier `sites/default` sur l'instance. Vous aurez besoin de ce fichier dans votre code source pour éviter de réinitialiser votre site lors des déploiements suivants, mais le fichier contient actuellement des codes secrets que vous ne souhaitez pas valider dans la source. Connectez-vous à l'instance d'application afin de récupérer les informations du fichier de paramètres.

Pour vous connecter à votre instance d'application avec SSH

1. Ouvrez la page des instances de la EC2 console Amazon.
2. Choisissez l'instance d'application. C'est celui qui porte le nom de votre environnement Elastic Beanstalk.
3. Choisissez Se connecter.
4. Suivez les instructions pour vous connecter à l'instance avec SSH. La commande ressemble à ce qui suit.

```
$ ssh -i ~/.ssh/mykey ec2-user@ec2-00-55-33-222.us-west-2.compute.amazonaws.com
```

Obtenez l'ID du répertoire de synchronisation dans la dernière ligne du fichier de paramètres.

```
[ec2-user ~]$ tail -n 1 /var/app/current/sites/default/settings.php
$config_directories['sync'] = 'sites/default/files/
config_4ccfX2sPQm79p1mk5IbUq9S_FokcENO4mxyC-L18-4g_xKj_7j9ydn31kDOYOgnzMu071Tvc4Q/
sync';
```

Le fichier contient également la clé de hachage actuelle des sites, mais vous pouvez ignorer la valeur actuelle et utiliser la votre.

Attribuez le chemin d'accès au répertoire et la clé de hachage aux propriétés de l'environnement. Le fichier de paramètres personnalisés du référentiel de projet lit ces propriétés pour configurer le site pendant le déploiement, en plus des propriétés de connexion de base de données que vous avez définies précédemment.

Propriétés de configuration de Drupal

- `SYNC_DIR` – Chemin d'accès au répertoire de synchronisation.
- `HASH_SALT` – Toute valeur de chaîne qui respecte les exigences de propriété d'environnement.

Pour configurer les variables d'environnement dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Faites défiler la page jusqu'à Variables d'environnement d'exécution.
6. Sélectionnez Ajouter une variable d'environnement.
7. Pour Source, sélectionnez Texte brut.

> **Note**
>
> Les valeurs Secrets Manager et SSM Parameter Store de la liste déroulante permettent de configurer les variables d'environnement en tant que secrets pour stocker des données sensibles, telles que les informations d'identification et les clés d'API. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic AWS Secrets Manager AWS Systems Manager Beanstalk avec et Parameter Store.

8. Entrez le nom de la variable d'environnement et les paires de valeurs de la variable d'environnement.
9. Si vous devez ajouter d'autres variables, répétez les étapes 6 à 8.
10. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Enfin l'exemple de projet inclut un fichier de configuration (`loadbalancer-sg.config`) qui crée un groupe de sécurité et l'affecte à l'équilibreur de charge de l'environnement, à l'aide de l'adresse IP que vous avez configurée dans `dev.config` afin de restreindre l'accès HTTP sur le port 80 aux connexions provenant de votre réseau. Sinon, un tiers pourrait se connecter éventuellement à votre site avant que vous ayez installé Drupal et configuré votre compte d'administrateur.

Pour mettre à jour la configuration de Drupal et retirer les restrictions d'accès

1. Supprimez le fichier `.ebextensions/loadbalancer-sg.config` de votre répertoire de projet.

```
~/drupal-beanstalk$ rm .ebextensions/loadbalancer-sg.config
```

2. Copiez le fichier `settings.php` personnalisé dans le dossier des sites.

```
~/drupal-beanstalk$ cp beanstalk-settings.php sites/default/settings.php
```

3. Créez une solution groupée source.

```
~/eb-drupal$ zip ../drupal-beanstalk-v2.zip -r * .[^.]* -x "vendor/*"
```

Chargez le bundle source sur Elastic Beanstalk pour déployer Drupal dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

Configuration de votre groupe Auto Scaling

Enfin, configurez le groupe Auto Scaling de votre environnement avec un nombre minimum d'instances plus élevé. Exécutez au moins deux instances en permanence afin d'empêcher que les serveurs web de votre environnement constituent un point de défaillance unique et pour vous permettre de déployer des modifications sans mettre votre site hors service.

Pour configurer le groupe Auto Scaling de votre environnement pour une haute disponibilité

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Auto Scaling group (Groupe Auto Scaling), définissez les Min instances (Instances min.) sur **2**.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Pour la prise en charge des chargements de contenu sur plusieurs instances, l'exemple de projet utilise Amazon Elastic File System pour créer un système de fichiers partagé. Créez une publication sur le site et chargez le contenu afin de le stocker sur le système de fichiers partagé. Affichez la publication et actualisez la page plusieurs fois pour atteindre les deux instances et vérifier que le système de fichiers partagé fonctionne.

Nettoyage

Une fois que vous avez fini d'utiliser le code de démonstration, vous pouvez mettre fin à votre environnement. Elastic Beanstalk supprime toutes les ressources AWS associées, telles que les instances EC2 Amazon, les instances de base de données, les équilibreurs de charge, lesgroupes de sécurité et les alarmes.

La suppression de ressources n'entraîne pas la suppression de l'application Elastic Beanstalk. Vous pouvez donc créer de nouveaux environnements pour votre application à tout moment.

Pour mettre fin à votre environnement Elastic Beanstalk depuis la console

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Terminate Environment (Résilier l’environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Vous pouvez également arrêter les ressources de base de données que vous avez créées hors de votre environnement Elastic Beanstalk. Lorsque vous résiliez une instance de base de données

Amazon RDS, vous pouvez prendre un instantané et restaurer les données dans une autre instance ultérieurement.

Pour résilier votre instance DB RDS

1. Ouvrez la console Amazon RDS.
2. Choisissez Databases (Bases de données).
3. Sélectionnez votre instance DB.
4. Choisissez Actions, puis Supprimer.
5. Choisissez si vous souhaitez créer un instantané, puis choisissez Delete (Supprimer).

Étapes suivantes

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le charger manuellement sur la console Elastic Beanstalk. L'interface de ligne de commande Elastic Beanstalk (EB CLI easy-to-use) fournit des commandes pour créer, configurer et déployer des applications dans les environnements Elastic Beanstalk à partir de la ligne de commande.

L'exemple d'application utilise des fichiers de configuration pour configurer les paramètres PHP et pour créer une table dans la base de données si elle n'existe pas déjà. Vous pouvez également utiliser un fichier de configuration pour configurer les paramètres de groupe de sécurité de vos instances lors de la création de l'environnement afin d'éviter les mises à jour de configuration, qui prennent du temps. Pour plus d'informations, consultez Personnalisation d'environnement avancée avec fichiers de configuration (`.ebextensions`).

Pour le développement et les tests, vous pouvez utiliser la fonctionnalité Elastic Beanstalk permettant d'ajouter directement une instance de base de données gérée à votre environnement. Pour savoir comment configurer une base de données dans votre environnement, consultez Ajout d'une base de données à votre environnement Elastic Beanstalk.

Si vous avez besoin d'une base de données hautes performances, envisagez d'utiliser Amazon Aurora. Amazon Aurora est un moteur de base de données compatible MySQL qui offre des fonctionnalités de base de données commerciales à faible coût. Pour connecter votre application à une autre base de données, répétez la procédure de configuration du groupe de sécurité et mettez à jour les propriétés d'environnement associées à RDS.

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez configurer un nom de domaine personnalisé pour votre environnement et activer HTTPS pour des connexions sécurisées.

Déploiement d'applications Python avec Elastic Beanstalk

Ce chapitre fournit des instructions pour configurer et déployer votre application Web Python sur AWS Elastic Beanstalk. Elastic Beanstalk facilite le déploiement, la gestion et le dimensionnement de vos applications Web Python à l'aide d'Amazon Web Services.

Vous pouvez déployer votre application en quelques minutes à l'aide de l'interface de ligne de commande Elastic Beanstalk (EB CLI) ou de la console Elastic Beanstalk. Après avoir déployé votre application Elastic Beanstalk, vous pouvez continuer à utiliser l'EB CLI pour gérer votre application et votre environnement, ou vous pouvez utiliser la console Elastic Beanstalk, ou le. AWS CLI APIs

Suivez les étapes décrites dans les step-by-step instructions QuickStart pour Python pour créer et déployer une application Web Python Hello World avec l'EB CLI.

Rubriques

- QuickStart: Déployer une application Python sur Elastic Beanstalk
- Configuration de votre environnement de développement Python pour Elastic Beanstalk
- Utilisation de la plateforme Python Elastic Beanstalk
- Déploiement d'une application Flask sur Elastic Beanstalk
- Déploiement d'une application Django sur Elastic Beanstalk
- Ajouter une instance de base de données Amazon RDS à votre environnement Python Elastic Beanstalk
- Outils et ressources Python

QuickStart: Déployer une application Python sur Elastic Beanstalk

Ce QuickStart didacticiel explique le processus de création d'une application Python et de son déploiement dans un AWS Elastic Beanstalk environnement.

> ⚠ Non destiné à une utilisation en production
>
> Les exemples sont destinés uniquement à des fins de démonstration. N'utilisez pas d'exemples d'applications en production.

Sections

- Votre AWS compte
- Conditions préalables
- Étape 1 : Création d'une application Python
- Étape 2 : Exécutez votre application localement
- Étape 3 : Déployez votre application Python avec l'EB CLI
- Étape 4 : Exécutez votre application sur Elastic Beanstalk
- Étape 5 : nettoyer
- AWS ressources pour votre application
- Étapes suivantes
- Déployez avec la console Elastic Beanstalk

Votre AWS compte

Si vous n'êtes pas encore AWS client, vous devez créer un AWS compte. L'inscription vous permet d'accéder à Elastic Beanstalk AWS et aux autres services dont vous avez besoin.

Si vous avez déjà un AWS compte, vous pouvez passer àConditions préalables.

Créez un AWS compte

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez l'https://portal.aws.amazon.com/billing/inscription.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d’inscription, vous recevrez un appel téléphonique ou un SMS et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

   Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l’utilisateur racine a accès à l’ensemble des Services AWS et des ressources de ce compte. La meilleure pratique de sécurité consiste à attribuer un accès administratif à un utilisateur, et à utiliser uniquement l’utilisateur racine pour effectuer les tâches nécessitant un accès utilisateur racine.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. À tout moment, vous pouvez consulter l'activité actuelle de votre compte et gérer votre compte en accédant à https://aws.amazon.com/et en choisissant Mon compte.

Création d’un utilisateur doté d’un accès administratif

Une fois que vous vous êtes inscrit à un utilisateur administratif Compte AWS, que vous Utilisateur racine d'un compte AWS l'avez sécurisé AWS IAM Identity Center, que vous l'avez activé et que vous en avez créé un, afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l’aide pour vous connecter en utilisant l’utilisateur racine, consultez Connexion en tant qu’utilisateur racine dans le Guide de l’utilisateur Connexion à AWS .
2. Activez l’authentification multifactorielle (MFA) pour votre utilisateur racine.

   Pour obtenir des instructions, voir Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d’un utilisateur doté d’un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d’ AWS IAM Identity Center dans le Guide de l’utilisateur AWS IAM Identity Center .
2. Dans IAM Identity Center, octroyez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connexion en tant qu’utilisateur doté d’un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l’URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l’utilisateur IAM Identity Center.

Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribution d’un accès à d’autres utilisateurs

1. Dans IAM Identity Center, créez un ensemble d’autorisations qui respecte la bonne pratique consistant à appliquer les autorisations de moindre privilège.

   Pour obtenir des instructions, consultez Création d’un ensemble d’autorisations dans le Guide de l’utilisateur AWS IAM Identity Center .
2. Attribuez des utilisateurs à un groupe, puis attribuez un accès par authentification unique au groupe.

   Pour obtenir des instructions, consultez Ajout de groupes dans le Guide de l’utilisateur AWS IAM Identity Center .

Conditions préalables

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

INTERFACE DE LIGNE DE COMMANDE (CLI) EB

Ce tutoriel utilise également l'interface de ligne de commande Elastic Beanstalk (CLI EB). Pour de plus amples informations sur l'installation et la configuration de la CLI EB, veuillez consulter Installation de la CLI EB avec un script de configuration (recommandé) et Configuration de l'interface de ligne de commande EB.

Framework Python et Flask

Vérifiez que vous disposez d'une version fonctionnelle de Python `pip` installée en exécutant les commandes suivantes.

```
~$ python3 --version
Python 3.N.N
>~$ python3 -m pip --version
pip X.Y.Z from ... (python 3.N.N)
```

Si l'une des commandes précédentes renvoie « Python introuvable », exécutez les commandes suivantes qui utilisent à la `python` place de`python3`. La configuration des alias et des liens symboliques peut varier en fonction du système d'exploitation et des personnalisations individuelles. Il est donc possible que la `python3` commande ne fonctionne pas sur votre machine.

```
~$ python --version
Python 3.N.N
>~$ python -m pip --version
pip X.Y.Z from ... (python 3.N.N)
```

Si Python n'est pas installé sur votre machine locale, vous pouvez le télécharger depuis la page de téléchargement de Python sur le site Web de Python. Pour obtenir la liste des versions du langage Python prises en charge par Elastic Beanstalk, consultez la section Plateformes Python AWS Elastic Beanstalk prises en charge dans le guide des plateformes. Le site Web de téléchargement de Python fournit un lien vers le guide du développeur Python, où vous trouverez des instructions d'installation et de configuration.

> **Note**
>
> Le `pip` package Python est inclus par défaut dans Python 3.4 ou version ultérieure.

Si votre résultat indique que vous disposez d'une version compatible de Python, mais que ce n'est pas le cas`pip`, consultez la page d'installation sur le site Web pip.pypa.io. Il fournit des conseils pour installer pip dans un environnement Python qui ne l'a pas.

Vérifiez si Flask est installé en exécutant la commande suivante :

```
~$ pip list | grep Flask
```

Si Flask n'est pas installé, vous pouvez l'installer à l'aide de la commande suivante :

```
~$ pip install Flask
```

Étape 1 : Création d'une application Python

Créez un répertoire de projet.

```
~$ mkdir eb-python
~$ cd eb-python
```

Créez un exemple « Hello Elastic Beanstalk ! » Application Python que vous déploierez à l'aide d'Elastic Beanstalk.

Créez un fichier texte nommé `application.py` dans le répertoire que vous venez de créer avec le contenu suivant.

Example~/eb-python/application.py

```
from flask import Flask
application = Flask(__name__)

@application.route('/')
def hello_elastic_beanstalk():
        return 'Hello Elastic Beanstalk!'
```

Créez un fichier texte nommé `requirements.txt` avec la ligne suivante. Ce fichier contient les `pip` packages nécessaires à l'exécution de l'application.

Example~/eb-python/requirements.txt

```
Flask
```

Étape 2 : Exécutez votre application localement

Exécutez la commande suivante pour exécuter votre application localement.

```
~/eb-python$ export FLASK_APP=application.py && flask run --port 5000
```

Vous devriez voir un résultat similaire à ce qui suit

```
Serving Flask app 'application.py'
Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a
 production WSGI server instead.
Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [01/Jan/1970 00:00:00] "GET / HTTP/1.1" 200 -
```

Accédez à `http://localhost:5000` dans votre navigateur Web. Le navigateur Web doit afficher « Hello Elastic Beanstalk ! ».

Étape 3 : Déployez votre application Python avec l'EB CLI

Exécutez les commandes suivantes pour créer un environnement Elastic Beanstalk pour cette application.

Pour créer un environnement et déployer votre application Python

1. Initialisez votre référentiel de la CLI EB avec la commande eb init :

   ```
   ~/eb-python$ eb init -p python-3.9 python-tutorial --region us-east-2
   ```

   Cette commande crée une application nommée `python-tutorial` et configure votre dépôt local pour créer des environnements avec la version de plate-forme Python fournie.
2. (facultatif) Exécutez à nouveau la commande eb init pour configurer une paire de clés par défaut afin de pouvoir vous connecter à l'instance EC2 qui exécute votre application.

   ```
   ~/eb-python$ eb init
   Do you want to set up SSH for your instances?
   (y/n): y
   Select a keypair.
   1) my-keypair
   2) [ Create new KeyPair ]
   ```

   Sélectionnez une paire de clés si vous en avez déjà une, ou suivez les invites pour en créer une. Si vous ne voyez pas l'invite ou que vous avez besoin de modifier vos paramètres ultérieurement, exécutez eb init -i.

3. Créez un environnement et déployez-y votre application avec eb create. Elastic Beanstalk crée automatiquement un fichier zip pour votre application et le démarre sur le port 5000.

   ```
   ~/eb-python$ eb create python-env
   ```

   Il faut environ cinq minutes à Elastic Beanstalk pour créer votre environnement.

Étape 4 : Exécutez votre application sur Elastic Beanstalk

Lorsque le processus de création de votre environnement est terminé, ouvrez votre site Web aveceb open.

```
~/eb-python$ eb open
```

Félicitations ! Vous avez déployé une application Python avec Elastic Beanstalk ! Celle-ci ouvre une fenêtre de navigation en utilisant le nom de domaine créé pour votre application.

Étape 5 : nettoyer

Vous pouvez mettre fin à votre environnement lorsque vous avez fini d'utiliser votre application. Elastic Beanstalk AWS met fin à toutes les ressources associées à votre environnement.

Pour mettre fin à votre environnement Elastic Beanstalk avec l'EB CLI, exécutez la commande suivante.

```
~/eb-python$ eb terminate
```

AWS ressources pour votre application

Vous venez de créer une application à instance unique. Il s'agit d'un exemple d'application simple avec une seule instance EC2, de sorte qu'il ne nécessite pas d'équilibrage de charge ni de dimensionnement automatique. Pour les applications à instance unique, Elastic Beanstalk crée les ressources suivantes : AWS

- Instance EC2 – Une machine virtuelle Amazon EC2 configurée pour exécuter des applications web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble distinct de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web

spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse qui traite le trafic web devant votre application web, lui transmet les demandes, traite les ressources statiques et génère des journaux d'accès et d'erreurs.

- Groupe de sécurité de l'instance – Un groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

Elastic Beanstalk gère toutes ces ressources. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

Étapes suivantes

Dès que vous disposez d'un environnement exécutant une application, vous pouvez à tout moment déployer une nouvelle version de l'application ou une application totalement différente. Le déploiement d'une nouvelle version d'application est très rapide, car il n'est pas nécessaire de mettre en service ni de redémarrer les instances EC2. Vous pouvez également explorer votre nouvel environnement à l'aide de la console Elastic Beanstalk. Pour connaître les étapes détaillées, consultez la section Explorez votre environnement dans le chapitre Mise en route de ce guide.

> **Essayez d'autres didacticiels**
>
> Si vous souhaitez essayer d'autres didacticiels avec différents exemples d'applications, consultez les didacticiels suivants :
>
> - Déploiement d'une application Flask sur Elastic Beanstalk

- Déploiement d'une application Django sur Elastic Beanstalk

Une fois que vous avez déployé un ou deux exemples d'applications et que vous êtes prêt à commencer à développer et à exécuter des applications Python localement, consultezConfiguration de votre environnement de développement Python pour Elastic Beanstalk.

Déployez avec la console Elastic Beanstalk

Vous pouvez également utiliser la console Elastic Beanstalk pour lancer l'exemple d'application. Pour connaître les étapes détaillées, voir Création d'un exemple d'application dans le chapitre Mise en route de ce guide.

Configuration de votre environnement de développement Python pour Elastic Beanstalk

Cette rubrique fournit des instructions pour configurer un environnement de développement Python afin de tester votre application localement avant de la déployer AWS Elastic Beanstalk. Il fait également référence à des sites Web qui fournissent des instructions d'installation pour des outils utiles.

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

Sections

- Conditions préalables
- Utilisation d’un environnement virtuel
- Configuration d'un projet Python pour Elastic Beanstalk

Conditions préalables

La liste suivante fournit les prérequis courants pour travailler avec Elastic Beanstalk et vos applications Python :

- Langage Python : installez la version du langage Python incluse dans la version de plateforme Elastic Beanstalk Python que vous avez choisie. Pour obtenir la liste des versions de langage Python prises en charge, consultez la section Plateformes Python prises en charge dans le guide AWS Elastic Beanstalk des plateformes. Si Python n'est pas encore configuré sur votre machine de développement, consultez la page de téléchargement de Python sur le site Web de Python.
- **`pip`**utilitaire — L'`pip`utilitaire est le programme d'installation de packages de Python. Il installe et répertorie les dépendances de votre projet, afin qu'Elastic Beanstalk sache comment configurer l'environnement de votre application. Pour plus d'informations`pip`, consultez la page pip sur le site Web pip.pypa.io.
- (Facultatif) L'interface de ligne de commande Elastic Beanstalk (EB CLI) : l'interface de ligne de commande EB peut empaqueter votre application avec les fichiers de déploiement nécessaires. Il peut également créer un environnement Elastic Beanstalk et y déployer votre application. Vous pouvez également effectuer des déploiements via la console Elastic Beanstalk, de sorte que l'EB CLI n'est pas strictement nécessaire.
- Une **`SSH`** installation fonctionnelle — Vous pouvez vous connecter à vos instances en cours d'exécution à l'aide du protocole SSH pour examiner ou déboguer un déploiement.
- **`virtualenv`**package — Cet `virtualenv` outil crée un environnement de développement et de test pour votre application. Elastic Beanstalk peut répliquer cet environnement sans installer de packages supplémentaires qui ne sont pas requis par votre application. Pour plus d'informations, consultez le site Web de virtualenv. Après avoir installé Python, vous pouvez installer le `virtualenv` package à l'aide de la commande suivante :

```
$ pip install virtualenv
```

Utilisation d’un environnement virtuel

Une fois les conditions préalables en place, configurez un environnement virtuel avec `virtualenv` pour installer les dépendances de votre application. En utilisant un environnement virtuel, vous pouvez déterminer exactement quels packages sont nécessaires à votre application afin que les packages requis soient installés sur les EC2 instances qui exécutent votre application.

Pour configurer un environnement virtuel

1. Ouvrez une fenêtre de ligne de commande et tapez :

   ```
   $ virtualenv /tmp/eb_python_app
   ```

   *eb_python_app*Remplacez-le par un nom adapté à votre application (utiliser le nom de votre application est une bonne idée). La commande `virtualenv` crée un environnement virtuel pour vous dans le répertoire spécifié et imprime les résultats de ses actions :

   ```
   Running virtualenv with interpreter /usr/bin/python
   New python executable in /tmp/eb_python_app/bin/python3.12
   Also creating executable in /tmp/eb_python_app/bin/python
   Installing setuptools, pip...done.
   ```

2. Une fois que votre environnement virtuel est prêt, démarrez-le en exécutant le script `activate` situé dans le répertoire `bin` de l'environnement. Par exemple, pour démarrer l'environnement eb_python_app créé à l'étape précédente, tapez :

   ```
   $ source /tmp/eb_python_app/bin/activate
   ```

   L'environnement virtuel affiche son nom (par exemple : `(eb_python_app)`) au début de chaque invite de commande, pour vous rappeler que vous êtes dans un environnement Python virtuel.

3. Pour arrêter d'utiliser votre environnement virtuel et revenir à l'interpréteur Python par défaut du système avec toutes les bibliothèques installées, exécutez la commande `deactivate`.

   ```
   (eb_python_app) $ deactivate
   ```

> **Note**
>
> Une fois l'environnement virtuel créé, vous pouvez le redémarrer à tout moment en exécutant à nouveau son script `activate`.

Configuration d'un projet Python pour Elastic Beanstalk

Vous pouvez utiliser l'interface de ligne de commande (CLI) Elastic Beanstalk pour préparer vos applications Python à déployer avec Elastic Beanstalk.

Pour configurer une application Python à déployer avec Elastic Beanstalk

1. Depuis votre environnement virtuel, revenez vers le haut de l'arborescence de votre projet (`python_eb_app`) et saisissez :

   ```
   pip freeze >requirements.txt
   ```

   Cette commande copie les noms et les versions des packages qui sont installés dans votre environnement virtuel dans `requirements.txt`. Par exemple, si le package PyYAML, version 3.11 est installé dans votre environnement virtuel, le fichier contiendra la ligne :

   ```
   PyYAML==3.11
   ```

   Cela permet à Elastic Beanstalk de répliquer l'environnement Python de votre application à l'aide des mêmes packages et de la même version que ceux utilisés pour développer et tester votre application.
2. Configurez le référentiel de l'interface de ligne de commande (CLI) EB avec la commande eb init. Suivez les invites pour choisir une région, une plateforme et d'autres options.

Par défaut, Elastic Beanstalk recherche un fichier appelé `application.py` pour démarrer votre application. S'il n'existe pas dans le projet Python que vous avez créé, certains ajustements de l'environnement de votre application sont nécessaires. Vous devrez également définir des variables d'environnement afin que les modules de votre application puissent être chargés. Pour plus d'informations, consultez Utilisation de la plateforme Python Elastic Beanstalk.

Utilisation de la plateforme Python Elastic Beanstalk

Cette rubrique explique comment configurer, créer et exécuter vos applications Python sur Elastic Beanstalk.

AWS Elastic Beanstalk prend en charge un certain nombre de branches de plate-forme pour différentes versions du langage de programmation Python. Voir Python dans le document AWS Elastic Beanstalk Platforms pour une liste complète.

Les applications Web Python peuvent s'exécuter derrière un serveur proxy avec WSGI. Elastic Beanstalk fournit Gunicorn comme serveur WSGI par défaut.

Vous pouvez ajouter un `Procfile` à votre solution groupée source pour spécifier et configurer le serveur WSGI pour votre application. Pour en savoir plus, consultez the section called “Procfile”.

Vous pouvez utiliser les fichiers `Pipfile.lock` et `Pipfile` créés par Pipenv pour spécifier les dépendances du package Python et d'autres exigences. Pour plus d'informations sur la spécification des dépendances, consultez the section called “Spécification des dépendances”.

Elastic Beanstalk propose des options de configuration que vous pouvez utiliser pour personnaliser le logiciel qui s'exécute EC2 sur les instances de votre environnement Elastic Beanstalk. Vous pouvez configurer des variables d'environnement nécessaires pour votre application, activer la rotation des journaux sur Amazon S3 et mapper des dossiers dans la source de votre application contenant des fichiers statiques vers des chemins desservis par le serveur proxy.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour modifier la configuration d'un environnement en cours d'exécution. Pour éviter de perdre la configuration de votre environnement en le résiliant, vous pouvez utiliser des configurations enregistrées pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des fichiers de configuration. Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

Les paramètres appliqués dans la console Elastic Beanstalk remplacent les mêmes paramètres des fichiers de configuration, s'ils existent. Cela vous permet d'utiliser les paramètres par défaut dans les fichiers de configuration et de les remplacer par des paramètres spécifiques à l'environnement dans la console. Pour plus d'informations sur la priorité et les autres méthodes de modification des paramètres, consultez Options de configuration.

Pour les packages Python disponibles à partir de `pip`, vous pouvez également inclure un fichier d'exigences dans la racine du code source de votre application. Elastic Beanstalk installe tous les

packages de dépendance spécifiés dans un fichier de configuration lors du déploiement. Pour en savoir plus, consultez the section called “Spécification des dépendances”.

Pour plus d'informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, consultez the section called “Extension des plateformes Linux”.

Configuration de votre environnement Python

Les paramètres de la plateforme Python vous permettent d'affiner le comportement de vos EC2 instances Amazon. Vous pouvez modifier la configuration de l'instance Amazon de l'environnement Elastic Beanstalk à l'aide EC2 de la console Elastic Beanstalk.

Utilisez la console Elastic Beanstalk pour configurer les AWS X-Ray paramètres des processus Python, activer, activer la rotation des journaux vers Amazon S3 et configurer les variables que votre application peut lire depuis l'environnement.

Pour configurer votre environnement Python dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.

Paramètres Python

- Proxy server (Serveur proxy) – Serveur proxy à utiliser sur vos instances d'environnement. Le serveur nginx est utilisé par défaut.
- Chemin WSGI – Nom du chemin d'accès à votre fichier d'application principal. Par exemple, `application.py` ou `django/wsgi.py`.
- NumProcesses— Le nombre de processus à exécuter sur chaque instance d'application.
- NumThreads— Le nombre de threads à exécuter dans chaque processus.

AWS X-Ray paramètres

- Démon X-Ray — Exécutez le AWS X-Ray démon pour traiter les données de trace provenant du. Kit SDK AWS X-Ray pour Python

Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Activer la rotation des fichiers journaux vers Amazon S3 : indique si les fichiers journaux des EC2 instances Amazon de votre application sont copiés dans le compartiment Amazon S3 associé à votre application.

Fichiers statiques

Pour améliorer les performances, la section des Fichiers statiques vous permet de configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Pour chaque répertoire, vous définissez le chemin virtuel sur le mappage de répertoires. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Pour en savoir plus sur la configuration des fichiers statiques à l'aide des fichiers de configuration ou de la console Elastic Beanstalk, consultez the section called “Fichiers statiques”.

Par défaut, le serveur proxy d'un environnement Python sert tous les fichiers dans un dossier nommé `static` sur le chemin d'accès `/static`. Par exemple, si votre code source d'application contient un fichier nommé `logo.png` dans un dossier nommé `static`, le serveur proxy le sert aux utilisateurs dans *`subdomain`*`.elasticbeanstalk.com/static/logo.png`. Vous pouvez configurer d'autres mappages comme expliqué dans cette section.

Propriétés de l'environnement

Vous pouvez utiliser les propriétés de l'environnement afin de fournir des informations à votre application et de configurer des variables d'environnement. Par exemple, vous pouvez créer une

propriété de l'environnement nommée `CONNECTION_STRING` qui spécifie une chaîne de connexion que votre application peut utiliser pour se connecter à une base de données.

A l'intérieur de l'environnement Python s'exécutant dans Elastic Beanstalk, ces valeurs sont accessibles à l'aide du dictionnaire `os.environ` Python. Pour plus d'informations, consultez http://docs.python. org/library/os.html.

Vous pouvez utiliser un code similaire au suivant pour accéder aux clés et aux paramètres :

```
import os
endpoint = os.environ['API_ENDPOINT']
```

Les propriétés de l'environnement peuvent également fournir des informations à une infrastructure. Par exemple, vous pouvez créer une propriété nommée `DJANGO_SETTINGS_MODULE` pour configurer Django pour utiliser un module de paramètres spécifique. Selon l'environnement, cette valeur peut être `development.settings`, `production.settings`, etc.

Pour plus d'informations, consultez Variables d'environnement et autres paramètres du logiciel.

Espaces de noms de la configuration Python

Vous pouvez utiliser un fichier de configuration pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être spécifiques à la plate-forme ou s'appliquer à toutes les plateformes du service Elastic Beanstalk dans son ensemble. Les options de configuration sont organisées en espaces de noms.

La plateforme Python définit des options dans les espaces de noms `aws:elasticbeanstalk:environment:proxy`, `aws:elasticbeanstalk:environment:proxy:staticfiles` et `aws:elasticbeanstalk:container:python`.

L'exemple suivant de fichier de configuration spécifie des paramètres d'option de configuration pour créer une propriété d'environnement nommée `DJANGO_SETTINGS_MODULE`, deux options de fichiers statiques qui mappent un répertoire nommé `statichtml` avec le chemin d'accès `/html`, un répertoire nommé `staticimages` avec le chemin d'accès `/images` et des paramètres supplémentaires dans l'espace de noms `aws:elasticbeanstalk:container:python`. Cet espace de noms contient des options qui vous permettent de spécifier l'emplacement du script WSGI dans votre code source et le nombre de threads et de processus à exécuter dans WSGI.

```
option_settings:
```

```
  aws:elasticbeanstalk:application:environment:
    DJANGO_SETTINGS_MODULE: production.settings
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache
  aws:elasticbeanstalk:environment:proxy:staticfiles:
    /html: statichtml
    /images: staticimages
  aws:elasticbeanstalk:container:python:
    WSGIPath: ebdjango.wsgi:application
    NumProcesses: 3
    NumThreads: 20
```

Remarques

- Si vous utilisez une version de plateforme AMI Python Amazon Linux (antérieure à Amazon Linux 2), remplacez la valeur `WSGIPath` par `ebdjango/wsgi.py`. La valeur de l'exemple fonctionne avec le serveur WSGI Gunicorn, qui n'est pas pris en charge sur les versions de la plateforme AMI Amazon Linux.
- De plus, ces anciennes versions de plateforme utilisent un espace de noms différent pour configurer les fichiers statiques `aws:elasticbeanstalk:container:python:staticfiles`. Il a les mêmes noms d'option et la même sémantique que l'espace de noms de fichier statique standard.

Les fichiers de configuration prennent également en charge plusieurs clés permettant de modifier davantage le logiciel sur les instances de votre environnement. Cet exemple utilise la clé de packages pour installer Memcached avec `yum` et des commandes de conteneur pour exécuter des commandes qui configurent le serveur durant le déploiement :

```
packages:
  yum:
    libmemcached-devel: '0.31'

container_commands:
  collectstatic:
    command: "django-admin.py collectstatic --noinput"
  01syncdb:
    command: "django-admin.py syncdb --noinput"
    leader_only: true
  02migrate:
```

```
    command: "django-admin.py migrate"
    leader_only: true
  03wsgipass:
    command: 'echo "WSGIPassAuthorization On" >> ../wsgi.conf'
  99customize:
    command: "scripts/customize.sh"
```

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d' AWS CLI. Pour plus d'informations, consultez Options de configuration.

L'**python3**exécutable

La version de l'`python3`exécutable disponible sur les EC2 instances dans les environnements Python d'Elastic Beanstalk ne correspondra pas toujours à la même version de Python utilisée par la plateforme. Par exemple, sur la plate-forme Python 3.12 AL2 023, `/usr/bin/python3` pointe vers Python 3.9. Cela est dû au fait que Python 3.9 est le système Python sur AL2 023. Pour plus d'informations, consultez Python in AL2 023 dans le guide de l'utilisateur Amazon Linux 2023. Vous pouvez accéder à un exécutable correspondant à la version de Python utilisée par la plateforme à un emplacement versionné (par exemple`/usr/bin/python3.12`) ou dans le `bin` répertoire de l'environnement virtuel de l'application (par exemple`/var/app/venv/staging-LQM1lest/bin/python3`). La plate-forme utilise le bon exécutable Python correspondant à la branche de plate-forme.

Configuration du serveur WSGI avec un profil sur Elastic Beanstalk

Vous pouvez ajouter un `Procfile` à votre solution groupée source pour spécifier et configurer le serveur WSGI pour votre application. Vous pouvez spécifier des commandes de démarrage et d'exécution personnalisées dans le`Procfile`.

Lorsque vous utilisez un `Procfile`, il remplace les options d'espace de noms `aws:elasticbeanstalk:container:python` que vous définissez à l'aide des fichiers de configuration.

L'exemple suivant utilise un `Procfile` pour spécifier UWSGi comme serveur et le configurer.

Example Procfile

```
web: uwsgi --http :8000 --wsgi-file application.py --master --processes 4 --threads 2
```

L'exemple suivant utilise un `Procfile` pour configurer Gunicorn, le serveur WSGI par défaut.

Example Procfile

```
web: gunicorn --bind :8000 --workers 3 --threads 2 project.wsgi:application
```

Remarques

- Si vous configurez un serveur WSGI autre que Gunicorn, assurez-vous de le spécifier également en tant que dépendance de votre application, afin qu'il soit installé sur vos instances d'environnement. Pour plus d'informations sur la spécification de dépendance, consultez the section called “Spécification des dépendances”.
- Le port par défaut du serveur WSGI est 8 000. Si vous spécifiez un numéro de port différent dans votre commande `Procfile`, définissez également la propriété d'environnement `PORT` sur ce numéro de port.

Spécification des dépendances à l'aide d'un fichier d'exigences sur Elastic Beanstalk

Cette rubrique décrit comment configurer votre application pour installer les autres packages Python dont elle a besoin. Une application Python classique comporte des dépendances par rapport à d'autres packages Python tiers. Avec la plateforme Elastic Beanstalk Python, vous disposez de plusieurs méthodes pour spécifier les packages Python dont dépend votre application.

Utilisation de **`pip`** et **`requirements.txt`**

L'outil standard pour installer les packages Python est `pip`. Il s'agit d'une fonctionnalité qui vous permet de spécifier tous les packages dont vous avez besoin (ainsi que leurs versions) dans un fichier Requirements unique. Pour plus d'informations sur le fichier d'exigences, consultez Format du fichier d'exigences (français non garanti) sur le site web de documentation de pip.

Créez un fichier `requirements.txt` et placez-le dans le répertoire de niveau supérieur de votre solution groupée source. Voici un exemple de fichier `requirements.txt` pour Django.

```
Django==2.2
mysqlclient==2.0.3
```

Dans votre environnement de développement, vous pouvez utiliser la commande `pip freeze` pour générer votre fichier Requirements.

```
~/my-app$ pip freeze > requirements.txt
```

Pour vous assurer que votre fichier Requirements contient uniquement des packages qui sont réellement utilisés par votre application, utilisez un environnement virtuel sur lequel seuls ces packages sont installés. En dehors d'un environnement virtuel, le résultat de `pip freeze` comprendra tous les packages `pip` installés sur votre ordinateur de développement, y compris ceux livrés avec votre système d'exploitation.

> **Note**
>
> Sur les versions de la plateforme AMI Python Amazon Linux, Elastic Beanstalk ne prend pas en charge nativement Pipenv ou Pipfiles. Si vous utilisez Pipenv pour gérer les dépendances de votre application, exécutez la commande suivante pour générer un fichier `requirements.txt`.
>
> ```
> ~/my-app$ pipenv lock -r > requirements.txt
> ```
>
> Pour en savoir plus, consultez Generating a requirements.txt dans la documentation.

Utilisation de Pipenv et `Pipfile`

Pipenv est un outil d'emballage Python moderne. Il combine l'installation de packages avec la création et la gestion d'un fichier de dépendance et d'un environnement virtuel pour votre application. Pour plus d'informations, consultez Pipenv: Python Dev Workflow for Humans.

Pipenv maintient deux fichiers :

- `Pipfile` – Ce fichier contient différents types de dépendances et d'exigences.
- `Pipfile.lock` – Ce fichier contient un instantané de version qui permet des créations déterministes.

Vous pouvez créer ces fichiers dans votre environnement de développement et les inclure dans le répertoire de premier niveau de la solution groupée source que vous déployez sur Elastic Beanstalk.

Pour plus d'informations sur ces deux fichiers, consultez Exemples de Pipfile et de Pipfile.lock (français non garanti).

L'exemple suivant utilise Pipenv pour installer Django et le framework REST Django. Ces commandes créent les fichiers `Pipfile` et `Pipfile.lock`.

```
~/my-app$ pipenv install django
~/my-app$ pipenv install djangorestframework
```

Priorité

Si vous incluez plusieurs fichiers d'exigences décrits dans cette rubrique, Elastic Beanstalk n'en utilise qu'un. La liste suivante montre la priorité, par ordre décroissant.

1. `requirements.txt`
2. `Pipfile.lock`
3. `Pipfile`

> **Note**
>
> À partir de la version de la plateforme Amazon Linux 2 du 7 mars 2023, si vous fournissez plusieurs de ces fichiers, Elastic Beanstalk émettra un message de console indiquant lequel des fichiers de dépendances a été utilisé lors d'un déploiement.

Les étapes suivantes décrivent la logique suivie par Elastic Beanstalk pour installer les dépendances lors du déploiement d'une instance.

- S'il existe un fichier `requirements.txt`, nous utilisons la commande `pip install -r requirements.txt`.
- À partir de la version de la plateforme Amazon Linux 2 du 7 mars 2023, s'il n'y a pas de fichier `requirements.txt`, mais qu'il y a un fichier `Pipfile.lock`, nous utilisons la commande `pipenv sync`. Avant cette version, nous utilisions `pipenv install --ignore-pipfile`.
- S'il n'y a ni fichier `requirements.txt` ni fichier `Pipfile.lock`, mais qu'il y en a un fichier `Pipfile`, on utilise la commande `pipenv install --skip-lock`.

- Si aucun des trois fichiers d'exigences n'est trouvé, nous n'installons aucune dépendance d'application.

Déploiement d'une application Flask sur Elastic Beanstalk

Ce didacticiel vous explique le processus de génération d'une application Flask et de son déploiement dans un AWS Elastic Beanstalk environnement. Flask est une infrastructure d'application web open source pour Python.

Dans le cadre de ce didacticiel, vous effectuerez les tâches suivantes :

- Configuration d'un environnement virtuel Python avec Flask
- Création d'une application Flask
- Déploiement de votre site avec l'interface de ligne de commande (CLI) EB
- Nettoyage

Conditions préalables

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans Découvrez comment démarrer avec Elastic Beanstalk pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

Dans ce didacticiel, nous utilisons Python 3.11 et la version correspondante de la plateforme Elastic Beanstalk. Installez Python en suivant les instructions à l'adresse Configuration de votre environnement de développement Python pour Elastic Beanstalk.

L'infrastructure Flask sera installée dans le cadre de ce didacticiel.

Ce didacticiel utilise également l'interface de ligne de commande (CLI) Elastic Beanstalk (EB). Pour de plus amples informations sur l'installation et la configuration de la CLI EB, veuillez consulter Installation de la CLI EB avec un script de configuration (recommandé) et Configuration de l'interface de ligne de commande EB.

Configuration d'un environnement virtuel Python avec Flask

Créez un répertoire de projet et un environnement virtuel pour votre application, puis installez Flask.

Pour configurer votre environnement de projet

1. Créez un répertoire de projet.

```
~$ mkdir eb-flask
~$ cd eb-flask
```

2. Créez et activez un environnement virtuel nommé `virt` :

```
~/eb-flask$ virtualenv virt
~$ source virt/bin/activate
(virt) ~/eb-flask$
```

   Vous verrez `(virt)` ajouté à votre invite de commande, ce qui indique que vous êtes dans un environnement virtuel. Utilisez l'environnement virtuel pour le reste du didacticiel.

3. Installez Flask avec `pip install` :

```
(virt)~/eb-flask$ pip install flask==3.0.3
```

4. Affichez les bibliothèques installées avec `pip freeze` :

```
(virt)~/eb-flask$ pip freeze
blinker==1.8.2
click==8.1.7
Flask==3.0.3
importlib_metadata==8.5.0
itsdangerous==2.2.0
Jinja2==3.1.4
MarkupSafe==2.1.5
Werkzeug==3.0.4
```

```
zipp==3.20.2
```

Cette commande répertorie tous les packages installés dans votre environnement virtuel. Étant donné que vous êtes dans un environnement virtuel, globalement les packages installés tels que l'interface de ligne de commande (CLI) EB ne sont pas affichés.

5. Enregistrez la sortie de `pip freeze` dans un fichier nommé `requirements.txt`.

```
(virt)~/eb-flask$ pip freeze > requirements.txt
```

Ce fichier indique à Elastic Beanstalk d'installer les bibliothèques pendant le déploiement. Pour de plus amples informations, veuillez consulter Spécification des dépendances à l'aide d'un fichier d'exigences sur Elastic Beanstalk.

Création d'une application Flask

Créez ensuite une application qui vous allez déployer à l'aide d'Elastic Beanstalk. Nous allons créer un service RESTful Web « Hello World ».

Créez un nouveau fichier texte dans ce répertoire nommé `application.py` avec le contenu suivant :

Example~/eb-flask/application.py

```
from flask import Flask

# print a nice greeting.
def say_hello(username = "World"):
    return '<p>Hello %s!</p>\n' % username

# some bits of text for the page.
header_text = '''
    <html>\n<head> <title>EB Flask Test</title> </head>\n<body>'''
instructions = '''
    <p><em>Hint</em>: This is a RESTful web service! Append a username
    to the URL (for example: <code>/Thelonious</code>) to say hello to
    someone specific.</p>\n'''
home_link = '<p><a href="/">Back</a></p>\n'
footer_text = '</body>\n</html>'

# EB looks for an 'application' callable by default.
```

```
application = Flask(__name__)

# add a rule for the index page.
application.add_url_rule('/', 'index', (lambda: header_text +
    say_hello() + instructions + footer_text))

# add a rule when the page is accessed with a name appended to the site
# URL.
application.add_url_rule('/<username>', 'hello', (lambda username:
    header_text + say_hello(username) + home_link + footer_text))

# run the app.
if __name__ == "__main__":
    # Setting debug to True enables debug output. This line should be
    # removed before deploying a production app.
    application.debug = True
    application.run()
```

Cet exemple affiche un message d'accueil personnalisé qui varie selon le chemin d'accès utilisé pour accéder au service.

> **Note**
>
> En ajoutant `application.debug = True` avant d'exécuter l'application, la sortie de débogage est activée pour parer à une éventuelle défaillance. C'est une bonne pratique pour le développement, mais vous devez supprimer les instructions de débogage dans le code de production, car la sortie de débogage peut révéler des aspects internes de votre application.

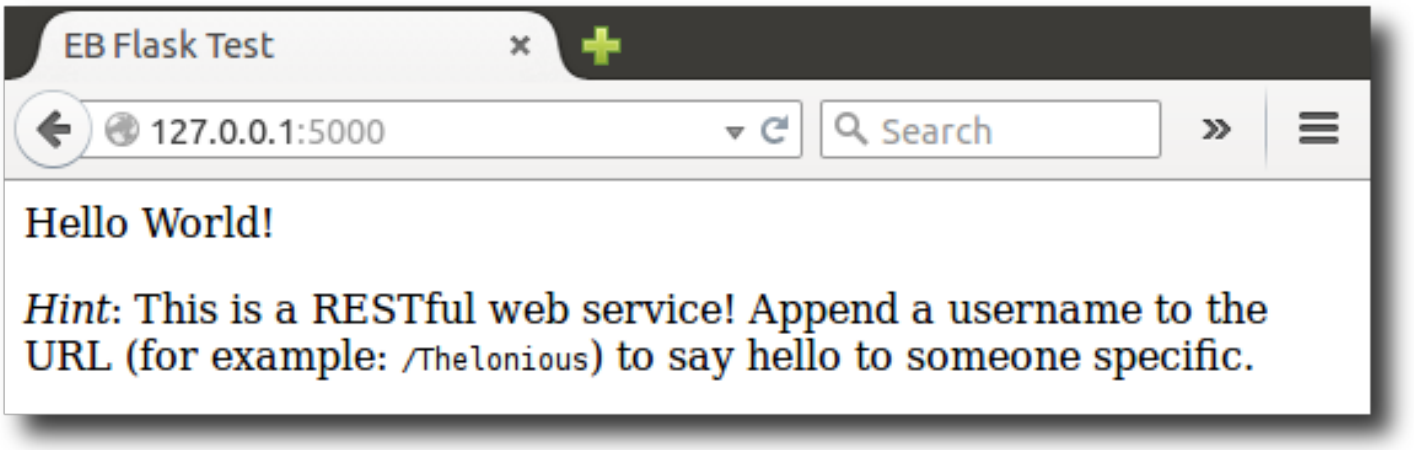
Utiliser `application.py` comme nom de fichier et fournir un objet `application` joignable (dans ce cas, l'objet Flask) permet à Elastic Beanstalk de trouver facilement le code de votre application.

Exécutez `application.py` avec Python :

```
(virt) ~/eb-flask$ python application.py
 * Serving Flask app "application" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

```
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 313-155-123
```

Ouvrez `http://127.0.0.1:5000/` dans votre navigateur web. Vous devriez voir l'application en cours d'exécution, affichant la page d'index :



Consultez le journal du serveur pour voir le résultat de votre demande. Vous pouvez arrêter le serveur web et revenir à votre environnement virtuel en appuyant sur Ctrl+C.

Si vous avez obtenu la sortie de débogage à la place, corrigez les erreurs et veillez à ce que l'application s'exécute localement avant de procéder à la configuration pour Elastic Beanstalk.

Déploiement de votre site avec l'interface de ligne de commande (CLI) EB

Vous avez ajouté tout ce dont vous avez besoin pour déployer votre application sur Elastic Beanstalk. Votre répertoire de projet devrait maintenant ressembler à ceci :

```
~/eb-flask/
|-- virt
|-- application.py
`-- requirements.txt
```

Le dossier `virt`, toutefois, n'est pas requis pour que l'application s'exécute sur Elastic Beanstalk. Au moment du déploiement, Elastic Beanstalk crée un environnement virtuel sur les instances de serveur et installe les bibliothèques répertoriées dans `requirements.txt`. Pour réduire la taille du bundle source que vous chargez pendant le déploiement, ajoutez un fichier .ebignore qui demande à l'interface de ligne de commande (CLI) EB d'exclure le dossier `virt`.

Example~/eb-flask/.ebignore

```
virt
```

Ensuite, vous allez créer votre environnement d'applications et déployer votre application configurée avec Elastic Beanstalk.

Pour créer un environnement et déployer votre application Flask

1. Initialisez votre référentiel d'interface de ligne de commande (CLI) EB avec la commande eb init :

   ```
   ~/eb-flask$ eb init -p python-3.11 flask-tutorial --region us-east-2
   Application flask-tutorial has been created.
   ```

   Cette commande crée une nouvelle application nommée `flask-tutorial` et configure votre dépôt local pour créer des environnements avec la dernière version de la plateforme Python 3.11.
2. (facultatif) Exécutez à eb init nouveau pour configurer une paire de clés par défaut afin de pouvoir vous connecter à l' EC2 instance qui exécute votre application avec SSH :

   ```
   ~/eb-flask$ eb init
   Do you want to set up SSH for your instances?
   (y/n): y
   Select a keypair.
   1) my-keypair
   2) [ Create new KeyPair ]
   ```

   Sélectionnez une paire de clés si vous en avez déjà une, ou suivez les invites pour en créer une. Si vous ne voyez pas l'invite ou que vous avez besoin de modifier vos paramètres ultérieurement, exécutez eb init -i.
3. Créez un environnement et déployez-y votre application avec eb create :

   ```
   ~/eb-flask$ eb create flask-env
   ```

La création d'un environnement prend environ 5 minutes et crée les ressources suivantes :

- EC2 instance — Une machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications Web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Équilibreur de charge – Équilibreur de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité d'équilibrage de charge : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibreur de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et qui sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack : Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

> **Sécurité du domaine**
>
> Pour renforcer la sécurité de vos applications Elastic Beanstalk, le domaine elasticbeanstalk.com est enregistré dans la liste des suffixes publics (PSL).
> Si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Elastic Beanstalk, nous vous recommandons d'utiliser des cookies `__Host-`

avec un préfixe pour une sécurité accrue. Cette pratique protège votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

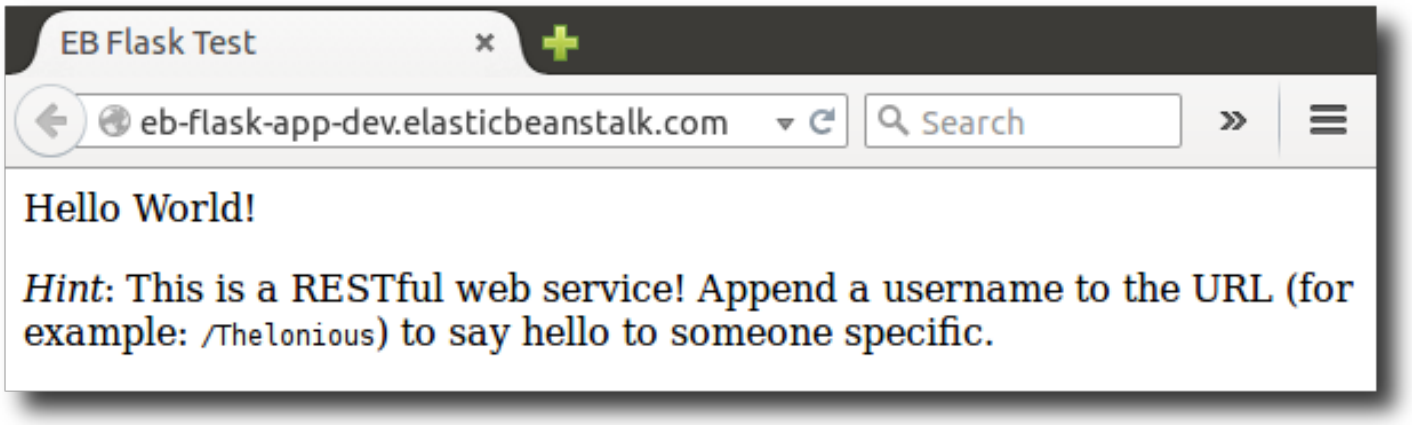
Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

> **Note**
>
> Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon S3.

Lorsque le processus de création de l'environnement est terminé, ouvrez votre site web avec eb open :

```
~/eb-flask$ eb open
```

Cela ouvre une fenêtre de navigateur à l'aide du nom de domaine créé pour votre application. Vous devez voir le même site web Flask que celui que vous avez créé et testé localement.



Si vous ne voyez pas votre application en cours d'exécution ou si vous obtenez un message d'erreur, consultez Déploiements afin d'obtenir de l'aide concernant la façon de déterminer la cause de l'erreur.

Si vous voyez effectivement votre application en cours d'exécution, alors félicitations, vous avez déployé votre première application Flask avec Elastic Beanstalk !

Nettoyage

Une fois que vous avez fini d'utiliser le code de démonstration, vous pouvez mettre fin à votre environnement. Elastic Beanstalk supprime toutes les ressources AWS associées, telles que les instances EC2 Amazon, les instances de base de données, les équilibreurs de charge, lesgroupes de sécurité et les alarmes.

La suppression de ressources n'entraîne pas la suppression de l'application Elastic Beanstalk. Vous pouvez donc créer de nouveaux environnements pour votre application à tout moment.

Pour mettre fin à votre environnement Elastic Beanstalk depuis la console

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Terminate Environment (Résilier l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Ou, avec l'interface de ligne de commande (CLI) EB :

```
~/eb-flask$ eb terminate flask-env
```

Étapes suivantes

Pour plus d'informations sur Flask, visitez flask.pocoo.org.

Si vous souhaitez essayer un autre cadre web Python, veuillez consulter Déploiement d'une application Django sur Elastic Beanstalk.

Déploiement d'une application Django sur Elastic Beanstalk

Ce didacticiel vous guide tout au long du déploiement d'un site web Django par défaut, généré automatiquement, sur un environnement AWS Elastic Beanstalk exécutant Python. Ce tutoriel vous

montre comment héberger une application web Python dans le cloud à l'aide d'un environnement Elastic Beanstalk.

Dans le cadre de ce didacticiel, vous effectuerez les tâches suivantes :

- Configuration d'un environnement virtuel Python et installation de Django
- Création d'un projet Django
- Configurer votre application Django pour Elastic Beanstalk
- Déploiement de votre site avec l'interface de ligne de commande (CLI) EB
- Mise à jour de votre application
- Nettoyage

Conditions préalables

Pour suivre ce didacticiel, la Configuration de votre environnement de développement Python doit être terminée et les packages suivants installés :

- Python 3.7 ou version ultérieure
- `pip`
- `virtualenv`
- `awsebcli`

L'infrastructure Django est installée dans le cadre de ce tutoriel.

> **Note**
>
> Créer des environnements avec l'interface de ligne de commande (CLI) EB nécessite un rôle de service. Vous pouvez créer un rôle de service en créant un environnement dans la console Elastic Beanstalk. Si vous n'avez pas de rôle de service, l'interface de ligne de commande (CLI) EB essaie d'en créer un lorsque vous exécutez `eb create`.

Configuration d'un environnement virtuel Python et installation de Django

Créez un environnement virtuel avec `virtualenv` et utilisez-le pour installer Django et ses dépendances. En utilisant un environnement virtuel, vous pouvez savoir exactement de quels

packages votre application a besoin, afin que les packages requis soient installés sur les EC2 instances Amazon qui exécutent votre application.

Les étapes suivantes illustrent les commandes que vous devez entrer pour les systèmes Unix et Windows, affichées sur des onglets distincts.

Pour configurer votre environnement virtuel

1. Créez un environnement virtuel nommé `eb-virt`.

   Unix-based systems

   ```
   ~$ virtualenv ~/eb-virt
   ```

   Windows

   ```
   C:\> virtualenv %HOMEPATH%\eb-virt
   ```

2. Activez l'environnement virtuel.

   Unix-based systems

   ```
   ~$ source ~/eb-virt/bin/activate
   (eb-virt) ~$
   ```

   Windows

   ```
   C:\>%HOMEPATH%\eb-virt\Scripts\activate
   (eb-virt) C:\>
   ```

   Vous verrez `(eb-virt)` ajouté à votre invite de commande, ce qui indique que vous êtes dans un environnement virtuel.

   > **Note**
   >
   > Les instructions restantes montrent l'invite de commande Linux dans votre répertoire de base ~$. Sous Windows`C:\Users\`*USERNAME*>, voici où se *USERNAME* trouve votre nom de connexion Windows.

3. Utilisez `pip` pour installer Django.

```
(eb-virt)~$ pip install django==2.2
```

> **Note**
>
> La version Django que vous installez doit être compatible avec la version Python sur la configuration Elastic Beanstalk Python que vous choisissez pour déployer votre application. Pour plus d'informations sur le déploiement, consultez ??? dans cette rubrique.
> Pour plus d'informations sur les versions actuelles de la plateforme Python, consultez Python dans le document Plateformes prises en charge par AWS Elastic Beanstalk .
> Pour plus d'informations sur la compatibilité de la version Django avec Python, consultez Quelle version de Python puis-je utiliser avec Django ?

4. Pour vérifier que Django est installé, entrez ce qui suit.

```
(eb-virt)~$ pip freeze
Django==2.2
...
```

Cette commande répertorie tous les packages installés dans votre environnement virtuel. Plus tard, vous utiliserez le résultat de cette commande pour configurer votre projet à utiliser avec Elastic Beanstalk.

Création d'un projet Django

Vous êtes maintenant prêt à créer un projet Django et à l'exécuter sur votre ordinateur, à l'aide de l'environnement virtuel.

> **Note**
>
> Ce didacticiel utilise SQLite un moteur de base de données inclus dans Python. La base de données est déployée avec vos fichiers de projet. Pour les environnements de production, nous vous recommandons d'utiliser Amazon Relational Database Service (Amazon RDS) et de le séparer de votre environnement. Pour de plus amples informations, veuillez consulter

[Ajouter une instance de base de données Amazon RDS à votre environnement Python Elastic Beanstalk](#).

Pour générer une application Django

1. Activez votre environnement virtuel.

   Unix-based systems

   ```
   ~$ source ~/eb-virt/bin/activate
   (eb-virt) ~$
   ```

   Windows

   ```
   C:\>%HOMEPATH%\eb-virt\Scripts\activate
   (eb-virt) C:\>
   ```

   Vous verrez le préfixe `(eb-virt)` ajouté à votre invite de commande, ce qui indique que vous êtes dans un environnement virtuel.

   > **Note**
   >
   > Les instructions restantes montrent l'invite de commande Linux ~$ dans votre répertoire de base et le répertoire de base Linux ~/. Sous Windows`C:\Users\`*USERNAME*>, voici où se *USERNAME* trouve votre nom de connexion Windows.

2. Utilisez la commande `django-admin startproject` pour créer un projet Django nommé `ebdjango`.

   ```
   (eb-virt)~$ django-admin startproject ebdjango
   ```

   Cette commande crée un site Django standard nommé ebdjango avec la structure de répertoires suivante.

   ```
   ~/ebdjango
     |-- ebdjango
     |   |-- __init__.py
   ```

```
|   |-- settings.py
|   |-- urls.py
|   `-- wsgi.py
`-- manage.py
```

3. Exécutez votre site Django localement avec `manage.py runserver`.

```
(eb-virt) ~$ cd ebdjango
```

```
(eb-virt) ~/ebdjango$ python manage.py runserver
```

4. Dans un navigateur web, ouvrez `http://127.0.0.1:8000/` pour afficher le site.
5. Consultez le journal du serveur pour voir le résultat de votre demande. Pour arrêter le serveur web et revenir à votre environnement virtuel, appuyez sur Ctrl+C.

```
Django version 2.2, using settings 'ebdjango.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
[07/Sep/2018 20:14:09] "GET / HTTP/1.1" 200 16348
Ctrl+C
```

Configurer votre application Django pour Elastic Beanstalk

Maintenant que vous avez un site Django résidant sur votre ordinateur local, vous pouvez le configurer pour le déploiement avec Elastic Beanstalk.

Par défaut, Elastic Beanstalk recherche un fichier nommé `application.py` pour démarrer votre application. Comme il n'existe pas dans le projet Django que vous avez créé, vous devez apporter quelques modifications à l'environnement de votre application. Vous devez également définir des variables d'environnement afin que les modules de votre application puissent être chargés.

Pour configurer votre site pour Elastic Beanstalk

1. Activez votre environnement virtuel.

   Unix-based systems

   ```
   ~/ebdjango$ source ~/eb-virt/bin/activate
   ```

Windows

```
C:\Users\USERNAME\ebdjango>%HOMEPATH%\eb-virt\Scripts\activate
```

2. Exécutez `pip freeze`, puis enregistrez les données de sortie dans un fichier nommé `requirements.txt`.

```
(eb-virt) ~/ebdjango$ pip freeze > requirements.txt
```

Elastic `requirements.txt` Beanstalk permet de déterminer le package à installer EC2 sur les instances qui exécutent votre application.

3. Créez un répertoire nommé `.ebextensions`.

```
(eb-virt) ~/ebdjango$ mkdir .ebextensions
```

4. Dans le répertoire `.ebextensions`, ajoutez un fichier de configuration nommé `django.config` avec le texte suivant.

Example~/ebdjango/.ebextensions/django.config

```
option_settings:
  aws:elasticbeanstalk:container:python:
    WSGIPath: ebdjango.wsgi:application
```

Ce paramètre, `WSGIPath`, spécifie l'emplacement du script WSGI qu'Elastic Beanstalk utilise pour lancer votre application.

> **Note**
>
> Si vous utilisez une version de plateforme AMI Python Amazon Linux (antérieure à Amazon Linux 2), remplacez la valeur `WSGIPath` par `ebdjango/wsgi.py`. La valeur de l'exemple fonctionne avec le serveur WSGI Gunicorn, qui n'est pas pris en charge sur les versions de la plateforme AMI Amazon Linux.

5. Désactivez votre environnement virtuel avec la commande `deactivate`.

```
(eb-virt) ~/ebdjango$ deactivate
```

Réactivez votre environnement virtuel chaque fois que vous devez ajouter des packages à votre application ou exécuter votre application localement.

Déploiement de votre site avec l'interface de ligne de commande (CLI) EB

Vous avez ajouté tout ce dont vous avez besoin pour déployer votre application sur Elastic Beanstalk. Le répertoire de votre projet devrait maintenant se présenter comme suit.

```
~/ebdjango/
|-- .ebextensions
|   `-- django.config
|-- ebdjango
|   |-- __init__.py
|   |-- settings.py
|   |-- urls.py
|   `-- wsgi.py
|-- db.sqlite3
|-- manage.py
`-- requirements.txt
```

Ensuite, vous allez créer votre environnement d'applications et déployer votre application configurée avec Elastic Beanstalk.

Immédiatement après le déploiement, vous allez modifier la configuration de Django afin d'ajouter le nom de domaine attribué par Elastic Beanstalk à votre application au code `ALLOWED_HOSTS` de Django. Ensuite, vous redéploierez votre application. Il s'agit d'une exigence de sécurité de Django, conçue pour empêcher les attaques de l'en-tête HTTP `Host`. Pour plus d'informations, consultez la section Validation de l'en-tête Host.

Pour créer un environnement et déployer votre application Django

> **Note**
>
> Ce tutoriel utilise l'interface de ligne de commande (CLI) EB comme mécanisme de déploiement, mais vous pouvez également utiliser la console Elastic Beanstalk pour déployer un fichier ZIP où se trouve le contenu de votre projet.

1. Initialisez votre référentiel de la CLI EB avec la commande eb init :

```
~/ebdjango$ eb init -p python-3.7 django-tutorial
Application django-tutorial has been created.
```

Cette commande crée une application nommée `django-tutorial`. Elle configure également votre référentiel local pour créer des environnements avec la dernière version de la plateforme Python 3.7.

2. (Facultatif) Exécutez à eb init nouveau pour configurer une paire de clés par défaut afin de pouvoir utiliser SSH pour vous connecter à l' EC2 instance qui exécute votre application.

```
~/ebdjango$ eb init
Do you want to set up SSH for your instances?
(y/n): y
Select a keypair.
1) my-keypair
2) [ Create new KeyPair ]
```

Sélectionnez une paire de clés si vous en avez déjà une, ou suivez les invites pour en créer une. Si vous ne voyez pas l'invite ou que vous avez besoin de modifier vos paramètres ultérieurement, exécutez eb init -i.

3. Créez un environnement et déployez-y votre application avec eb create.

```
~/ebdjango$ eb create django-env
```

> **Note**
>
> Si un message d'erreur « rôle de service requis » s'affiche, exécutez `eb create` de manière interactive (sans spécifier de nom d'environnement) et l'interface de ligne de commande (CLI) EB créera le rôle pour vous.

Cette commande crée un environnement Elastic Beanstalk équilibré en charge nommé `django-env`. La création d'un environnement prend environ 5 minutes. Lorsqu'Elastic Beanstalk crée les ressources nécessaires pour exécuter votre application, il génère des messages d'information que l'interface de ligne de commande (CLI) EB relaie à votre terminal.

4. Lorsque le processus de création de l'environnement est terminé, recherchez le nom de domaine de votre nouvel environnement en exécutant eb status.

```
~/ebdjango$ eb status
Environment details for: django-env
  Application name: django-tutorial
  ...
  CNAME: eb-django-app-dev.elasticbeanstalk.com
  ...
```

Le nom de domaine de votre environnement est la valeur de la propriété `CNAME`.

5. Ouvrez le fichier `settings.py` dans le répertoire `ebdjango`. Recherchez le paramètre `ALLOWED_HOSTS`, puis ajoutez le nom de domaine de votre application que vous avez trouvé à l'étape précédente à la valeur de ce paramètre. Si vous ne trouvez pas ce paramètre dans le fichier, ajoutez-le sur une nouvelle ligne.

```
...
ALLOWED_HOSTS = ['eb-django-app-dev.elasticbeanstalk.com']
```

6. Enregistrez le fichier, puis déployez votre application en exécutant eb deploy. Lorsque vous exécutez eb deploy, l'interface de ligne de commande (CLI) EB crée un bundle avec le contenu de votre répertoire de projet et le déploie dans votre environnement.

```
~/ebdjango$ eb deploy
```

> **Note**
>
> Si vous utilisez Git avec votre projet, veuillez consulter Utilisation de l'interface de ligne de commande EB avec Git.

7. Lorsque le processus de mise à jour de l'environnement est terminé, ouvrez votre site web avec la commande eb open.

```
~/ebdjango$ eb open
```

Celle-ci ouvre une fenêtre de navigation en utilisant le nom de domaine créé pour votre application. Vous devez voir le même site web Django que celui que vous avez créé et testé localement.

Si vous ne voyez pas votre application en cours d'exécution ou si vous obtenez un message d'erreur, consultez Déploiements afin d'obtenir de l'aide concernant la façon de déterminer la cause de l'erreur.

Si vous voyez effectivement votre application en cours d'exécution, alors félicitations, vous avez déployé votre première application Django avec Elastic Beanstalk !

Mise à jour de votre application

Maintenant que vous disposez d'une application en cours d'exécution sur Elastic Beanstalk, vous pouvez mettre à jour et redéployer votre application ou sa configuration, et Elastic Beanstalk va effectuer les tâches de mise à jour de vos instances et de démarrage de la nouvelle version de votre application.

Pour cet exemple, nous activerons la console d'administration Django et configurerons quelques autres paramètres.

Modification des paramètres de votre site

Par défaut, votre site web Django utilise le fuseau horaire UTC pour afficher l'heure. Vous pouvez modifier cela en spécifiant un fuseau horaire dans `settings.py`.

Pour modifier le fuseau horaire de votre site

1. Modifiez le paramètre `TIME_ZONE` dans `settings.py`.

   Example~/ .py ebdjango/ebdjango/settings

   ```
   ...
   # Internationalization
   LANGUAGE_CODE = 'en-us'
   TIME_ZONE = 'US/Pacific'
   USE_I18N = True
   USE_L10N = True
   USE_TZ = True
   ```

   Pour obtenir une liste des fuseaux horaires, consultez cette page.
2. Déployez l'application dans votre environnement Elastic Beanstalk.

   ```
   ~/ebdjango/$ eb deploy
   ```

Création d'un administrateur de site

Vous pouvez créer un administrateur de site pour votre application Django afin d'accéder à la console d'administration directement à partir du site web. Les informations de connexion administrateur sont stockées en toute sécurité dans l'image de base de données locale incluse dans le projet par défaut que Django génère.

Pour créer un administrateur de site

1. Initialisez la base de données locale de votre application Django.

```
(eb-virt) ~/ebdjango$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying sessions.0001_initial... OK
```

2. Exécutez `manage.py createsuperuser` pour créer un administrateur.

```
(eb-virt) ~/ebdjango$ python manage.py createsuperuser
Username: admin
Email address: me@mydomain.com
Password: ********
Password (again): ********
Superuser created successfully.
```

3. Pour dire à Django où stocker les fichiers statiques, définissez `STATIC_ROOT` dans `settings.py`.

Example~/ .py ebdjango/ebdjango/settings

```
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/2.2/howto/static-files/
STATIC_URL = '/static/'
STATIC_ROOT = 'static'
```

4. Exécutez `manage.py collectstatic` pour remplir le `static` répertoire avec des ressources statiques (JavaScript, CSS et images) pour le site d'administration.

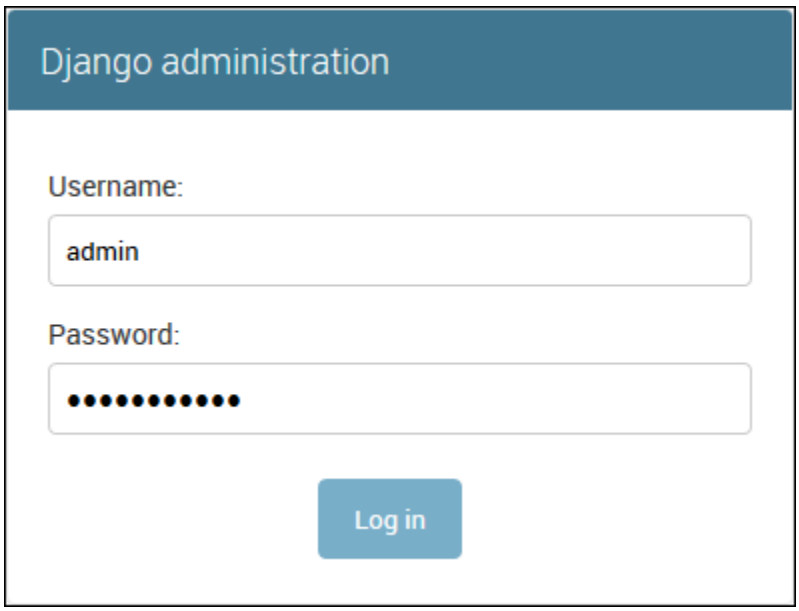
```
(eb-virt) ~/ebdjango$ python manage.py collectstatic
119 static files copied to ~/ebdjango/static
```

5. Déployez votre application.

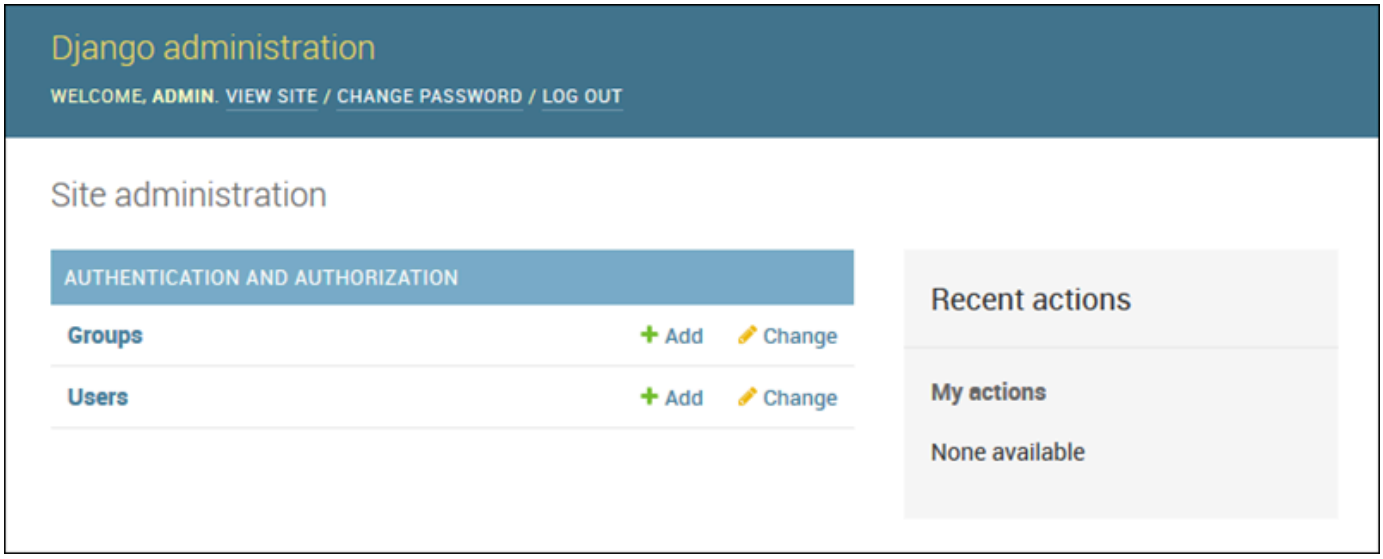
```
~/ebdjango$ eb deploy
```

6. Affichez la console d'administration en ouvrant le site dans votre navigateur, en ajoutant `/admin/` à l'URL du site, comme suit.

```
http://djang-env.p33kq46sfh.us-west-2.elasticbeanstalk.com/admin/
```



7. Connectez-vous avec le nom utilisateur et le mot de passe que vous avez configurés à l'étape 2.



Vous pouvez utiliser une procédure similaire de local updating/testing suivie deeb deploy. Elastic Beanstalk prend en charge les tâches de mise à jour de vos serveurs en direct, afin que vous puissiez vous concentrer sur le développement d'applications au lieu de l'administration du serveur !

Ajout d'un fichier de configuration de migration de base de données

Vous pouvez ajouter des commandes à votre script `.ebextensions`, qui sont exécutées lors de la mise à jour de votre site. Cela vous permet de générer automatiquement des migrations de base de données.

Pour ajouter une étape de migration lorsque votre application est déployée

1. Créez un fichier de configuration nommé `db-migrate.config` avec le contenu suivant.

   Example~/ebdjango/.ebextensions/db-migrate.config

   ```
   container_commands:
     01_migrate:
       command: "source /var/app/venv/*/bin/activate && python3 manage.py migrate"
       leader_only: true
   option_settings:
     aws:elasticbeanstalk:application:environment:
       DJANGO_SETTINGS_MODULE: ebdjango.settings
   ```

   Ce fichier de configuration active l'environnement virtuel du serveur et exécute la commande `manage.py migrate` lors du processus de déploiement, avant le démarrage de votre

application. Étant donné que cette commande est exécutée avant le démarrage de l'application, vous devez également configurer la variable d'environnement `DJANGO_SETTINGS_MODULE` explicitement (généralement, `wsgi.py` s'occupe de cette tâche pour vous pendant le démarrage). La spécification de `leader_only: true` dans la commande garantit qu'elle est exécutée une seule fois lors d'un déploiement sur plusieurs instances.

2. Déployez votre application.

```
~/ebdjango$ eb deploy
```

Nettoyage

Pour économiser des heures d'instance et d'autres AWS ressources entre les sessions de développement, mettez fin à votre environnement Elastic Beanstalk avec. eb terminate

```
~/ebdjango$ eb terminate django-env
```

Cette commande met fin à l'environnement et à toutes les AWS ressources qui s'y exécutent. Cependant, elle ne supprime pas l'application. Vous pouvez donc toujours créer d'autres environnements avec la même configuration en exécutant à nouveau eb create.

Si vous en avez terminé avec l'exemple d'application, vous pouvez également supprimer le dossier du projet et l'environnement virtuel.

```
~$ rm -rf ~/eb-virt
~$ rm -rf ~/ebdjango
```

Étapes suivantes

Pour plus d'informations sur Django, y compris un didacticiel approfondi, consultez la documentation officielle.

Si vous souhaitez essayer un autre cadre web Python, veuillez consulter Déploiement d'une application Flask sur Elastic Beanstalk.

Ajouter une instance de base de données Amazon RDS à votre environnement Python Elastic Beanstalk

Cette rubrique fournit des instructions pour créer un Amazon RDS à l'aide de la console Elastic Beanstalk. Vous pouvez utiliser une instance de base de données Amazon Relational Database Service (Amazon RDS) pour stocker les données collectées et modifiées par votre application. La base de données peut être liée à votre environnement et gérée par Elastic Beanstalk, ou elle peut être créée comme déliée et gérée en externe par un autre service. Dans ces instructions, la base de données est couplée à votre environnement et gérée par Elastic Beanstalk. Pour plus d'informations sur l'intégration d'un Amazon RDS avec Elastic Beanstalk, consultez Ajout d'une base de données à votre environnement Elastic Beanstalk.

Sections

- Ajout d'une instance de base de données à votre environnement
- Téléchargement d'un pilote
- Connexion à une base de données

Ajout d'une instance de base de données à votre environnement

Pour ajouter une instance DB à votre environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

L'ajout d'une instance DB prend environ 10 minutes. Une fois la mise à jour de l'environnement terminée, le nom d'hôte de l'instance DB et les autres informations de connexion sont disponibles dans votre application, via les propriétés d'environnement suivantes :

Nom de la propriété	Description	Valeur de la propriété
`RDS_HOSTNAME`	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
`RDS_PORT`	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
`RDS_DB_NAME`	Nom de la base de données, **ebdb**.	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
`RDS_USERNAME`	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
`RDS_PASSWORD`	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

Pour plus d'informations sur la configuration d'une instance de base de données liée à un environnement Elastic Beanstalk, consultez Ajout d'une base de données à votre environnement Elastic Beanstalk.

Téléchargement d'un pilote

Ajoutez le pilote de base de données au fichier Requirements de votre projet.

Example requirements.txt – Django avec MySQL

```
Django==2.2
mysqlclient==2.0.3
```

Packages de pilotes courants pour Python

- MySQL – `mysqlclient`
- PostgreSQL – `psycopg2`
- Oracle – `cx_Oracle`
- SQL Server – `adodbapi`

Pour plus d'informations, consultez Python DatabaseInterfaces et Django 2.2 : bases de données prises en charge.

Connexion à une base de données

Elastic Beanstalk fournit des informations de connexion pour les instances de base de données attachées dans les propriétés de l'environnement. Utilisez `os.environ['VARIABLE']` pour lire les propriétés et configurer une connexion de base de données.

Example Fichier de paramètres Django – Dictionnaire DATABASES

```
import os

if 'RDS_HOSTNAME' in os.environ:
    DATABASES = {
        'default': {
            'ENGINE': 'django.db.backends.mysql',
            'NAME': os.environ['RDS_DB_NAME'],
            'USER': os.environ['RDS_USERNAME'],
            'PASSWORD': os.environ['RDS_PASSWORD'],
            'HOST': os.environ['RDS_HOSTNAME'],
            'PORT': os.environ['RDS_PORT'],
        }
    }
```

Outils et ressources Python

Il existe plusieurs endroits auxquels vous pouvez accéder pour obtenir une aide supplémentaire lors du développement de vos applications Python :

Ressource	Description
AWS SDK pour Python (Boto3) sur GitHub	Installez Boto3 à partir de. GitHub
AWS SDK pour Python (Boto3) page d'accueil	La AWS SDK pour Python (Boto3) page d'accueil.
Centre pour développeurs Python	Guichet unique pour l'exemple de code, la documenta tion, les outils et les ressources supplémentaires.

Déploiement d'applications Ruby avec Elastic Beanstalk

Ce chapitre fournit des instructions pour configurer et déployer votre application Web Ruby sur AWS Elastic Beanstalk. Elastic Beanstalk facilite le déploiement, la gestion et le dimensionnement de vos applications Web Ruby à l'aide d'Amazon Web Services.

Vous pouvez déployer votre application en quelques minutes à l'aide de l'interface de ligne de commande Elastic Beanstalk (EB CLI) ou de la console Elastic Beanstalk. Après avoir déployé votre application Elastic Beanstalk, vous pouvez continuer à utiliser l'EB CLI pour gérer votre application et votre environnement, ou vous pouvez utiliser la console Elastic Beanstalk, ou le. AWS CLI APIs

Ce chapitre fournit également des step-by-step instructions pour déployer un exemple d'application sur Elastic Beanstalk à l'aide de l'EB CLI, puis pour mettre à jour l'application pour utiliser les frameworks d'applications Web Rails et Sinatra.

Les rubriques de ce chapitre supposent que vous avez une certaine connaissance des environnements Elastic Beanstalk. Si vous n'avez jamais utilisé Elastic Beanstalk, essayez le tutoriel de mise en route pour acquérir les bases.

Rubriques

- Configuration de votre environnement de développement Ruby pour Elastic Beanstalk
- Utilisation de la plateforme Elastic Beanstalk Ruby
- Déploiement d'une application rails sur Elastic Beanstalk
- Déploiement d'une application sinatra sur Elastic Beanstalk
- Ajouter une instance de base de données Amazon RDS à votre environnement Ruby Elastic Beanstalk

Configuration de votre environnement de développement Ruby pour Elastic Beanstalk

Ce chapitre fournit des instructions pour configurer un environnement de développement Ruby afin de tester votre application localement avant de la déployer AWS Elastic Beanstalk. Il fait également référence à des sites Web qui fournissent des instructions d'installation pour des outils utiles.

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

Sections

- Installation de Ruby
- Installation du AWS SDK pour Ruby
- Installation d'un IDE ou d'un éditeur de texte

Installation de Ruby

Installez GCC si vous n'avez pas de compilateur C. Sur Ubuntu, utilisez `apt`.

```
~$ sudo apt install gcc
```

Sur Amazon Linux, utilisez `yum`.

```
~$ sudo yum install gcc
```

Installez RVM pour gérer les installations de langage Ruby sur votre machine. Utilisez les commandes sur rvm.io pour obtenir les clés du projet et exécuter le script d'installation.

```
~$ gpg2 --recv-keys key1 key2
~$ curl -sSL https://get.rvm.io | bash -s stable
```

Ce script installe RVM dans un dossier nommé `.rvm` dans votre répertoire utilisateur et modifie votre profil shell pour charger un script de configuration chaque fois que vous ouvrez un nouveau terminal. Chargez le script manuellement pour commencer.

```
~$ source ~/.rvm/scripts/rvm
```

Utilisez `rvm get head` pour obtenir la version la plus récente.

```
~$ rvm get head
```

Affichez les versions disponibles de Ruby.

```
~$ rvm list known
```

Consultez Ruby dans le document Plateformes AWS Elastic Beanstalk pour trouver la dernière version de Ruby disponible sur une plateforme Elastic Beanstalk. Installez cette version.

```
~$ rvm install 3.2
```

Testez votre installation Ruby.

```
~$ ruby --version
```

Installation du AWS SDK pour Ruby

Si vous devez gérer AWS des ressources depuis votre application, installez le AWS SDK pour Ruby. Par exemple, avec le kit SDK pour Ruby, vous pouvez utiliser Amazon DynamoDB (DynamoDB) pour stocker les informations utilisateur et session sans créer de base de données relationnelle.

Installez le kit SDK pour Ruby et ses dépendances avec la commande `gem`.

```
$ gem install aws-sdk
```

Consultez la page d'accueil AWS SDK pour Ruby pour plus d'informations et pour connaître les instructions d'installation.

Installation d'un IDE ou d'un éditeur de texte

Les environnements de développement intégrés (IDEs) fournissent un large éventail de fonctionnalités qui facilitent le développement d'applications. Si vous n'avez pas utilisé d'IDE pour le développement de Ruby, essayez Aptana RubyMine et voyez celui qui vous convient le mieux.

- Installez Aptana

- RubyMine

> **Note**
>
> Un IDE peut ajouter des fichiers dans votre dossier de projet que vous pouvez ne pas souhaiter engager sur le contrôle de code source. Pour empêcher la validation de ces fichiers de contrôle de code source, utilisez `.gitignore` ou l'équivalent de votre outil de contrôle de source.

Si vous souhaitez simplement commencer le codage et que vous n'avez pas besoin de toutes les fonctionnalités d'un IDE, pensez à installer Sublime Text.

Utilisation de la plateforme Elastic Beanstalk Ruby

Cette rubrique explique comment configurer, créer et exécuter vos applications Ruby sur Elastic Beanstalk.

AWS Elastic Beanstalk prend en charge un certain nombre de branches de plate-forme pour différentes versions du langage de programmation Ruby. Voir Ruby dans le document AWS Elastic Beanstalk Platforms pour une liste complète.

L'application Web Ruby peut s'exécuter derrière un serveur proxy NGINX sous un serveur d'applications Puma. Si vous l'utilisez RubyGems, vous pouvez en inclure un `Gemfile`dans votre bundle source pour installer des packages lors du déploiement.

Configuration du serveur de l'application

Elastic Beanstalk installe le serveur d'applications Puma en fonction de la branche de plateforme Ruby que vous choisissez lors de la création de votre environnement. Pour plus d'informations sur les composants fournis avec les versions de la plateforme Ruby, consultez Plateformes prises en charge dans le guide AWS Elastic Beanstalk Plateformes.

Vous pouvez configurer votre application avec le serveur Puma que vous avez fourni. Cela permet d'utiliser une version de Puma autre que celle préinstallée avec la branche de plateforme Ruby. Vous pouvez également configurer votre application pour utiliser un autre serveur d'applications, tel que Passenger. Pour ce faire, vous devez inclure et personnaliser un `Gemfile` dans votre déploiement. Vous devez également configurer un `Procfile` pour démarrer le serveur d'applications. Pour plus d'informations, consultez Configuration du processus de l'application avec un Profil.

Autres options de configuration

Elastic Beanstalk fournit des options de configuration que vous pouvez utiliser pour personnaliser le logiciel qui s'exécute sur les instances Amazon Elastic Compute Cloud (Amazon EC2) dans votre environnement Elastic Beanstalk. Vous pouvez configurer des variables d'environnement nécessaires pour votre application, activer la rotation des journaux sur Amazon S3 et mapper des dossiers dans la source de votre application contenant des fichiers statiques vers des chemins desservis par le serveur proxy. La plateforme prédéfinit également certaines variables d'environnement courantes associées à Rails et Rack pour faciliter la découverte et l'utilisation.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour modifier la configuration d'un environnement en cours d'exécution. Pour éviter de perdre la configuration de votre environnement en le résiliant, vous pouvez utiliser des configurations enregistrées pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des fichiers de configuration. Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

Les paramètres appliqués dans la console Elastic Beanstalk remplacent les mêmes paramètres des fichiers de configuration, s'ils existent. Cela vous permet d'utiliser les paramètres par défaut dans les fichiers de configuration et de les remplacer par des paramètres spécifiques à l'environnement dans la console. Pour plus d'informations sur la priorité et les autres méthodes de modification des paramètres, consultez Options de configuration.

Pour plus d'informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, consultez the section called “Extension des plateformes Linux”.

Configuration de votre environnement Ruby

Vous pouvez utiliser la console Elastic Beanstalk pour activer la rotation de journal sur Amazon S3 et configurer les variables que votre application peut lire depuis l'environnement.

Pour accéder aux paramètres de configuration du logiciel pour votre environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.

Options du journal

La section Log Options (Options du journal) a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Enable log file rotation to Amazon S3 (Permettre la rotation du fichier journal sur Amazon S3) – Indique si les fichiers journaux des instances Amazon EC2 de votre application doivent être copiés dans le compartiment Amazon S3 associé à votre application.

Fichiers statiques

Pour améliorer les performances, la section des Fichiers statiques vous permet de configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Pour chaque répertoire, vous définissez le chemin virtuel sur le mappage de répertoires. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Pour en savoir plus sur la configuration des fichiers statiques à l'aide des fichiers de configuration ou de la console Elastic Beanstalk, consultez the section called “Fichiers statiques”.

Par défaut, le serveur proxy d'un environnement Ruby est configuré pour servir des fichiers statiques comme suit :

- Les fichiers du dossier `public` sont servis à partir du chemin `/public` et de la racine du domaine (chemin `/`).
- Les fichiers du sous-dossier `public/assets` sont servis à partir du chemin `/assets`.

Les exemples suivants illustrent le fonctionnement de la configuration par défaut :

- Si votre code source d'application contient un fichier nommé `logo.png` dans un dossier nommé `public`, le serveur proxy le sert aux utilisateurs dans *`subdomain`*`.elasticbeanstalk.com/public/logo.png` et *`subdomain`*`.elasticbeanstalk.com/logo.png`.
- Si votre code source d'application contient un fichier nommé `logo.png` dans un dossier nommé `assets`dans le dossier `public`, le serveur proxy le sert aux utilisateurs dans *`subdomain`*`.elasticbeanstalk.com/assets/logo.png`.

Vous pouvez configurer d'autres mappages pour les fichiers statiques. Pour plus d'informations, consultez Espaces de noms de configuration Ruby plus loin dans cette rubrique.

> **Note**
>
> Pour les versions de plate-forme antérieures à Ruby 2.7 AL2 version 3.3.7, la configuration par défaut du serveur proxy nginx Elastic Beanstalk ne prend pas en charge le service de fichiers statiques à partir de la racine du domaine (). *`subdomain`*`.elasticbeanstalk.com/` Cette version de plateforme a été publiée le 21 octobre 2021. Pour plus d'informations, consultez la section .Nouvelles versions de plateforme - Ruby dans les Notes de mise à jour AWS Elastic Beanstalk .

Propriétés de l'environnement

La section Propriétés de l'environnement vous permet de spécifier des paramètres de configuration de l'environnement sur les instances Amazon EC2 exécutant votre application. Les propriétés de l'environnement sont passées en tant que paires clé-valeur à l'application.

La plateforme Ruby définit les propriétés suivantes pour la configuration de l'environnement :

- BUNDLE_WITHOUT – Une liste séparée par des deux-points de groupes à ignorer lors de l'installation de dépendances à partir d'un Gemfile.
- BUNDLER_DEPLOYMENT_MODE – Définissez la valeur sur `true` (valeur par défaut) pour installer les dépendances en mode de déploiement à l'aide de Bundler. Définissez la valeur sur `false` pour exécuter `bundle install` en mode de développement.

> **Note**
>
> Cette propriété d'environnement n'est pas définie sur les branches de la plateforme Amazon Linux AMI Ruby (auparavant Amazon Linux 2).

- RAILS_SKIP_ASSET_COMPILATION – Définissez ce paramètre sur `true` pour ignorer l'exécution de `rake assets:precompile` durant le déploiement.
- RAILS_SKIP_MIGRATIONS – Définissez ce paramètre sur `true` pour ignorer l'exécution de `rake db:migrate` durant le déploiement.
- RACK_ENV – Spécifiez l'étape de l'environnement pour Rack. Par exemple, `development`, `production` ou `test`.

Dans l'environnement Ruby en cours d'exécution dans Elastic Beanstalk, les variables d'environnement sont accessibles à l'aide de l'objet `ENV`. Par exemple, vous pouvez lire une propriété nommée `API_ENDPOINT` sur une variable avec le code suivant :

```
endpoint = ENV['API_ENDPOINT']
```

Pour plus d'informations, consultez Variables d'environnement et autres paramètres du logiciel.

Espaces de noms de configuration Ruby

Vous pouvez utiliser un fichier de configuration pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être spécifiques à la plate-forme ou s'appliquer à toutes les plateformes du service Elastic Beanstalk dans son ensemble. Les options de configuration sont organisées en espaces de noms.

Vous pouvez utiliser l'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles` pour configurer le proxy d'environnement afin de traiter les fichiers statiques. Vous définissez les mappages des chemins virtuels vers les répertoires d'applications.

La plate-forme Ruby ne définit aucun espace de noms spécifique à la plate-forme. Au lieu de cela, elle définit les propriétés d'environnement pour les options Rails et Rack communes.

Le fichier de configuration suivant spécifie une option de fichiers statiques qui mappe un répertoire nommé `staticimages` au chemin d'accès `/images`, définit chacune des propriétés

d'environnement définies par la plate-forme et définit une propriété d'environnement supplémentaire nommée `LOGGING`.

Example.ebextensions/ruby-settings.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:staticfiles:
    /images: staticimages
  aws:elasticbeanstalk:application:environment:
    BUNDLE_WITHOUT: test
    BUNDLER_DEPLOYMENT_MODE: true
    RACK_ENV: development
    RAILS_SKIP_ASSET_COMPILATION: true
    RAILS_SKIP_MIGRATIONS: true
    LOGGING: debug
```

> **Note**
>
> La propriété d' environnement `BUNDLER_DEPLOYMENT_MODE` et l'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles` ne sont pas définis sur les branches de la plateforme Amazon Linux AMI Ruby (anciennement Amazon Linux 2).

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d' AWS CLI. Pour plus d’informations, consultez Options de configuration.

Installation de packages avec un Gemfile sur Elastic Beanstalk

RubyGems Pour installer les packages requis par votre application, incluez un `Gemfile` fichier à la racine de la source de votre projet.

Example Gemfile

```
source "https://rubygems.org"
gem 'sinatra'
gem 'json'
gem 'rack-parser'
```

Lorsqu'un fichier `Gemfile` est présent, Elastic Beanstalk exécute `bundle install` pour installer les dépendances. Pour plus d'informations, veuillez consulter les pages Gemfile et Bundle sur le site Web Bundler.io.

> **Note**
>
> Vous pouvez utiliser une version différente de Puma en plus de la version par défaut préinstallée sur la plateforme Ruby. Pour ce faire, insérez une entrée dans `Gemfile` qui sert à indiquer la version. Vous pouvez également spécifier un autre serveur d'applications, tel que Passenger, en utilisant un `Gemfile` personnalisé.
> Pour ces deux cas, vous devez également configurer un `Procfile` pour démarrer le serveur d'applications.
> Pour plus d'informations, consultez Configuration du processus de l'application avec un Profil.

Configuration du processus de candidature avec un Procfile sur Elastic Beanstalk.

Pour spécifier la commande qui démarre votre application Ruby, incluez un fichier appelé `Procfile` à la racine de votre solution groupée source.

> **Note**
>
> Elastic Beanstalk ne prend pas en charge cette fonctionnalité sur les branches de plateforme Amazon Linux AMI Ruby (auparavant Amazon Linux 2). Les branches de plateforme dont les noms contiennent Puma ou Passenger, quelle que soit leur version Ruby, précèdent Amazon Linux 2 et ne prennent pas en charge la fonctionnalité `Procfile`.

Pour plus de détails sur l'écriture et l'utilisation d'un`Procfile`, voirBuildfile et Procfile.

Lorsque vous ne fournissez pas de`Procfile`, Elastic Beanstalk génère une valeur par défaut. `Procfile` Si vous `Gemfile` incluez Puma, Elastic Beanstalk part du principe que vous souhaitez utiliser la version de Puma que vous avez fournie et génère la version par défaut suivante. `Procfile`

```
web: bundle exec puma -C /opt/elasticbeanstalk/config/private/pumaconf.rb
```

Si vous `Gemfile` n'incluez pas Puma, Elastic Beanstalk suppose que vous utilisez le serveur d'applications Puma préinstallé et génère la valeur par défaut suivante. `Procfile` Sur les branches de la plateforme Amazon Linux 2 Ruby, Elastic Beanstalk génère toujours la valeur par défaut `Procfile` suivante si vous ne fournissez pas de. `Procfile`

```
web: puma -C /opt/elasticbeanstalk/config/private/pumaconf.rb
```

> **Note**
>
> Le 10 octobre 2024, les dernières branches de la plateforme Ruby Amazon Linux 2 ont été supprimées. Toutes les branches de la plateforme Ruby actuellement prises en charge sont basées sur Amazon Linux 2023. Pour plus d'informations sur la migration, consultezMigration depuis Amazon Linux 2 vers Amazon Linux 2023.

Si vous souhaitez utiliser le serveur d'applications Passenger, utilisez les exemples de fichiers suivants pour configurer votre environnement Ruby pour installer et utiliser Passenger.

1. Utilisez ce fichier d'exemple pour installer Passenger.

   Example Gemfile

   ```
   source 'https://rubygems.org'
   gem 'passenger'
   ```

2. Utilisez cet exemple de fichier pour demander à Elastic Beanstalk de démarrer Passenger.

   Example Procfile

   ```
   web: bundle exec passenger start /var/app/current --socket /var/run/puma/my_app.sock
   ```

> **Note**
>
> Aucun changement ne doit être apporté à la configuration du serveur proxy nginx pour utiliser Passenger. Pour utiliser d'autres serveurs d'applications, vous devrez peut-être personnaliser la configuration nginx pour transférer correctement les demandes à votre application.

Déploiement d'une application rails sur Elastic Beanstalk

Rails est un framework open source model-view-controller (MVC) pour Ruby. Ce didacticiel vous explique le processus de génération d'une application Rails et de son déploiement AWS Elastic Beanstalk dans un environnement.

Sections

- Conditions préalables
- Connaissances de base sur Elastic Beanstalk
- Lancer un environnement Elastic Beanstalk
- Installation de Rails et génération d'un site web
- Configuration des paramètres Rails
- Déploiement de votre application
- Nettoyage
- Étapes suivantes

Conditions préalables

Connaissances de base sur Elastic Beanstalk

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans Découvrez comment démarrer avec Elastic Beanstalk pour lancer votre premier environnement Elastic Beanstalk.

Ligne de commande

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

Dépendances Rails

L'infrastructure Rails 6.1.4.1 a les dépendances suivantes. Veillez à ce qu'elles soient toutes installées.

- Ruby 2.5.0 ou version ultérieure – Pour obtenir les instructions d'installation, consultez Configuration de votre environnement de développement Ruby pour Elastic Beanstalk.

  Dans ce tutoriel, nous utilisons Ruby 3.0.2 et la version correspondante de la plateforme Elastic Beanstalk.
- Node.js – Pour obtenir des instructions d'installation, consultez Installation de Node.js via le gestionnaire de package.
- Yarn – Pour obtenir des instructions d'installation, veuillez consulter Installation sur le site web Yarn.

Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Choisissez la plateforme Ruby et acceptez les paramètres par défaut et l'exemple de code.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk à l'aide de ce lien préconfiguré : console.aws.amazon.com/elasticbeanstalk/home#/newApplication? Nom de l'application = Tutoriels et type d'environnement = LoadBalanced
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Create app (Créer une application).

La création d'un environnement prend environ 5 minutes et crée les ressources suivantes :

- EC2 instance — Une machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications Web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Équilibreur de charge – Équilibreur de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité d'équilibrage de charge : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibreur de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et qui sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

> **Sécurité du domaine**
>
> Pour renforcer la sécurité de vos applications Elastic Beanstalk, le domaine elasticbeanstalk.com est enregistré dans la liste des suffixes publics (PSL).
> Si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Elastic Beanstalk, nous vous recommandons d'utiliser des cookies `__Host-`

avec un préfixe pour une sécurité accrue. Cette pratique protège votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

> **Note**
>
> Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon S3.

Installation de Rails et génération d'un site web

Installez Rails et ses dépendances avec la commande `gem`.

```
~$ gem install rails
Fetching: concurrent-ruby-1.1.9.gem
Successfully installed concurrent-ruby-1.1.9
Fetching: rack-2.2.3.gem
Successfully installed rack-2.2.3
...
```

Testez votre installation Rails.

```
~$ rails --version
Rails 6.1.4.1
```

Utilisez `rails new` avec le nom de l'application pour créer un projet Rails.

```
~$ rails new ~/eb-rails
```

Rails crée un répertoire avec le nom spécifié, génère tous les fichiers nécessaires pour exécuter un exemple de projet localement et exécute ensuite Bundler pour installer toutes les dépendances (Gems) définies dans le Gemfile du projet.

> **Note**
>
> Ce processus installe la dernière version de Puma pour le projet. Cette version peut être différente de celle fournie par Elastic Beanstalk sur la version de la plateforme Ruby de votre environnement. Pour consulter les versions Puma fournies par Elastic Beanstalk, veuillez consulter la section Historique de la plateforme Ruby dans le Guide des plateformes AWS Elastic Beanstalk . Pour plus d'informations sur la version la plus récente de Puma, consultez le site Web Puma.io. En cas de décalage entre les deux versions de Puma, utilisez l'une des options suivantes :
>
> - Utiliser la version Puma installée par la commande `rails new` précédente. Dans ce cas, vous devez ajouter un `Procfile` pour que la plateforme utilise la version du serveur Puma fournie par vos soins. Pour de plus amples informations, veuillez consulter Configuration du processus de candidature avec un Procfile sur Elastic Beanstalk..
> - Mettre à jour la version Puma pour qu'elle soit cohérente avec celle qui est préinstallée sur la version de la plateforme Ruby de votre environnement. Pour ce faire, modifiez la version Puma dans le Gemfile situé à la racine du répertoire source de votre projet. Ensuite, exécutez `bundle update`. Pour plus d'informations, veuillez consulter la page bundle update sur le site Web Bundler.io.

Testez votre installation Rails en exécutant le projet par défaut localement.

```
~$ cd eb-rails
~/eb-rails$ rails server
=> Booting Puma
=> Rails 6.1.4.1 application starting in development
=> Run `bin/rails server --help` for more startup options
Puma starting in single mode...
* Puma version: 5.5.2 (ruby 3.0.2-p107) ("Zawgyi")
*  Min threads: 5
*  Max threads: 5
*  Environment: development
*          PID: 77857
* Listening on http://127.0.0.1:3000
* Listening on http://[::1]:3000
Use Ctrl-C to stop
...
```

Ouvrez `http://localhost:3000` dans un navigateur web pour afficher le projet par défaut en action.

Cette page est uniquement visible en mode développement. Ajoutez du contenu à la première page de l'application afin de prendre en charge le déploiement de production sur Elastic Beanstalk. Utilisez `rails generate` pour créer un contrôleur, un itinéraire et une vue pour votre page d'accueil.

```
~/eb-rails$ rails generate controller WelcomePage welcome
      create  app/controllers/welcome_page_controller.rb
       route  get 'welcome_page/welcome'
      invoke  erb
      create    app/views/welcome_page
      create    app/views/welcome_page/welcome.html.erb
      invoke  test_unit
      create    test/controllers/welcome_page_controller_test.rb
      invoke  helper
      create    app/helpers/welcome_page_helper.rb
      invoke    test_unit
      invoke  assets
```

```
      invoke    coffee
      create      app/assets/javascripts/welcome_page.coffee
      invoke    scss
      create      app/assets/stylesheets/welcome_page.scss.
```

Cela vous permet de disposer de tout ce dont vous avez besoin pour accéder à la page à l'adresse `/welcome_page/welcome`. Avant de publier les modifications, toutefois, modifiez le contenu dans la vue et ajoutez un itinéraire pour que cette page s'affiche au niveau supérieur du site.

Utilisez un éditeur de texte pour modifier le contenu dans `app/views/welcome_page/welcome.html.erb`. Pour cet exemple, vous utiliserez `cat` pour remplacer simplement le contenu du fichier existant.

Example app/views/welcome_page/welcome.html.erb

```
<h1>Welcome!</h1>
<p>This is the front page of my first Rails application on Elastic Beanstalk.</p>
```

Enfin, ajoutez l'itinéraire suivant à `config/routes.rb` :

Example config/routes.rb

```
Rails.application.routes.draw do
  get 'welcome_page/welcome'
  root 'welcome_page#welcome'
```

Cela indique à Rails d'acheminer les demandes à la racine du site web à la méthode de bienvenue du contrôleur de la page d'accueil, qui affiche le contenu dans la vue d'accueil (`welcome.html.erb`).

Pour que Elastic Beanstalk puisse déployer correctement l'application sur la plateforme Ruby, nous devons mettre à jour `Gemfile.lock`. Certaines dépendances de `Gemfile.lock` peuvent être spécifiques à la plateforme. Par conséquent, nous devons ajouter **`platform ruby`** à `Gemfile.lock` afin que toutes les dépendances requises soient installées avec le déploiement.

Example

```
~/eb-rails$ bundle lock --add-platform ruby
Fetching gem metadata from https://rubygems.org/.....
Resolving dependencies...
```

```
Writing lockfile to /Users/janedoe/EBDPT/RubyApps/eb-rails-doc-app/Gemfile.lock
```

Configuration des paramètres Rails

Utilisez la console Elastic Beanstalk pour configurer Rails avec les propriétés d'environnement. Définissez la propriété d'environnement `SECRET_KEY_BASE` sous la forme d'une chaîne de 256 caractères alphanumériques au plus.

Rails utilise cette propriété pour créer des clés. Par conséquent, vous devez la garder secrète et ne pas la stocker dans le contrôle de code source en texte brut. Au lieu de cela, vous la fournissez au code Rails dans votre environnement via une propriété d'environnement.

Pour configurer les variables d'environnement dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Faites défiler la page jusqu'à Variables d'environnement d'exécution.
6. Sélectionnez Ajouter une variable d'environnement.
7. Pour Source, sélectionnez Texte brut.

> **Note**
>
> Les valeurs Secrets Manager et SSM Parameter Store de la liste déroulante permettent de configurer les variables d'environnement en tant que secrets pour stocker des données sensibles, telles que les informations d'identification et les clés d'API. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic AWS Secrets Manager AWS Systems Manager Beanstalk avec et Parameter Store.

8. Entrez le nom de la variable d'environnement et les paires de valeurs de la variable d'environnement.
9. Si vous devez ajouter d'autres variables, répétez les étapes 6 à 8.
10. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Maintenant, vous êtes prêt à déployer le site dans votre environnement.

Déploiement de votre application

Créez un bundle source contenant les fichiers créés par Rails. La commande suivante permet de créer une solution groupée source nommée `rails-default.zip`.

```
~/eb-rails$ zip ../rails-default.zip -r * .[^.]*
```

Téléchargez le bundle source sur Elastic Beanstalk pour déployer Rails dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

Nettoyage

Une fois que vous avez fini d'utiliser le code de démonstration, vous pouvez mettre fin à votre environnement. Elastic Beanstalk supprime toutes les ressources AWS associées, telles que les instances EC2 Amazon, les instances de base de données, les équilibreurs de charge, lesgroupes de sécurité et les alarmes.

La suppression de ressources n'entraîne pas la suppression de l'application Elastic Beanstalk. Vous pouvez donc créer de nouveaux environnements pour votre application à tout moment.

Pour mettre fin à votre environnement Elastic Beanstalk depuis la console

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Terminate Environment (Résilier l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Étapes suivantes

Pour plus d'informations sur Rails, visitez rubyonrails.org.

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le charger manuellement sur la console Elastic Beanstalk. L'interface de ligne de commande Elastic Beanstalk (EB CLI easy-to-use) fournit des commandes pour créer, configurer et déployer des applications dans les environnements Elastic Beanstalk à partir de la ligne de commande.

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez configurer un nom de domaine personnalisé pour votre environnement et activer HTTPS pour des connexions sécurisées.

Déploiement d'une application sinatra sur Elastic Beanstalk

Cette procédure détaillée explique comment déployer une application web simple Sinatra sur AWS Elastic Beanstalk.

Conditions préalables

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans Découvrez comment démarrer avec Elastic Beanstalk pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

Sinatra 2.1.0 nécessite Ruby 2.3.0 ou version postérieure. Dans ce tutoriel, nous utilisons Ruby 3.0.2 et la version correspondante de la plateforme Elastic Beanstalk. Installez Ruby en suivant les instructions de la section Configuration de votre environnement de développement Ruby pour Elastic Beanstalk.

Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Choisissez la plateforme Ruby et acceptez les paramètres par défaut et l'exemple de code.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk à l'aide de ce lien préconfiguré : console.aws.amazon.com/elasticbeanstalk/home#/newApplication? Nom de l'application = Tutoriels et type d'environnement = LoadBalanced
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Create app (Créer une application).

La création d'un environnement prend environ 5 minutes et crée les ressources suivantes :

- EC2 instance — Une machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications Web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.
- Groupe de sécurité d'instance : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource permet au trafic HTTP provenant de l'équilibreur de charge

d'atteindre l' EC2 instance qui exécute votre application Web. Par défaut, le trafic n'est pas autorisé sur les autres ports.

- Équilibreur de charge – Équilibreur de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité d'équilibrage de charge : groupe EC2 de sécurité Amazon configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibreur de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et qui sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *`subdomain`*.*`region`*.elasticbeanstalk.com.

> **Sécurité du domaine**
>
> Pour renforcer la sécurité de vos applications Elastic Beanstalk, le domaine elasticbeanstalk.com est enregistré dans la liste des suffixes publics (PSL).
> Si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Elastic Beanstalk, nous vous recommandons d'utiliser des cookies `__Host-` avec un préfixe pour une sécurité accrue. Cette pratique protège votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

> **Note**
>
> Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon S3.

Écriture d'un site web Sinatra

Pour créer et déployer une application Sinatra

1. Créez un fichier de configuration nommé config.ru avec le contenu suivant.

   Example config.ru

   ```
   require './helloworld'
   run Sinatra::Application
   ```

2. Créez un fichier de code Ruby nommé helloworld.rb avec le contenu suivant.

   Example helloworld.rb

   ```
   require 'sinatra'
   get '/' do
     "Hello World!"
   end
   ```

3. Créez un Gemfile avec le contenu suivant.

   Example Gemfile

   ```
   source 'https://rubygems.org'
   gem 'sinatra'
   gem 'puma'
   ```

4. Exécutez l'installation du bundle pour générer le fichier `Gemfile.lock`

   Example

   ```
   ~/eb-sinatra$ bundle install
   Fetching gem metadata from https://rubygems.org/....
   ```

```
Resolving dependencies...
Using bundler 2.2.22
Using rack 2.2.3
...
```

5. Pour que Elastic Beanstalk puisse déployer correctement l'application sur la plateforme Ruby, nous devons mettre à jour `Gemfile.lock`. Certaines dépendances de `Gemfile.lock` peuvent être spécifiques à la plateforme. Par conséquent, nous devons ajouter **`platform ruby`** à `Gemfile.lock` afin que toutes les dépendances requises soient installées avec le déploiement.

   Example

   ```
   ~/eb-sinatra$ bundle lock --add-platform ruby
   Fetching gem metadata from https://rubygems.org/....
   Resolving dependencies...
   Writing lockfile to /Users/janedoe/EBDPT/RubyApps/eb-sinatra/Gemfile.lock
   ```

6. Créez un Procfile avec le contenu suivant.

   Example Procfile

   ```
   web: bundle exec puma -C /opt/elasticbeanstalk/config/private/pumaconf.rb
   ```

Déploiement de votre application

Créez un bundle source contenant vos fichiers source. La commande suivante permet de créer une solution groupée source nommée `sinatra-default.zip`.

```
~/eb-sinatra$ zip ../sinatra-default.zip -r * .[^.]*
```

Téléchargez le bundle source sur Elastic Beanstalk pour déployer Sinatra dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

Nettoyage

Une fois que vous avez fini d'utiliser le code de démonstration, vous pouvez mettre fin à votre environnement. Elastic Beanstalk supprime toutes les ressources AWS associées, telles que les instances EC2 Amazon, les instances de base de données, les équilibreurs de charge, lesgroupes de sécurité et les alarmes.

La suppression de ressources n'entraîne pas la suppression de l'application Elastic Beanstalk. Vous pouvez donc créer de nouveaux environnements pour votre application à tout moment.

Pour mettre fin à votre environnement Elastic Beanstalk depuis la console

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Terminate Environment (Résilier l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Étapes suivantes

Pour plus d'informations sur Sinatra, consultez sinatrarb.com.

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le charger manuellement sur la console Elastic Beanstalk. L'interface de ligne de commande Elastic Beanstalk (EB CLI easy-to-use) fournit des commandes pour créer, configurer et déployer des applications dans les environnements Elastic Beanstalk à partir de la ligne de commande.

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez configurer un nom de domaine personnalisé pour votre environnement et activer HTTPS pour des connexions sécurisées.

Ajouter une instance de base de données Amazon RDS à votre environnement Ruby Elastic Beanstalk

Cette rubrique fournit des instructions pour créer un Amazon RDS à l'aide de la console Elastic Beanstalk. Vous pouvez utiliser une instance de base de données Amazon Relational Database Service (Amazon RDS) pour stocker les données collectées et modifiées par votre application. La base de données peut être liée à votre environnement et gérée par Elastic Beanstalk, ou elle peut être créée comme déliée et gérée en externe par un autre service. Dans ces instructions, la base de données est couplée à votre environnement et gérée par Elastic Beanstalk. Pour plus d'informations sur l'intégration d'un Amazon RDS avec Elastic Beanstalk, consultez Ajout d'une base de données à votre environnement Elastic Beanstalk.

Sections

- Ajout d'une instance de base de données à votre environnement
- Téléchargement d'une carte
- Connexion à une base de données

Ajout d'une instance de base de données à votre environnement

Pour ajouter une instance DB à votre environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

L'ajout d'une instance DB prend environ 10 minutes. Une fois la mise à jour de l'environnement terminée, le nom d'hôte de l'instance DB et les autres informations de connexion sont disponibles dans votre application, via les propriétés d'environnement suivantes :

Nom de la propriété	Description	Valeur de la propriété
`RDS_HOSTNAME`	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
`RDS_PORT`	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
`RDS_DB_NAME`	Nom de la base de données, **`ebdb`**.	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
`RDS_USERNAME`	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
`RDS_PASSWORD`	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

Pour plus d'informations sur la configuration d'une instance de base de données liée à un environnement Elastic Beanstalk, consultez Ajout d'une base de données à votre environnement Elastic Beanstalk.

Téléchargement d'une carte

Ajoutez la carte de base de données au Gemfile de votre projet.

Example Gemfile – Rails avec MySQL

```
source 'https://rubygems.org'
gem 'puma'
gem 'rails', '~> 6.1.4', '>= 6.1.4.1'
gem 'mysql2'
```

Fichiers Gem de carte courants pour Ruby

- MySQL – mysql2
- PostgreSQL – pg
- Oracle – activerecord-oracle_enhanced-adapter
- SQL Server – activerecord-sqlserver-adapter

Connexion à une base de données

Elastic Beanstalk fournit des informations de connexion pour les instances de base de données attachées dans les propriétés de l'environnement. Utilisez `ENV['VARIABLE']` pour lire les propriétés et configurer une connexion de base de données.

Example config/database.yml – Ruby sur une configuration de base de données Rails (MySQL)

```
production:
  adapter: mysql2
  encoding: utf8
  database: <%= ENV['RDS_DB_NAME'] %>
  username: <%= ENV['RDS_USERNAME'] %>
  password: <%= ENV['RDS_PASSWORD'] %>
  host: <%= ENV['RDS_HOSTNAME'] %>
  port: <%= ENV['RDS_PORT'] %>
```

Déploiement avec des conteneurs Docker sur Elastic Beanstalk

Ce chapitre explique comment utiliser Elastic Beanstalk pour déployer des applications Web à partir de conteneurs Docker. Les conteneurs Docker sont indépendants, et incluent toutes les informations de configuration et les logiciels dont votre application web a besoin pour fonctionner. Les conteneurs Docker vous permettent de définir votre propre environnement d'exécution. Vous pouvez également choisir votre propre langage de programmation et les dépendances des applications, telles que les gestionnaires de packages ou les outils, qui ne sont généralement pas pris en charge par les autres plateformes Elastic Beanstalk.

Suivez les étapes ci-dessous QuickStart pour Docker pour créer une application Docker « Hello World » et déployez-la dans un environnement Elastic Beanstalk à l'aide de l'EB CLI.

Rubriques

- Branches de la plateforme Elastic Beanstalk Docker
- Utilisation de la branche de la plateforme Elastic Beanstalk Docker
- Utilisation de la branche de plateforme Docker gérée par ECS dans Elastic Beanstalk
- Authentification avec des référentiels d'images
- Configuration des environnements Elastic Beanstalk Docker
- Plateformes traditionnelles

Branches de la plateforme Elastic Beanstalk Docker

La plateforme Docker pour Elastic Beanstalk prend en charge les branches de plateforme suivantes :

Docker exécutant Amazon Linux 2 et Docker exécutant AL2023

Elastic Beanstalk déploie des conteneurs Docker et du code source sur des instances EC2, puis les gère. Ces branches de plateforme offrent une prise en charge de plusieurs conteneurs. Vous pouvez utiliser l'outil Docker Compose pour simplifier la configuration, les tests et le déploiement de vos applications. Pour de plus amples informations sur cette branche de plateforme, veuillez consulter the section called “Branche de plateforme Docker”.

ECS s'exécute sur Amazon Linux 2 et ECS s'exécute sur AL2023

Nous proposons cette branche aux clients qui ont besoin d'un chemin de migration AL2023 AL2 vers/depuis la branche de plate-forme retirée fonctionnant sur plusieurs conteneurs Docker (AMI Amazon Linux). Les dernières branches de plateforme prennent en charge toutes les fonctionnalités de la branche de plateforme retirée. Aucune modification du code source n'est requise. Pour de plus amples informations, veuillez consulter Migration de votre application Elastic Beanstalk depuis AL1 Docker multi-conteneurs géré par ECS vers ECS sur Amazon Linux 2023. Si vous ne disposez pas d'un environnement Elastic Beanstalk exécuté sur une branche de plate-forme basée sur ECS, nous vous recommandons d'utiliser la branche de plate-forme, Docker Running on 64 bits. AL2023 Ceci offre une approche plus simple et nécessite moins de ressources.

Pour obtenir la liste des versions des composants logiciels associées à chacune de ces branches de plate-forme, consultez Docker dans le document AWS Elastic Beanstalk Platforms.

Branches de plateforme retirées exécutées sur l'AMI Amazon Linux (AL1)

Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Développez chaque section qui suit pour en savoir plus sur chaque branche de plateforme retirée et son chemin de migration vers la dernière branche de plateforme s'exécutant sur Amazon Linux 2 ou Amazon Linux 2023 (recommandé).

Docker (AMI Amazon Linux)

Cette branche de plateforme peut déployer une image Docker, décrite dans un Dockerfile ou une définition `Dockerrun.aws.json` v1. Cette branche de plateforme n'exécute qu'un seul conteneur pour chaque instance. Ses branches de plateforme suivantes, Docker fonctionnant sur 64 bits AL2023 et Docker fonctionnant sur 64 bits, Amazon Linux 2 prennent en charge plusieurs conteneurs Docker par instance.

Nous vous recommandons de créer vos environnements avec la branche de plate-forme Docker la plus récente et prise en charge fonctionnant sur 64 AL2023 bits. Vous pouvez ensuite migrer votre application vers l'environnement nouvellement créé. Pour plus d'informations sur la création de ces environnements, consultez the section called “Branche de plateforme Docker”. Pour en savoir plus sur la migration, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Docker multiconteneurs (AMI Amazon Linux)

Cette branche de plateforme utilise Amazon ECS pour coordonner un déploiement de plusieurs conteneurs Docker vers un cluster Amazon ECS dans un environnement Elastic Beanstalk. Si vous

utilisez actuellement cette branche de plateforme retirée, nous vous recommandons de migrer vers la dernière version de branche de plateforme, ECS s'exécutant sur Amazon Linux 2023. La dernière branche de plateforme prend en charge toutes les fonctions de cette branche de plateforme abandonnée. Aucune modification du code source n'est requise. Pour de plus amples informations, veuillez consulter Migration de votre application Elastic Beanstalk depuis AL1 Docker multi-conteneurs géré par ECS vers ECS sur Amazon Linux 2023.

Conteneurs Docker préconfigurés

Outre les plateformes Docker mentionnées précédemment, il existe également la branche de GlassFish plate-forme Docker préconfigurée qui s'exécute sur le système d'exploitation Amazon Linux AMI (). AL1

Cette branche de plate-forme a été remplacée par les branches de plate-forme Docker fonctionnant sur 64 bits AL2023 et Docker fonctionnant sur Amazon Linux 2 64 bits. Pour plus d'informations, consultez Déploiement d'une GlassFish application sur la plateforme Docker.

Utilisation de la branche de la plateforme Elastic Beanstalk Docker

Cette section explique comment préparer votre image Docker pour le lancement avec Docker en cours d'exécution ou avec Docker, l'une des branches de la plateforme Elastic Beanstalk. AL2 AL2023

Suivez les étapes ci-dessous QuickStart pour Docker pour créer une application Docker « Hello World » et déployez-la dans un environnement Elastic Beanstalk à l'aide de l'EB CLI.

Rubriques

- QuickStart: Déployer une application Docker sur Elastic Beanstalk
- QuickStart: Déployer une application Docker Compose sur Elastic Beanstalk
- Préparation de votre image Docker pour le déploiement sur Elastic Beanstalk

QuickStart: Déployer une application Docker sur Elastic Beanstalk

Ce QuickStart didacticiel explique le processus de création d'une application Docker et de son déploiement dans un AWS Elastic Beanstalk environnement.

⚠ Non destiné à une utilisation en production

Les exemples sont destinés uniquement à des fins de démonstration. N'utilisez pas d'exemples d'applications en production.

Sections

- Votre AWS compte
- Conditions préalables
- Étape 1 : créer une application et un conteneur Docker
- Étape 2 : Exécutez votre application localement
- Étape 3 : Déployez votre application Docker avec l'EB CLI
- Étape 4 : Exécutez votre application sur Elastic Beanstalk
- Étape 5 : nettoyer
- AWS ressources pour votre application
- Étapes suivantes
- Déployez avec la console Elastic Beanstalk

Votre AWS compte

Si vous n'êtes pas encore AWS client, vous devez créer un AWS compte. L'inscription vous permet d'accéder à Elastic Beanstalk AWS et aux autres services dont vous avez besoin.

Si vous avez déjà un AWS compte, vous pouvez passer àConditions préalables.

Créez un AWS compte

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez l'https://portal.aws.amazon.com/billing/inscription.
2. Suivez les instructions en ligne.

Dans le cadre de la procédure d'inscription, vous recevrez un appel téléphonique ou un SMS et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l'utilisateur racine a accès à l'ensemble des Services AWS et des ressources de ce compte. La meilleure pratique de sécurité consiste à attribuer un accès administratif à un utilisateur, et à utiliser uniquement l'utilisateur racine pour effectuer les tâches nécessitant un accès utilisateur racine.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. À tout moment, vous pouvez consulter l'activité actuelle de votre compte et le gérer en accédant à https://aws.amazon.com/et en choisissant Mon compte.

Création d'un utilisateur doté d'un accès administratif

Après vous être inscrit à un Compte AWS, sécurisez Utilisateur racine d'un compte AWS AWS IAM Identity Center, activez et créez un utilisateur administratif afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l'aide pour vous connecter en utilisant l'utilisateur racine, consultez Connexion en tant qu'utilisateur racine dans le Guide de l'utilisateur Connexion à AWS .
2. Activez l'authentification multifactorielle (MFA) pour votre utilisateur racine.

   Pour obtenir des instructions, voir Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d'un utilisateur doté d'un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d' AWS IAM Identity Center dans le Guide de l'utilisateur AWS IAM Identity Center .

2. Dans IAM Identity Center, octroyez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connexion en tant qu'utilisateur doté d'un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l'URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l'utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribution d'un accès à d'autres utilisateurs

1. Dans IAM Identity Center, créez un ensemble d'autorisations qui respecte la bonne pratique consistant à appliquer les autorisations de moindre privilège.

   Pour obtenir des instructions, consultez Création d'un ensemble d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .
2. Attribuez des utilisateurs à un groupe, puis attribuez un accès par authentification unique au groupe.

   Pour obtenir des instructions, consultez Ajout de groupes dans le Guide de l'utilisateur AWS IAM Identity Center .

Conditions préalables

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

INTERFACE DE LIGNE DE COMMANDE (CLI) EB

Ce tutoriel utilise également l'interface de ligne de commande Elastic Beanstalk (CLI EB). Pour de plus amples informations sur l'installation et la configuration de la CLI EB, veuillez consulter Installation de la CLI EB avec un script de configuration (recommandé) et Configuration de l'interface de ligne de commande EB.

Docker

Pour suivre ce didacticiel, vous aurez besoin d'une installation locale fonctionnelle de Docker. Pour de plus amples informations, veuillez consulter Get Docker (Obtenir Docker) sur le site web de documentation de Docker.

Vérifiez que le daemon Docker est en cours d'exécution en exécutant la commande suivante.

```
~$ docker info
```

Étape 1 : créer une application et un conteneur Docker

Dans cet exemple, nous créons une image Docker de l'exemple d'application Flask qui est également référencée dans. Déploiement d'une application Flask sur Elastic Beanstalk

L'application est composée de deux fichiers :

- `app.py`— le fichier Python qui contient le code qui sera exécuté dans le conteneur.
- `Dockerfile`— le Dockerfile pour créer votre image.

Placez les deux fichiers à la racine d'un répertoire.

```
~/eb-docker-flask/
|-- Dockerfile
|-- app.py
```

Ajoutez le contenu suivant à votre`Dockerfile`.

Example~/eb-docker-flask/Dockerfile

```
FROM public.ecr.aws/docker/library/python:3.12
COPY . /app
WORKDIR /app
RUN pip install Flask==3.1.1
EXPOSE 5000
CMD [ "python3", "-m" , "flask", "run", "--host=0.0.0.0"]
```

Ajoutez le contenu suivant à votre `app.py` fichier.

Example~/eb-docker-flask/app.py

```
from flask import Flask
app = Flask(__name__)
@app.route('/')
def hello_world():
    return 'Hello Elastic Beanstalk! This is a Docker application'
```

Utilisez la commande docker build pour créer votre image de conteneur localement, en l'étiquetant avec. `eb-docker-flask` Le point (`.`) à la fin de la commande indique que le chemin est un répertoire local.

```
~/eb-docker-flask$ docker build -t eb-docker-flask .
```

Étape 2 : Exécutez votre application localement

Exécutez votre conteneur avec la commande docker run. La commande imprimera l'ID du conteneur en cours d'exécution. L'-doption exécute docker en mode arrière-plan. L'-poption expose votre application sur le port 5000. Elastic Beanstalk achemine le trafic vers le port 5000 de la plateforme Docker par défaut.

```
~/eb-docker-flask$ docker run -dp 127.0.0.1:5000:5000 eb-docker-flask
```

Accédez à `http://127.0.0.1:5000/` dans votre navigateur. Vous devriez voir le texte « Hello Elastic Beanstalk ! Il s'agit d'une application Docker ».

Exécutez la commande docker kill pour terminer le conteneur.

```
~/eb-docker-flask$ docker kill container-id
```

Étape 3 : Déployez votre application Docker avec l'EB CLI

Exécutez les commandes suivantes pour créer un environnement Elastic Beanstalk pour cette application.

Pour créer un environnement et déployer votre application Docker

1. Initialisez votre référentiel de la CLI EB avec la commande eb init :

   ```
   ~/eb-docker-flask$ eb init -p docker docker-tutorial --region us-east-2
   Application docker-tutorial has been created.
   ```

   Cette commande crée une application nommée `docker-tutorial` et configure votre référentiel local pour créer des environnements avec la dernière version de la plateforme Docker.
2. (facultatif) Exécutez à nouveau la commande eb init pour configurer une paire de clés par défaut afin de pouvoir vous connecter à l'instance EC2 qui exécute votre application.

   ```
   ~/eb-docker-flask$ eb init
   Do you want to set up SSH for your instances?
   (y/n): y
   Select a keypair.
   1) my-keypair
   2) [ Create new KeyPair ]
   ```

   Sélectionnez une paire de clés si vous en avez déjà une, ou suivez les invites pour en créer une. Si vous ne voyez pas l'invite ou que vous avez besoin de modifier vos paramètres ultérieurement, exécutez eb init -i.
3. Créez un environnement et déployez-y votre application avec eb create. Elastic Beanstalk crée automatiquement un fichier zip pour votre application et le démarre sur le port 5000.

   ```
   ~/eb-docker-flask$ eb create docker-tutorial
   ```

   Il faut environ cinq minutes à Elastic Beanstalk pour créer votre environnement.

Étape 4 : Exécutez votre application sur Elastic Beanstalk

Lorsque le processus de création de votre environnement est terminé, ouvrez votre site Web aveceb open.

```
~/eb-docker-flask$ eb open
```

Félicitations ! Vous avez déployé une application Docker avec Elastic Beanstalk ! Celle-ci ouvre une fenêtre de navigation en utilisant le nom de domaine créé pour votre application.

Étape 5 : nettoyer

Vous pouvez mettre fin à votre environnement lorsque vous avez fini d'utiliser votre application. Elastic Beanstalk AWS met fin à toutes les ressources associées à votre environnement.

Pour mettre fin à votre environnement Elastic Beanstalk avec l'EB CLI, exécutez la commande suivante.

```
~/eb-docker-flask$ eb terminate
```

AWS ressources pour votre application

Vous venez de créer une application à instance unique. Il s'agit d'un exemple d'application simple avec une seule instance EC2. Il ne nécessite donc pas d'équilibrage de charge ni de dimensionnement automatique. Pour les applications à instance unique, Elastic Beanstalk crée les ressources suivantes : AWS

- Instance EC2 – Une machine virtuelle Amazon EC2 configurée pour exécuter des applications web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble distinct de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse qui traite le trafic web devant votre application web, lui transmet les demandes, traite les ressources statiques et génère des journaux d'accès et d'erreurs.
- Groupe de sécurité de l'instance – Un groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et sont déclenchées si la charge est trop élevée ou trop faible.

Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.

- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

Elastic Beanstalk gère toutes ces ressources. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

Étapes suivantes

Dès que vous disposez d'un environnement exécutant une application, vous pouvez à tout moment déployer une nouvelle version de l'application ou une application totalement différente. Le déploiement d'une nouvelle version d'application est très rapide, car il n'est pas nécessaire de mettre en service ni de redémarrer les instances EC2. Vous pouvez également explorer votre nouvel environnement à l'aide de la console Elastic Beanstalk. Pour connaître les étapes détaillées, consultez la section Explorez votre environnement dans le chapitre Mise en route de ce guide.

Une fois que vous avez déployé un ou deux exemples d'applications et que vous êtes prêt à commencer à développer et à exécuter des applications Docker localement, consultezPréparation de votre image Docker pour le déploiement sur Elastic Beanstalk.

Déployez avec la console Elastic Beanstalk

Vous pouvez également utiliser la console Elastic Beanstalk pour lancer l'exemple d'application. Pour connaître les étapes détaillées, voir Création d'un exemple d'application dans le chapitre Mise en route de ce guide.

QuickStart: Déployer une application Docker Compose sur Elastic Beanstalk

Ce QuickStart didacticiel explique le processus de création d'une application Docker Compose multi-conteneurs et de son déploiement dans un AWS Elastic Beanstalk environnement. Vous allez créer une application Web Flask avec un proxy inverse nginx pour montrer comment Docker Compose simplifie l'orchestration de plusieurs conteneurs.

⚠ Non destiné à une utilisation en production

Les exemples sont destinés uniquement à des fins de démonstration. N'utilisez pas d'exemples d'applications en production.

Sections

- Votre AWS compte
- Conditions préalables
- Étape 1 : créer une application Docker Compose
- Étape 2 : Exécutez votre application localement
- Étape 3 : Déployez votre application Docker Compose avec l'EB CLI
- Étape 4 : Testez votre application sur Elastic Beanstalk
- Étape 5 : nettoyer
- AWS ressources pour votre application
- Étapes suivantes
- Déployez avec la console Elastic Beanstalk

Votre AWS compte

Si vous n'êtes pas encore AWS client, vous devez créer un AWS compte. L'inscription vous permet d'accéder à Elastic Beanstalk AWS et aux autres services dont vous avez besoin.

Si vous avez déjà un AWS compte, vous pouvez passer àConditions préalables.

Créez un AWS compte

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez l'https://portal.aws.amazon.com/billing/inscription.
2. Suivez les instructions en ligne.

Dans le cadre de la procédure d'inscription, vous recevrez un appel téléphonique ou un SMS et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l'utilisateur racine a accès à l'ensemble des Services AWS et des ressources de ce compte. La meilleure pratique de sécurité consiste à attribuer un accès administratif à un utilisateur, et à utiliser uniquement l'utilisateur racine pour effectuer les tâches nécessitant un accès utilisateur racine.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. À tout moment, vous pouvez consulter l'activité actuelle de votre compte et le gérer en accédant à https://aws.amazon.com/et en choisissant Mon compte.

Création d'un utilisateur doté d'un accès administratif

Après vous être inscrit à un Compte AWS, sécurisez Utilisateur racine d'un compte AWS AWS IAM Identity Center, activez et créez un utilisateur administratif afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l'aide pour vous connecter en utilisant l'utilisateur racine, consultez Connexion en tant qu'utilisateur racine dans le Guide de l'utilisateur Connexion à AWS .
2. Activez l'authentification multifactorielle (MFA) pour votre utilisateur racine.

   Pour obtenir des instructions, voir Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d'un utilisateur doté d'un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d' AWS IAM Identity Center dans le Guide de l'utilisateur AWS IAM Identity Center .

2. Dans IAM Identity Center, octroyez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connexion en tant qu'utilisateur doté d'un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l'URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l'utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribution d'un accès à d'autres utilisateurs

1. Dans IAM Identity Center, créez un ensemble d'autorisations qui respecte la bonne pratique consistant à appliquer les autorisations de moindre privilège.

   Pour obtenir des instructions, consultez Création d'un ensemble d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .
2. Attribuez des utilisateurs à un groupe, puis attribuez un accès par authentification unique au groupe.

   Pour obtenir des instructions, consultez Ajout de groupes dans le Guide de l'utilisateur AWS IAM Identity Center .

Conditions préalables

Pour suivre les procédures décrites dans ce guide, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite ($) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows, vous pouvez installer le sous-système Windows pour Linux afin d'obtenir une version intégrée à Windows d'Ubuntu et de Bash.

INTERFACE DE LIGNE DE COMMANDE (CLI) EB

Ce tutoriel utilise également l'interface de ligne de commande Elastic Beanstalk (CLI EB). Pour de plus amples informations sur l'installation et la configuration de la CLI EB, veuillez consulter Installation de la CLI EB avec un script de configuration (recommandé) et Configuration de l'interface de ligne de commande EB.

Docker et Docker Compose

Pour suivre ce didacticiel, vous aurez besoin d'une installation locale fonctionnelle de Docker et Docker Compose. Pour plus d'informations, consultez Télécharger Docker et installer Docker Compose sur le site Web de documentation de Docker.

Vérifiez que Docker et Docker Compose sont installés et en cours d'exécution en exécutant les commandes suivantes.

```
~$ docker info
~$ docker compose version
```

Étape 1 : créer une application Docker Compose

Dans cet exemple, nous créons une application multi-conteneurs à l'aide de Docker Compose composée d'une application Web Flask et d'un proxy inverse nginx. Cela montre comment Docker Compose simplifie l'orchestration de plusieurs conteneurs qui fonctionnent ensemble.

L'application inclut une configuration de surveillance de l'état qui permet à Elastic Beanstalk de collecter des métriques d'application détaillées à partir de votre proxy nginx.

La structure de l'application est la suivante :

```
~/eb-docker-compose-flask/
|-- docker-compose.yml
|-- web/
|   |-- Dockerfile
|   |-- app.py
|   `-- requirements.txt
```

```
|-- proxy/
|   |-- Dockerfile
|   `-- nginx.conf
`-- .platform/
    `-- hooks/
        `-- postdeploy/
            `-- 01_setup_healthd_permissions.sh
```

Créez la structure du répertoire et ajoutez les fichiers suivants :

Créez d'abord le `docker-compose.yml` fichier principal qui définit les services et leurs relations.

Example~/eb-docker-compose-flask/docker-compose.yml

```
services:
  web:
    build: ./web
    expose:
      - "5000"

  nginx-proxy:
    build: ./proxy
    ports:
      - "80:80"
    volumes:
      - "/var/log/nginx:/var/log/nginx"
    depends_on:
      - web
```

Créez l'application Web Flask dans le `web` répertoire. Ajoutez le contenu suivant à votre `app.py` fichier.

Example~/eb-docker-compose-flask/web/app.py

```
from flask import Flask
app = Flask(__name__)
@app.route('/')
def hello_world():
    return 'Hello Elastic Beanstalk! This is a Docker Compose application'
```

Ajoutez le contenu suivant à votre service Web`Dockerfile`.

Example~/eb-docker-compose-flask/web/Dockerfile

```
FROM public.ecr.aws/docker/library/python:3.12
COPY . /app
WORKDIR /app
RUN pip install Flask==3.1.1
EXPOSE 5000
CMD [ "python3", "-m" , "flask", "run", "--host=0.0.0.0"]
```

Créez le proxy inverse nginx dans le répertoire. `proxy` Ajoutez le contenu suivant à votre `nginx.conf` fichier.

Cette configuration inclut une configuration de surveillance de l'état qui permet à Elastic Beanstalk de collecter des métriques détaillées sur les applications. Pour plus d'informations sur la personnalisation des formats des journaux de surveillance de l'état, consultezFormat de journal d'intégrité améliorée.

Example~/eb-docker-compose-flask/proxy/nginx.conf

```
events {
    worker_connections 1024;
}

http {
    include       /etc/nginx/mime.types;
    default_type  application/octet-stream;

    map $http_upgrade $connection_upgrade {
        default       "upgrade";
    }

    # Health monitoring log format for Elastic Beanstalk
    log_format healthd
 '$msec"$uri"$status"$request_time"$upstream_response_time"$http_x_forwarded_for';

    upstream flask_app {
        server web:5000;
    }

    server {
        listen 80 default_server;
        root /usr/share/nginx/html;
```

```
        # Standard access log
        access_log /var/log/nginx/access.log;

        # Health monitoring log for Elastic Beanstalk
        if ($time_iso8601 ~ "^(\d{4})-(\d{2})-(\d{2})T(\d{2})") {
            set $year $1;
            set $month $2;
            set $day $3;
            set $hour $4;
        }
        access_log /var/log/nginx/healthd/application.log.$year-$month-$day-$hour
 healthd;

        location / {
            proxy_pass http://flask_app;
            proxy_http_version    1.1;

            proxy_set_header    Connection            $connection_upgrade;
            proxy_set_header    Upgrade               $http_upgrade;
            proxy_set_header    Host                  $host;
            proxy_set_header    X-Real-IP             $remote_addr;
            proxy_set_header    X-Forwarded-For       $proxy_add_x_forwarded_for;
        }
    }
}
```

Ajoutez le contenu suivant à votre service proxy`Dockerfile`.

Example~/eb-docker-compose-flask/proxy/Dockerfile

```
FROM public.ecr.aws/nginx/nginx:alpine
COPY nginx.conf /etc/nginx/nginx.conf
EXPOSE 80
```

Enfin, créez un script de connexion à la plateforme pour configurer les répertoires de journaux et les autorisations nécessaires à la surveillance de l'état de santé. Les hooks de plateforme vous permettent d'exécuter des scripts personnalisés pendant le processus de déploiement. Pour plus d'informations sur les hooks de plateforme, consultezHooks de plateforme.

Example~/eb-docker-compose-flask/.platform/hooks/postdeploy/01_setup_healthd_permissions.sh

```
#!/bin/bash
```

```
set -ex

NGINX_CONTAINER=$(docker ps --filter "name=nginx-proxy" -q)

if [ -z "$NGINX_CONTAINER" ]; then
    echo "Error: No nginx-proxy container found running"
    exit 1
fi

NGINX_UID=$(docker exec ${NGINX_CONTAINER} id -u nginx)
NGINX_GID=$(docker exec ${NGINX_CONTAINER} id -g nginx)

mkdir -p /var/log/nginx/healthd
chown -R ${NGINX_UID}:${NGINX_GID} /var/log/nginx
```

Étape 2 : Exécutez votre application localement

Utilisez la commande docker compose up pour créer et exécuter votre application multi-conteneurs localement. Docker Compose créera les deux images de conteneur et démarrera les services définis dans votre `docker-compose.yml` fichier.

```
~/eb-docker-compose-flask$ docker compose up --build
```

--buildCette option garantit que Docker Compose crée les images du conteneur avant de démarrer les services. Vous devriez voir une sortie indiquant le démarrage du service Web et du service nginx-proxy.

Accédez à `http://localhost` dans votre navigateur. Vous devriez voir le texte « Hello Elastic Beanstalk ! Il s'agit d'une application Docker Compose ». Le proxy nginx reçoit votre demande sur le port 80 et la transmet à l'application Flask exécutée sur le port 5000.

Lorsque vous avez terminé le test, arrêtez l'application Ctrl+C en appuyant sur le terminal ou exécutez la commande suivante dans un autre terminal :

```
~/eb-docker-compose-flask$ docker compose down
```

Étape 3 : Déployez votre application Docker Compose avec l'EB CLI

Exécutez les commandes suivantes pour créer un environnement Elastic Beanstalk pour cette application.

Pour créer un environnement et déployer votre application Docker Compose

1. Initialisez votre référentiel de la CLI EB avec la commande eb init :

   ```
   ~/eb-docker-compose-flask$ eb init -p docker docker-compose-tutorial --region us-east-2
   Application docker-compose-tutorial has been created.
   ```

   Cette commande crée une application nommée `docker-compose-tutorial` et configure votre référentiel local pour créer des environnements avec la dernière version de la plateforme Docker.

2. (facultatif) Exécutez à nouveau la commande eb init pour configurer une paire de clés par défaut afin de pouvoir vous connecter à l'instance EC2 qui exécute votre application.

   ```
   ~/eb-docker-compose-flask$ eb init
   Do you want to set up SSH for your instances?
   (y/n): y
   Select a keypair.
   1) my-keypair
   2) [ Create new KeyPair ]
   ```

   Sélectionnez une paire de clés si vous en avez déjà une, ou suivez les invites pour en créer une. Si vous ne voyez pas l'invite ou que vous avez besoin de modifier vos paramètres ultérieurement, exécutez eb init -i.

3. Créez un environnement et déployez-y votre application avec eb create. Elastic Beanstalk `docker-compose.yml` détecte automatiquement votre fichier et déploie votre application multi-conteneurs.

   ```
   ~/eb-docker-compose-flask$ eb create docker-compose-tutorial
   ```

   Il faut environ cinq minutes à Elastic Beanstalk pour créer votre environnement et déployer votre application multi-conteneurs.

Étape 4 : Testez votre application sur Elastic Beanstalk

Lorsque le processus de création de votre environnement est terminé, ouvrez votre site Web aveceb open.

```
~/eb-docker-compose-flask$ eb open
```

Excellent ! Vous avez déployé une application Docker Compose multi-conteneurs avec Elastic Beanstalk ! Celle-ci ouvre une fenêtre de navigation en utilisant le nom de domaine créé pour votre application. Vous devriez voir le message de votre application Flask, diffusé via le proxy inverse nginx.

Étape 5 : nettoyer

Vous pouvez mettre fin à votre environnement lorsque vous avez fini d'utiliser votre application. Elastic Beanstalk AWS met fin à toutes les ressources associées à votre environnement.

Pour mettre fin à votre environnement Elastic Beanstalk avec l'EB CLI, exécutez la commande suivante.

```
~/eb-docker-compose-flask$ eb terminate
```

AWS ressources pour votre application

Vous venez de créer une seule instance d'application exécutant plusieurs conteneurs. Il s'agit d'un exemple d'application simple avec une seule instance EC2. Il ne nécessite donc pas d'équilibrage de charge ni de dimensionnement automatique. Pour les applications à instance unique, Elastic Beanstalk crée les ressources suivantes : AWS

- Instance EC2 – Une machine virtuelle Amazon EC2 configurée pour exécuter des applications web sur la plateforme de votre choix.

  Chaque plateforme exécute un ensemble distinct de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse qui traite le trafic web devant votre application web, lui transmet les demandes, traite les ressources statiques et génère des journaux d'accès et d'erreurs.
- Groupe de sécurité de l'instance – Un groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.

- CloudWatch Alarmes Amazon : deux CloudWatch alarmes qui surveillent la charge sur les instances de votre environnement et sont déclenchées si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- CloudFormation stack — Elastic CloudFormation Beanstalk utilise pour lancer les ressources de votre environnement et propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la console CloudFormation.
- Nom de domaine : nom de domaine qui permet d'accéder à votre application Web dans le formulaire *subdomain*.*region*.elasticbeanstalk.com.

Elastic Beanstalk gère toutes ces ressources. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient. Votre application Docker Compose exécute plusieurs conteneurs sur une seule instance EC2, Elastic Beanstalk gérant automatiquement l'orchestration.

Étapes suivantes

Dès que vous disposez d'un environnement exécutant une application, vous pouvez à tout moment déployer une nouvelle version de l'application ou une application totalement différente. Le déploiement d'une nouvelle version d'application est très rapide, car il n'est pas nécessaire de mettre en service ni de redémarrer les instances EC2. Vous pouvez également explorer votre nouvel environnement à l'aide de la console Elastic Beanstalk. Pour connaître les étapes détaillées, consultez la section Explorez votre environnement dans le chapitre Mise en route de ce guide.

Une fois que vous avez déployé un ou deux exemples d'applications et que vous êtes prêt à commencer à développer et à exécuter des applications Docker Compose localement, consultezPréparation de votre image Docker pour le déploiement sur Elastic Beanstalk.

Déployez avec la console Elastic Beanstalk

Vous pouvez également utiliser la console Elastic Beanstalk pour lancer une application Docker Compose. Créez un fichier ZIP contenant votre `docker-compose.yml` fichier ainsi que tous les répertoires et fichiers associés, puis chargez-le lors de la création d'une nouvelle application. Pour connaître les étapes détaillées, voir Création d'un exemple d'application dans le chapitre Mise en route de ce guide.

Préparation de votre image Docker pour le déploiement sur Elastic Beanstalk

Cette section explique comment préparer votre image Docker pour le déploiement sur Elastic Beanstalk avec Docker en cours d'exécution ou une branche de plateforme. AL2 AL2023 Les fichiers de configuration dont vous aurez besoin varient selon que vos images sont locales, distantes et si vous utilisez Docker Compose.

> **Note**
>
> Pour un exemple de procédure qui lance un environnement Docker, consultez la QuickStart pour Docker rubrique.

Rubriques

- Gérer vos images avec Docker Compose dans Elastic Beanstalk
- Gérer des images sans Docker Compose dans Elastic Beanstalk
- Création d'images personnalisées avec un Dockerfile

Gérer vos images avec Docker Compose dans Elastic Beanstalk

Vous pouvez choisir d'utiliser Docker Compose pour gérer différents services dans un seul fichier YAML. Pour en savoir plus sur Docker Compose, voir Pourquoi utiliser Compose ? sur le site Web de Docker.

- Créez un `docker-compose.yml`. Ce fichier est obligatoire si vous utilisez Docker Compose pour gérer votre application avec Elastic Beanstalk. Si tous vos déploiements proviennent d'images stockées dans des référentiels publics, aucun autre fichier de configuration n'est requis. Si les images source de votre déploiement se trouvent dans un référentiel privé, vous devrez effectuer une configuration supplémentaire. Pour plus d'informations, consultez la section Utilisation d'images provenant d'un dépôt privé. Pour de plus amples informations sur le fichier `docker-compose.yml`, veuillez consulter Compose file reference sur le site web Docker.
- L'option `Dockerfile` est facultative. Créez-en une si vous avez besoin d'Elastic Beanstalk pour créer et exécuter une image personnalisée locale. Pour de plus amples informations sur le fichier `Dockerfile`, veuillez consulter Dockerfile reference sur le site web Docker.

- Il se peut que vous deviez créer un `.zip` fichier. Si vous utilisez uniquement un `Dockerfile` fichier pour déployer votre application, il n'est pas nécessaire d'en créer un. Si vous utilisez des fichiers de configuration supplémentaires, le fichier .zip doit inclure le `docker-compose.yml` fichier`Dockerfile`, vos fichiers d'application et toutes les dépendances des fichiers d'application. Les `Dockerfile` et `docker-compose.yml` doivent se trouver à la racine, ou au niveau supérieur, de l'archive .zip. Si vous utilisez la CLI EB pour déployer votre application, elle crée un fichier .zip pour vous.

Pour en savoir plus sur Docker Compose et savoir comment l'installer, veuillez consulter les sites Docker Overview of Docker Compose et Install Docker Compose.

Gérer des images sans Docker Compose dans Elastic Beanstalk

Si vous n'utilisez pas Docker Compose pour gérer vos images Docker, vous devez configurer un `Dockerrun.aws.json` fichier`Dockerfile`, ou les deux.

- Créez une `Dockerfile` pour qu'Elastic Beanstalk crée et exécute une image personnalisée localement.
- Créez un fichier `Dockerrun.aws.json v1` afin de déployer une image Docker à partir d'un référentiel hébergé dans Elastic Beanstalk.
- Il se peut que vous deviez créer un `.zip` fichier. Si vous n'utilisez que l'un des deux fichiers, le `Dockerfile` ou le`Dockerrun.aws.json`, vous n'avez pas besoin de créer un fichier .zip. Si vous utilisez les deux fichiers, vous avez besoin d'un fichier .zip. Le fichier .zip doit inclure à la fois le `Dockerfile` et le`Dockerrun.aws.json`, ainsi que le fichier contenant vos fichiers d'application, ainsi que toutes les dépendances des fichiers d'application. Si vous utilisez l'interface de ligne de commande (CLI) EB pour déployer votre application, elle crée un fichier `.zip` automatiquement.

`Dockerrun.aws.json`fichier de configuration v1

Un fichier `Dockerrun.aws.json` décrit comment déployer une image Docker distante en tant qu'application Elastic Beanstalk. Ce fichier JSON est spécifique à Elastic Beanstalk. Si votre application s'exécute sur une image qui est disponible dans un référentiel hébergé, vous pouvez spécifier l'image dans un fichier `Dockerrun.aws.json v1` et omettre le `Dockerfile`.

> **Versions `Dockerrun.aws.json`**
>
> Le paramètre `AWSEBDockerrunVersion` indique la version du fichier `Dockerrun.aws.json`.
>
> - Le Docker AL2 et AL2023 les plateformes utilisent les versions suivantes du fichier.
>   - `Dockerrun.aws.json v3`— environnements utilisant Docker Compose.
>   - `Dockerrun.aws.json v1`— environnements qui n'utilisent pas Docker Compose.
> - Le `Dockerrun.aws.json v2` fichier est AL2023 utilisé par ECS exécuté sur Amazon Linux 2 et ECS exécuté sur Amazon Linux 2. La plate-forme retirée ECS, l'AMI Multicontainer Docker Amazon Linux (AL1), utilisait également cette même version.

Dockerrun.aws.json v1

Les clés et valeurs valides pour le fichier `Dockerrun.aws.json v1` incluent les opérations suivantes.

AWSEBDockerrunVersion

(Obligatoire) Spécifiez le numéro de version 1 si vous n'utilisez pas Docker Compose pour gérer votre image.

Authentification

(Obligatoire uniquement pour les référentiels privés) Indique l'objet Amazon S3 qui stocke le fichier `.dockercfg`.

Voir Authentification avec des référentiels d'images la section Utilisation d'images provenant d'un dépôt privé plus loin dans ce chapitre.

Image

Spécifie l'image de base Docker sur un référentiel Docker existant, à partir de laquelle vous créez un conteneur Docker. Spécifiez la valeur de la clé Name au format *`<organization>/<image name>`* pour les images sur Docker Hub ou *`<site>/<organization name>/<image name>`* pour d'autres sites.

Lorsque vous spécifiez une image dans le fichier `Dockerrun.aws.json`, chaque instance de votre environnement Elastic Beanstalk exécute `docker pull` pour exécuter l'image. Vous

pouvez également inclure la clé Update. La valeur par défaut est `true` et demande à Elastic Beanstalk de vérifier le référentiel, d'extraire les mises à jour de l'image et de remplacer toutes les images mises en cache.

Lorsque vous utilisez un fichier `Dockerfile`, ne spécifiez pas la clé Image dans le fichier `Dockerrun.aws.json`. Elastic Beanstalk crée et utilise toujours l'image décrite dans le fichier `Dockerfile`, lorsqu'il existe.

Ports

(Obligatoire si vous spécifiez la clé Image) Répertorie les ports à exposer sur le conteneur Docker. Elastic Beanstalk ContainerPortutilise cette valeur pour connecter le conteneur Docker au proxy inverse exécuté sur l'hôte.

Vous pouvez spécifier plusieurs ports de conteneur, mais Elastic Beanstalk utilise uniquement le premier port. Il utilise ce port pour connecter votre conteneur au proxy inverse de l'hôte et acheminer les demandes depuis le réseau Internet public. Si vous utilisez un`Dockerfile`, la première ContainerPortvaleur doit correspondre à la première entrée `Dockerfile` de la liste EXPOSE.

Vous pouvez éventuellement spécifier une liste de ports dans HostPort. HostPortles entrées spécifient les ports hôtes auxquels ContainerPortles valeurs sont mappées. Si vous ne spécifiez aucune HostPortvaleur, la valeur par défaut est la même ContainerPort.

```
{
  "Image": {
    "Name": "image-name"
  },
  "Ports": [
    {
      "ContainerPort": 8080,
      "HostPort": 8000
    }
  ]
}
```

Volumes

Mappez les volumes d'une instance EC2 à votre conteneur Docker. Spécifiez un ou plusieurs groupes de volumes à mapper.

```
{
  "Volumes": [
    {
      "HostDirectory": "/path/inside/host",
      "ContainerDirectory": "/path/inside/container"
    }
  ]
...
```

Journalisation

Spécifiez le répertoire du conteneur dans lequel votre application écrit les journaux. Elastic Beanstalk télécharge tous les journaux de ce répertoire sur Amazon S3 lorsque vous demandez des journaux de processus ou de groupe. Si vous effectuez une rotation des journaux dans un dossier nommé `rotated` au sein de ce répertoire, vous pouvez également configurer Elastic Beanstalk afin de charger les journaux concernés dans Amazon S3 pour un stockage permanent. Pour plus d'informations, consultez Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk.

Commande

Spécifiez une commande à exécuter dans le conteneur. Si vous spécifiez un point d'entrée, la valeur Command (Commande) est ajoutée comme argument dans Entrypoint (Point d'entrée). Pour plus d'informations, consultezCMD dans la documentation Docker.

Entrypoint

Spécifiez une commande par défaut à exécuter lorsque le conteneur démarre. Pour plus d'informations, consultez ENTRYPOINT dans la documentation Docker.

L'extrait suivant est un exemple illustrant la syntaxe du fichier `Dockerrun.aws.json` pour un conteneur unique.

```
{
  "AWSEBDockerrunVersion": "1",
  "Image": {
    "Name": "janedoe/image",
    "Update": "true"
  },
  "Ports": [
    {
```

```
      "ContainerPort": "1234"
    }
  ],
  "Volumes": [
    {
      "HostDirectory": "/var/app/mydb",
      "ContainerDirectory": "/etc/mysql"
    }
  ],
  "Logging": "/var/log/nginx",
  "Entrypoint": "/app/bin/myapp",
  "Command": "--argument"
}>
```

Vous pouvez fournir à Elastic Beanstalk le fichier `Dockerrun.aws.json` uniquement, ou une archive `.zip` contenant les fichiers `Dockerrun.aws.json` et `Dockerfile`. Si vous fournissez les deux fichiers, le `Dockerfile` décrit l'image Docker et le fichier `Dockerrun.aws.json` fournit des informations supplémentaires pour le déploiement, comme indiqué ultérieurement dans cette section.

> **Note**
>
> Les deux fichiers doivent être à la racine, ou au niveau supérieur, de l'archive `.zip`. Ne créez pas l'archive à partir d'un répertoire contenant les fichiers. Au lieu de cela, parcourez ce répertoire et créez l'archive dans le répertoire.
> Si vous fournissez les deux fichiers, ne spécifiez aucune image dans le fichier `Dockerrun.aws.json`. Elastic Beanstalk crée et utilise l'image décrite dans le fichier `Dockerfile` et ignore celle spécifiée dans le fichier `Dockerrun.aws.json`.

Création d'images personnalisées avec un Dockerfile

Vous devez créer un `Dockerfile` si vous ne disposez d'aucune image existante hébergée dans un référentiel.

L'extrait suivant représente un exemple de `Dockerfile`. Si vous suivez les instructions de la page QuickStart pour Docker, vous pouvez charger ce fichier `Dockerfile` comme indiqué. Elastic Beanstalk exécute le jeu 2048 lorsque vous utilisez ce fichier `Dockerfile`.

Pour plus d'informations sur les instructions que vous pouvez inclure dans le `Dockerfile`, consultez la référence sur Dockerfile sur le site web de Docker.

```
FROM ubuntu:12.04

RUN apt-get update
RUN apt-get install -y nginx zip curl

RUN echo "daemon off;" >> /etc/nginx/nginx.conf
RUN curl -o /usr/share/nginx/www/master.zip -L https://codeload.github.com/
gabrielecirulli/2048/zip/master
RUN cd /usr/share/nginx/www/ && unzip master.zip && mv 2048-master/* . && rm -rf 2048-
master master.zip

EXPOSE 80

CMD ["/usr/sbin/nginx", "-c", "/etc/nginx/nginx.conf"]
```

> **Note**
>
> Vous pouvez exécuter des versions à plusieurs étapes à partir d'un seul Dockerfile pour produire des images de plus petite taille avec une réduction significative de la complexité. Pour plus d'informations, consultez Utiliser des versions à plusieurs étapes sur le site web de la documentation Docker.

Utilisation de la branche de plateforme Docker gérée par ECS dans Elastic Beanstalk

Cette rubrique fournit une vue d'ensemble des branches de la plateforme Docker gérées par Elastic Beanstalk ECS pour Amazon Linux 2 et Amazon Linux 2023. Il fournit également des informations de configuration spécifiques à la plate-forme gérée Docker ECS.

Migration depuis Docker multi-conteneurs sur AL1

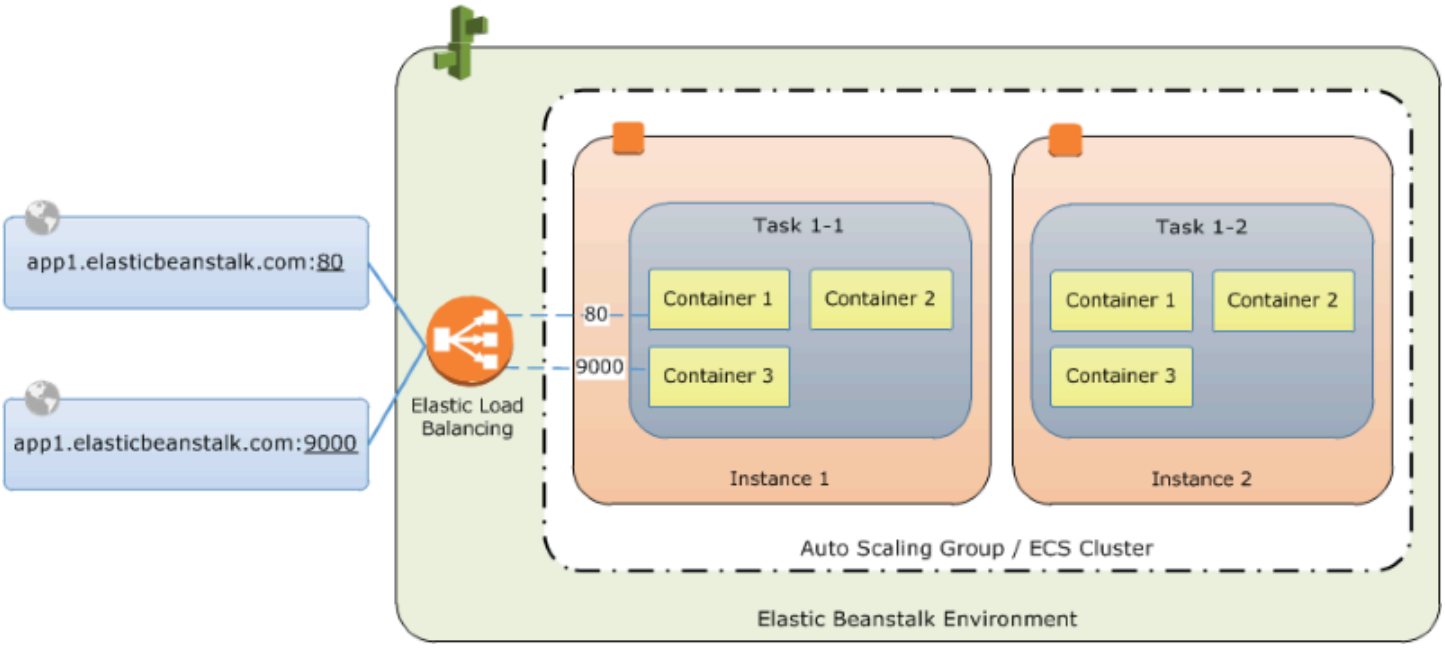
Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Bien que ce chapitre fournisse des informations de configuration pour cette plate-forme retirée, nous vous recommandons vivement de migrer vers la dernière branche de plate-forme prise en charge. Si vous utilisez actuellement le Docker multi-conteneur retiré qui s'exécute sur une branche de AL1 plateforme, vous pouvez migrer vers la dernière branche ECS Running on AL2023 platform. La dernière branche de plateforme prend en charge toutes les fonctions de la branche de plateforme abandonnée. Aucune modification

du code source n'est requise. Pour de plus amples informations, veuillez consulter Migration de votre application Elastic Beanstalk depuis AL1 Docker multi-conteneurs géré par ECS vers ECS sur Amazon Linux 2023.

Présentation de la plateforme Docker gérée par ECS

Elastic Beanstalk utilise Amazon Elastic Container Service (Amazon ECS) pour coordonner les déploiements de conteneurs vers des environnements Docker gérés par ECS. Amazon ECS fournit des outils permettant de gérer un cluster d'instances exécutant des conteneurs Docker. Elastic Beanstalk prend en charge les tâches Amazon ECS, y compris la création de clusters, la définition et l'exécution des tâches. Chacune des instances de l'environnement exécute le même ensemble de conteneurs, qui sont définis dans un fichier `Dockerrun.aws.json` v2. Afin de tirer pleinement parti de Docker, Elastic Beanstalk vous permet de créer un environnement dans lequel vos instances Amazon EC2 exécutent plusieurs conteneurs Docker côte à côte.

Le schéma suivant présente un exemple d'environnement Elastic Beanstalk configuré avec trois conteneurs Docker exécutés sur chaque instance Amazon EC2 d'un groupe Auto Scaling :



Ressources Amazon ECS créées par Elastic Beanstalk

Lorsque vous créez un environnement via la plateforme Docker gérée par ECS, Elastic Beanstalk crée et configure automatiquement plusieurs ressources Amazon Elastic Container Service tout en créant l'environnement. Ce faisant, il crée les conteneurs nécessaires sur chaque instance Amazon EC2.

- Cluster Amazon ECS – Dans Amazon ECS, les instances de conteneur sont organisées en clusters. En cas d'utilisation avec Elastic Beanstalk, un cluster est toujours créé pour chaque environnement Docker géré par ECS. Un cluster ECS contient également des fournisseurs de capacité de groupe Auto Scaling et d'autres ressources.
- Définition de tâche Amazon ECS : Elastic Beanstalk utilise le fichier `Dockerrun.aws.json` v2 dans votre projet pour générer la définition de tâche Amazon ECS qui est utilisée pour configurer les instances de conteneur dans l'environnement.
- Tâche Amazon ECS – Elastic Beanstalk communique avec Amazon ECS pour exécuter une tâche sur chaque instance de l'environnement afin de coordonner le déploiement de conteneur. Dans un environnement évolutif, Elastic Beanstalk lance une nouvelle tâche chaque fois qu'une instance est ajoutée au cluster.
- Agent de conteneur Amazon ECS – L'agent s'exécute dans un conteneur Docker sur les instances de votre environnement. L'agent interroge Amazon ECS service et attend l'exécution d'une tâche.
- Amazon ECS Data Volumes — Outre les volumes que vous définissez dans la `Dockerrun.aws.json` version v2, Elastic Beanstalk insère des définitions de volumes dans la définition des tâches afin de faciliter la collecte des journaux.

  Elastic Beanstalk crée des volumes de fichiers journaux sur l'instance de conteneur (un pour chaque conteneur), à l'emplacement `/var/log/containers/`*`containername`*. Ces volumes sont nommés `awseb-logs-`*`containername`* et sont fournis pour être montés par les conteneurs. Pour de plus amples informations sur la façon de les monter, veuillez consulter Format des définitions de conteneur.

Pour plus d'informations sur les ressources Amazon ECS, consultez le guide du développeur Amazon Elastic Container Service.

`Dockerrun.aws.json`fichier v2

Les instances de conteneur nécessitent un fichier de configuration nommé`Dockerrun.aws.json`. Les instances de conteneur font référence aux instances Amazon EC2 exécutant Docker géré par ECS dans un environnement Elastic Beanstalk. Ce fichier est propre à Elastic Beanstalk. Il peut être utilisé seul ou combiné au code source et au contenu d'un bundle de fichiers source afin de créer un environnement sur une plateforme Docker.

> **Note**
>
> La version 2 du format `Dockerrun.aws.json` ajoute la prise en charge de plusieurs conteneurs par instance Amazon EC2 et ne peut être utilisée qu'avec la plateforme Docker multiconteneurs gérée par ECS. Le format est très différent des autres versions de fichiers de configuration qui prennent en charge les branches de la plateforme Docker qui ne sont pas gérées par ECS.

Consultez le `Dockerrun.aws.json v2` pour plus de détails sur le format mis à jour et un exemple de fichier.

Images Docker

La plateforme Docker gérée par ECS pour Elastic Beanstalk nécessite que les images soient préconstruites et stockées dans un référentiel d'images en ligne public ou privé avant de créer un environnement Elastic Beanstalk.

> **Note**
>
> La création d'images personnalisées pendant le déploiement via un `Dockerfile` n'est pas prise en charge par la plateforme Docker gérée par ECS sur Elastic Beanstalk. Créez vos images et déployez-les dans un référentiel en ligne avant de créer un environnement Elastic Beanstalk.

Spécifiez des images par nom dans `Dockerrun.aws.json` v2.

Pour configurer Elastic Beanstalk afin qu'il s'authentifie auprès d'un référentiel privé, incluez le paramètre `authentication` dans votre fichier `Dockerrun.aws.json` v2.

Échec de déploiements de conteneurs

En cas d'échec d'une tâche Amazon ECS, un ou plusieurs conteneurs de votre environnement Elastic Beanstalk ne démarreront pas. Elastic Beanstalk ne restaure pas les environnements multiconteneurs suite à l'échec d'une tâche Amazon ECS. Si le démarrage d'un conteneur échoue dans votre environnement, redéployez la version actuelle ou une version de travail précédente à partir de la console Elastic Beanstalk.

Pour déployer une version existante

1. Ouvrez la console Elastic Beanstalk dans la région de votre environnement.
2. Cliquez sur Actions à droite du nom de votre application, puis cliquez sur View application versions (Afficher les versions de l'application).
3. Sélectionnez une version de votre application, puis cliquez sur Déploiement.

Extension des plateformes Docker basées sur ECS pour Elastic Beanstalk

Elastic Beanstalk propose des fonctionnalités d'extensibilité qui vous permettent d'appliquer vos propres commandes, scripts, logiciels et configurations à vos déploiements d'applications. Le flux de travail de déploiement pour l'ECS AL2 et les branches de AL2023 plate-forme varie légèrement par rapport aux autres plateformes basées sur Linux. Pour de plus amples informations, veuillez consulter Flux de déploiement d'instance pour ECS s'exécutant sur Amazon Linux 2 et versions ultérieures.

Configuration Docker gérée par ECS pour Elastic Beanstalk

Ce chapitre explique comment configurer votre environnement Docker géré par ECS. La liste suivante récapitule les éléments de configuration décrits dans ce chapitre.

- `Dockerrun.aws.jsonv2` — Ce fichier de configuration spécifie votre référentiel d'images et le nom de vos images Docker, entre autres composants.
- Rôle du profil d'instance EC2 : si vous avez un profil d'instance personnalisé, nous expliquons comment le configurer afin que les autorisations requises pour qu'ECS gère vos conteneurs restent à jour.
- Écouteurs Elastic Load Balancing : vous devez configurer plusieurs écouteurs Elastic Load Balancing si votre environnement doit prendre en charge le trafic entrant pour les proxys ou autres services qui ne s'exécutent pas sur le port HTTP par défaut.

Rubriques

- Configuration du fichier DockerRun.aws.json v2
- Politique de gestion des conteneurs et rôle d'instance EC2
- Utilisation de plusieurs écouteurs Elastic Load Balancing

Configuration du fichier DockerRun.aws.json v2

`Dockerrun.aws.json v2` est un fichier de configuration Elastic Beanstalk qui décrit comment déployer un ensemble de conteneurs Docker hébergés dans un cluster ECS dans un environnement Elastic Beanstalk. La plateforme Elastic Beanstalk crée une définition de tâche ECS, qui inclut une définition de conteneur ECS. Ces définitions sont décrites dans le fichier de configuration `Dockerrun.aws.json`.

La définition de conteneur contenue dans le fichier `Dockerrun.aws.json` décrit les conteneurs à déployer sur chaque instance Amazon EC2 du cluster ECS. Dans ce cas, une instance Amazon EC2 est également appelée instance de conteneur hôte, car elle héberge les conteneurs Docker. Le fichier de configuration décrit également les volumes de données à créer sur l'instance de conteneur hôte pour le montage des conteneurs Docker. Pour plus d'informations et un diagramme des composants d'un environnement Docker géré par ECS sur Elastic Beanstalk, consultez Présentation de la plateforme Docker gérée par ECS plus haut dans ce chapitre.

Un fichier `Dockerrun.aws.json` peut être utilisé de façon autonome ou compressé avec un code source supplémentaire dans une archive unique. Le code source archivé dans un fichier `Dockerrun.aws.json` est déployé dans les instances de conteneur Amazon EC2 et accessible dans le répertoire `/var/app/current/`.

Rubriques

- Dockerrun.aws.json v2
- Format de volume
- Format ARN du rôle d'exécution
- Format des définitions de conteneur
- Format d'authentification : utilisation d'images à partir d'un référentiel privé
- Exemple de Dockerrun.aws.json v2

`Dockerrun.aws.json` v2

Le fichier `Dockerrun.aws.json` comprend les sections suivantes :

AWSEBDockerrunVersion

Spécifie le numéro de version comme valeur 2 pour les environnements Docker gérés par ECS.

executionRoleArn

Spécifie les rôles IAM d'exécution des tâches pour différents objectifs et services associés à votre compte. Pour que votre application utilise les variables d'environnement Elastic Beanstalk stockées sous forme de secrets, vous devez spécifier l'ARN d'un rôle d'exécution de tâche qui accorde les autorisations requises. D'autres cas d'utilisation courants peuvent également nécessiter ce paramètre. Pour de plus amples informations, veuillez consulter Format ARN du rôle d'exécution.

volumes

Permet de créer des volumes à partir de dossiers de l'instance de conteneur Amazon EC2 ou à partir du bundle de fichiers source (déployé sur `/var/app/current`). Montez ces volumes sur les chemins de vos conteneurs Docker à l'aide des `mountPoints` de la section `containerDefinitions`.

containerDefinitions

Un tableau de définitions de conteneurs.

authentification (facultatif)

Dans Amazon S3, emplacement d'un fichier `.dockercfg` contenant les données d'authentification pour un référentiel privé.

Les sections containerDefinitions et volumes du fichier `Dockerrun.aws.json` utilisent le même format que les sections correspondantes d'un fichier de définition de tâche Amazon ECS. Pour en savoir plus sur le format de définition de tâche et obtenir la liste complète des paramètres de définition de tâche, consultez Définitions des tâches Amazon ECS (français non garanti) dans le Guide du développeur Amazon Elastic Container Service.

Format de volume

Le paramètre volume crée des volumes soit à partir de dossiers dans l'instance de conteneur Amazon EC2, soit à partir de votre solution groupée source (déployée sur `/var/app/current`).

Les volumes sont spécifiés dans le format suivant :

```
"volumes": [
    {
      "name": "volumename",
      "host": {
```

```
        "sourcePath": "/path/on/host/instance"
      }
    }
  ],
```

Montez ces volumes sur les chemins de vos conteneurs Docker à l'aide des `mountPoints` de la définition de conteneur.

Elastic Beanstalk configure des volumes supplémentaires pour les journaux (un pour chaque conteneur). Ils doivent être montés par vos conteneurs Docker afin d'écrire les journaux dans l'instance de l'hôte.

Pour plus de détails, consultez le champ `mountPoints` dans la section Format de définition de conteneur qui suit.

Format ARN du rôle d'exécution

Pour que votre application puisse utiliser les variables d'environnement Elastic Beanstalk stockées sous forme de secrets, vous devez spécifier un rôle IAM d'exécution de tâches. Le rôle doit autoriser le conteneur Amazon ECS à effectuer des appels d' AWS API en votre nom à l'aide de AWS Secrets Manager secrets ou de AWS Systems Manager paramètres du magasin de paramètres pour référencer des données sensibles. Pour obtenir des instructions sur la création d'un rôle IAM d'exécution de tâches avec les autorisations requises pour votre compte, consultez le rôle IAM d'exécution de tâches Amazon ECS dans le guide du développeur Amazon Elastic Container Service.

```
{
"AWSEBDockerrunVersion": 2,
  "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
```

Autorisations supplémentaires requises pour la plateforme Docker gérée par Amazon ECS

Attributions de profil d'instance EC2 **`iam:PassRole`** à ECS

Pour que votre profil d'instance EC2 puisse accorder ce rôle au conteneur ECS, vous devez inclure l'`iam:PassRole`autorisation démontrée dans l'exemple suivant. `iam:PassRole`Permet aux instances EC2 de transmettre le rôle d'exécution des tâches au conteneur ECS.

Dans cet exemple, nous limitons l'instance EC2 pour ne transmettre le rôle qu'au service ECS. Bien que cette condition ne soit pas obligatoire, nous l'ajoutons afin de suivre les meilleures pratiques afin de réduire la portée des autorisations partagées. Nous y parvenons avec l'`Condition`élément.

> **Note**
>
> Toute utilisation du rôle d'exécution de tâches ECS IAM nécessite une `iam:PassRole` autorisation. Il existe d'autres cas d'utilisation courants qui nécessitent le rôle de service géré d'exécution de tâches ECS. Pour plus d'informations, consultez le rôle IAM d'exécution de tâches Amazon ECS dans le manuel Amazon Elastic Container Service Developer Guide.

Example politique avec **`iam:PassRole`** autorisation

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::123456789012:role/ecs-task-execution-role"
            ],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "ecs-tasks.amazonaws.com"
                }
            }
        }
    ]
}
```

Octroi d'un accès aux secrets et aux paramètres à l'agent de conteneur Amazon ECS

Le rôle IAM d'exécution de tâches Amazon ECS a également besoin d'autorisations pour accéder aux secrets et aux magasins de paramètres. À l'instar des exigences du rôle de profil d'instance EC2, l'agent de conteneur ECS a besoin d'une autorisation pour extraire les ressources Secrets Manager ou Systems Manager nécessaires. Pour plus d'informations, consultez les autorisations Secrets Manager ou Systems Manager dans le guide du développeur Amazon Elastic Container Service.

Octroi de secrets et d'accès aux paramètres aux instances Elastic Beanstalk EC2

Pour prendre en charge les secrets configurés en tant que variables d'environnement, vous devez également ajouter des autorisations à votre profil d'instance EC2. Pour plus d'informations, consultez Extraction de secrets et de paramètres dans les variables d'environnement Elastic Beanstalk et Autorisations IAM requises pour Secrets Manager.

Les exemples suivants combinent l'`iam:PassRole`exemple précédent avec les exemples fournis dans le document référencéAutorisations IAM requises pour Secrets Manager. Ils ajoutent les autorisations dont les instances EC2 ont besoin pour accéder aux AWS Systems Manager magasins AWS Secrets Manager et récupérer les secrets et les données de paramètres afin d'initialiser les variables d'environnement Elastic Beanstalk qui ont été configurées pour les secrets.

Example Politique de Secrets Manager associée à une **`iam:PassRole`** autorisation

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
       {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::123456789012:role/ecs-task-execution-role"
            ],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "ecs-tasks.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:secretsmanager:us-east-1:111122223333:secret:my-secret",
                "arn:aws:kms:us-east-1:111122223333:key/my-key"
            ]
```

```
        }
    ]
}
```

Example Politique de Systems Manager associée à une **iam:PassRole** autorisation

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
       {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::123456789012:role/ecs-task-execution-role"
            ],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "ecs-tasks.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameter",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-1:111122223333:parameter/my-parameter",
                "arn:aws:kms:us-east-1:111122223333:key/my-key"
            ]
        }
    ]
}
```

Format des définitions de conteneur

Les exemples suivants illustrent un sous-ensemble de paramètres fréquemment utilisés dans la section containerDefinitions. D'autres paramètres facultatifs sont disponibles.

La plateforme Beanstalk crée une définition de tâche ECS, qui inclut une définition de conteneur ECS. Beanstalk prend en charge un sous-ensemble de paramètres pour la définition du conteneur ECS. Pour plus d'informations, consultez Définitions de conteneurs dans le Guide du développeur Amazon Elastic Container Service.

Un fichier `Dockerrun.aws.json` contient un tableau d'un ou plusieurs objets de définition de conteneur comportant les champs suivants :

nom

Nom du conteneur. Consultez Paramètres de définition de conteneur standards pour obtenir plus d'informations sur la longueur maximale et les caractères autorisés.

image

Nom d'une image Docker dans un référentiel Docker en ligne à partir de laquelle vous créez un conteneur Docker. Notez ces conventions :

- Les images dans les référentiels officiels sur Docker Hub utilisent un nom unique (par exemple, `ubuntu` ou `mongo`).
- Les images dans les autres référentiels sur Docker Hub sont qualifiées par un nom d'organisation (par exemple, `amazon/amazon-ecs-agent`.
- Les images dans les autres référentiels en ligne sont qualifiées par un nom de domaine (par exemple, `quay.io/assemblyline/ubuntu`).

environment

Tableau des variables d'environnement à transmettre au conteneur.

Par exemple, l'entrée suivante définit une variable d'environnement avec le nom **`Container`** et la valeur **`PHP`** :

```
"environment": [
  {
    "name": "Container",
    "value": "PHP"
```

```
  }
],
```

essential

True si la tâche doit s'arrêter en cas d'échec du conteneur. Les conteneurs non essentiels peuvent s'arrêter ou se bloquer sans conséquences sur les autres conteneurs de l'instance.

memory

Quantité de mémoire sur l'instance de conteneur à réserver pour le conteneur. Spécifiez un entier non nul pour l'un des paramètres `memory` ou `memoryReservation` (ou les deux) dans les définitions de conteneur.

memoryReservation

La limite flexible (en MiB) de mémoire à réserver pour le conteneur. Spécifiez un entier non nul pour l'un des paramètres `memory` ou `memoryReservation` (ou les deux) dans les définitions de conteneur.

mountPoints

Volumes à monter à partir de l'instance de conteneur Amazon EC2 et emplacement dans lequel ils doivent être montés dans le système de fichiers du conteneur Docker. Lorsque vous montez des volumes qui contiennent des contenus applicatifs, votre conteneur peut lire les données que vous chargez dans votre bundle source. Lorsque vous montez des volumes de journal pour écrire des données de journaux, Elastic Beanstalk peut collecter les données de journaux à partir de ces volumes.

Elastic Beanstalk crée les volumes de fichiers journaux sur l'instance de conteneur (un pour chaque conteneur Docker), à l'emplacement `/var/log/containers/`*`containername`*. Ces volumes sont nommés `awseb-logs-`*`containername`* et doivent être montés à l'emplacement où les journaux sont écrits dans le système de fichiers du conteneur.

Par exemple, le point de montage suivant mappe l'emplacement du journal nginx dans le conteneur sur le volume généré par Elastic Beanstalk pour le conteneur `nginx-proxy`.

```
{
  "sourceVolume": "awseb-logs-nginx-proxy",
  "containerPath": "/var/log/nginx"
}
```

portMappings

Mappe les ports réseau du conteneur sur les ports de l'hôte.

links

Liste des conteneurs à lier les uns aux autres. Les conteneurs liés peuvent s'identifier entre eux et communiquer en toute sécurité.

volumesFrom

Montez tous les volumes à partir d'un autre conteneur. Par exemple, pour monter les volumes à partir d'un conteneur nommé `web`:

```
"volumesFrom": [
  {
    "sourceContainer": "web"
  }
],
```

Format d'authentification : utilisation d'images à partir d'un référentiel privé

La section `authentication` contient les données d'authentification pour un référentiel privé. Cette entrée est facultative.

Ajoutez les informations relatives au compartiment Amazon S3 contenant le fichier d'authentification dans le paramètre `authentication` du fichier `Dockerrun.aws.json`. Assurez-vous que le paramètre `authentication` contient une clé et un compartiment Amazon S3 valides. Le compartiment Amazon S3 doit être hébergé dans la même région que l'environnement qui l'utilise. Elastic Beanstalk ne télécharge pas de fichiers à partir de compartiments Amazon S3 hébergés dans d'autres régions.

Utilise le format suivant :

```
"authentication": {
    "bucket": "amzn-s3-demo-bucket",
    "key": "mydockercfg"
  },
```

Pour plus d'informations sur la génération et le chargement du fichier d'authentification, consultez Authentification avec des référentiels d'images.

Exemple de Dockerrun.aws.json v2

L'extrait suivant est un exemple illustrant la syntaxe du fichier `Dockerrun.aws.json` pour une instance avec deux conteneurs.

```
{
  "AWSEBDockerrunVersion": 2,
  "volumes": [
    {
      "name": "php-app",
      "host": {
        "sourcePath": "/var/app/current/php-app"
      }
    },
    {
      "name": "nginx-proxy-conf",
      "host": {
        "sourcePath": "/var/app/current/proxy/conf.d"
      }
    }
  ],
  "containerDefinitions": [
    {
      "name": "php-app",
      "image": "php:fpm",
      "environment": [
        {
          "name": "Container",
          "value": "PHP"
        }
      ],
      "essential": true,
      "memory": 128,
      "mountPoints": [
        {
          "sourceVolume": "php-app",
          "containerPath": "/var/www/html",
          "readOnly": true
        }
      ]
    },
    {
      "name": "nginx-proxy",
      "image": "nginx",
```

```
      "essential": true,
      "memory": 128,
      "portMappings": [
        {
          "hostPort": 80,
          "containerPort": 80
        }
      ],
      "links": [
        "php-app"
      ],
      "mountPoints": [
        {
          "sourceVolume": "php-app",
          "containerPath": "/var/www/html",
          "readOnly": true
        },
        {
          "sourceVolume": "nginx-proxy-conf",
          "containerPath": "/etc/nginx/conf.d",
          "readOnly": true
        },
        {
          "sourceVolume": "awseb-logs-nginx-proxy",
          "containerPath": "/var/log/nginx"
        }
      ]
    }
  ]
}
```

Politique de gestion des conteneurs et rôle d'instance EC2

Lorsque vous créez un environnement dans la console Elastic Beanstalk, celle-ci vous invite à créer un profil d'instance par défaut qui inclut la politique gérée. `AWSElasticBeanstalkMulticontainerDocker` Dans un premier temps, votre profil d'instance EC2 par défaut doit donc inclure cette politique gérée. Si votre environnement utilise un rôle de profil d'instance EC2 personnalisé au lieu du rôle par défaut, assurez-vous que la politique gérée `AWSElasticBeanstalkMulticontainerDocker` est attachée afin que les autorisations requises pour la gestion des conteneurs soient conservées. up-to-date

Elastic Beanstalk utilise une AMI optimisée pour Elastic Beanstalk avec un agent de conteneur Amazon ECS qui s'exécute dans un conteneur Docker. L'agent communique avec Amazon ECS pour coordonner les déploiements de conteneur. Pour communiquer avec Amazon ECS, chaque instance Amazon EC2 doit disposer des autorisations IAM correspondantes, spécifiées dans cette politique gérée. Consultez le Guide AWSElasticBeanstalkMulticontainerDockerde référence des politiques AWS gérées pour consulter ces autorisations.

Si vous utilisez des variables d'environnement Elastic Beanstalk configurées pour accéder aux secrets ou aux paramètres AWS Secrets Manager stockés AWS Systems Manager dans ou Parameter Store, vous devez personnaliser votre profil d'instance EC2 avec des autorisations supplémentaires. Pour de plus amples informations, veuillez consulter Format ARN du rôle d'exécution.

Utilisation de plusieurs écouteurs Elastic Load Balancing

Vous pouvez configurer plusieurs écouteurs Elastic Load Balancing dans un environnement Docker géré par ECS afin de prendre en charge le trafic entrant pour les proxys ou autres services qui ne fonctionnent pas sur le port HTTP par défaut.

Créez un dossier `.ebextensions` dans votre bundle de fichiers source et ajoutez un fichier avec une extension `.config`. L'exemple suivant présente un fichier de configuration qui crée un écouteur Elastic Load Balancing sur le port 8080.

`.ebextensions/elb-listener.config`

```
option_settings:
  aws:elb:listener:8080:
    ListenerProtocol: HTTP
    InstanceProtocol: HTTP
    InstancePort: 8080
```

Si votre environnement s'exécute dans un Amazon Virtual Private Cloud (Amazon VPC) personnalisé que vous avez créé, Elastic Beanstalk s'occupe du reste. Dans un VPC par défaut, vous devez configurer le groupe de sécurité de votre instance pour autoriser le trafic entrant provenant de l'équilibreur de charge. Ajoutez un deuxième fichier de configuration, qui ajoute une règle de trafic entrant au groupe de sécurité :

`.ebextensions/elb-ingress.config`

```
Resources:
```

```
  port8080SecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 8080
      FromPort: 8080
      SourceSecurityGroupName: { "Fn::GetAtt": ["AWSEBLoadBalancer",
 "SourceSecurityGroup.GroupName"] }
```

Pour de plus amples informations sur le format du fichier de configuration, veuillez consulter Ajout et personnalisation des ressources de l'environnement Elastic Beanstalk et Paramètres d'option.

En plus d'ajouter un écouteur à la configuration Elastic Load Balancing et d'ouvrir un port dans le groupe de sécurité, vous devez mapper le port sur l'instance de l'hôte à un port sur le conteneur Docker, dans la section `containerDefinitions` du fichier `Dockerrun.aws.json` v2. L'extrait suivant en présente un exemple:

```
"portMappings": [
  {
    "hostPort": 8080,
    "containerPort": 8080
  }
]
```

Voir `Dockerrun.aws.json` v2 pour plus de détails sur le format de fichier `Dockerrun.aws.json` v2.

Création d'un environnement Docker géré par ECS avec la console Elastic Beanstalk

Ce didacticiel détaille la configuration des conteneurs et la préparation du code source pour un environnement Docker géré par ECS qui utilise deux conteneurs.

Les conteneurs, une application PHP et un proxy nginx, s'exécutent côte à côte sur chacune des instances Amazon Elastic Compute Cloud (EC2Amazon) dans un environnement Elastic Beanstalk. Après avoir créé l'environnement et vérifié que les applications sont en cours d'exécution, vous allez vous connecter à une instance de conteneur pour voir comment tout fonctionne ensemble.

Sections

- Définir les conteneurs Docker gérés par ECS
- Ajout de contenu
- Déploiement sur Elastic Beanstalk
- Connexion à une instance de conteneur
- Mise à jour de l'agent du conteneur Amazon ECS

Définir les conteneurs Docker gérés par ECS

La première étape de la création d'un environnement Docker consiste à créer un répertoire pour vos données d'application. Ce dossier peut être situé n'importe où sur votre ordinateur local et porter le nom de votre choix. En plus d'un fichier de configuration de conteneur, ce dossier inclut le contenu que vous téléchargez sur Elastic Beanstalk et que vous déployez dans votre environnement.

> **Note**
>
> L'ensemble du code de ce didacticiel est disponible dans le référentiel awslabs à l' GitHub adresse. https://github.com/awslabs/eb-docker-nginx-proxy

Le fichier utilisé par Elastic Beanstalk pour configurer les conteneurs sur une instance EC2 Amazon est un fichier texte au format JSON nommé v2. `Dockerrun.aws.json` Les versions de la plateforme Docker gérée par ECS utilisent le format version 2 de ce fichier. Ce format ne peut être utilisé qu'avec la plate-forme Docker gérée par ECS, car il diffère considérablement des autres versions de fichiers de configuration qui prennent en charge les branches de la plate-forme Docker qui ne sont pas gérées par ECS.

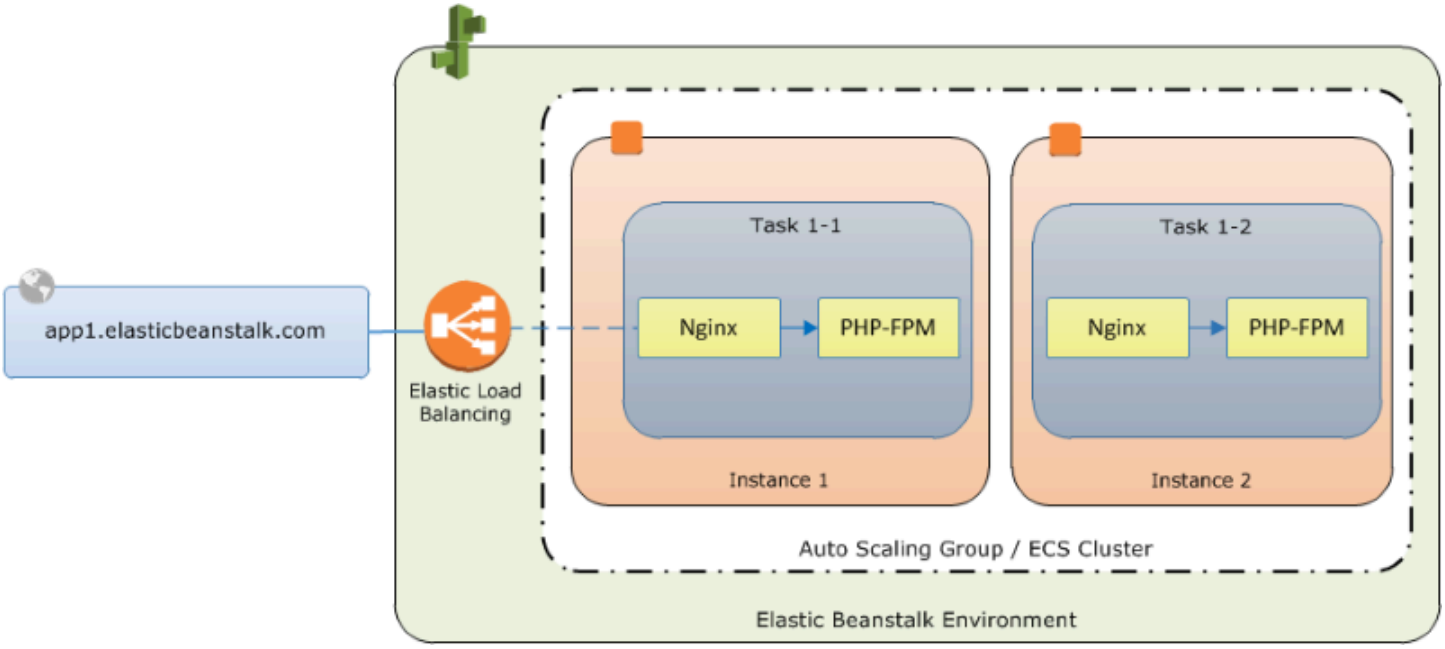
Créez un fichier texte `Dockerrun.aws.json` v2 portant ce nom à la racine de votre application et ajoutez le texte suivant :

```
{
  "AWSEBDockerrunVersion": 2,
  "volumes": [
    {
      "name": "php-app",
      "host": {
        "sourcePath": "/var/app/current/php-app"
      }
    },
```

```
    {
      "name": "nginx-proxy-conf",
      "host": {
        "sourcePath": "/var/app/current/proxy/conf.d"
      }
    }
  ],
  "containerDefinitions": [
    {
      "name": "php-app",
      "image": "php:fpm",
      "essential": true,
      "memory": 128,
      "mountPoints": [
        {
          "sourceVolume": "php-app",
          "containerPath": "/var/www/html",
          "readOnly": true
        }
      ]
    },
    {
      "name": "nginx-proxy",
      "image": "nginx",
      "essential": true,
      "memory": 128,
      "portMappings": [
        {
          "hostPort": 80,
          "containerPort": 80
        }
      ],
      "links": [
        "php-app"
      ],
      "mountPoints": [
        {
          "sourceVolume": "php-app",
          "containerPath": "/var/www/html",
          "readOnly": true
        },
        {
          "sourceVolume": "nginx-proxy-conf",
          "containerPath": "/etc/nginx/conf.d",
```

```
          "readOnly": true
        },
        {
          "sourceVolume": "awseb-logs-nginx-proxy",
          "containerPath": "/var/log/nginx"
        }
      ]
    }
  ]
}
```

Cet exemple de configuration définit deux conteneurs, un site web PHP avec un proxy nginx devant. Ces deux conteneurs s'exécutent côte à côte dans des conteneurs Docker sur chaque instance dans votre environnement Elastic Beanstalk, accédant à du contenu partagé (le contenu du site web) à partir de volumes sur l'instance hôte, qui sont aussi définis dans ce fichier. Les conteneurs eux-mêmes sont créés à partir d'images hébergées dans des référentiels officiels sur Docker Hub. Vous obtenez alors un environnement similaire au suivant :



Les volumes définis dans la configuration correspondent au contenu que vous allez créer ensuite et télécharger dans le cadre du groupe source de votre application. Les conteneurs accèdent au contenu sur l'hôte en montant des volumes dans la section `mountPoints` des définitions de conteneur.

Pour plus d'informations sur le format de la `Dockerrun.aws.json` v2 et ses paramètres, consultezFormat des définitions de conteneur.

Ajout de contenu

Ensuite, vous allez ajouter du contenu à votre site PHP pour l'afficher aux visiteurs, et un fichier de configuration pour le proxy nginx.

php-app/index.php

```
<h1>Hello World!!!</h1>
<h3>PHP Version <pre><?= phpversion()?></pre></h3>
```

php-app/static.html

```
<h1>Hello World!</h1>
<h3>This is a static HTML page.</h3>
```

proxy/conf.d/default.conf

```
server {
  listen 80;
  server_name localhost;
  root /var/www/html;

  index index.php;

  location ~ [^/]\.php(/|$) {
    fastcgi_split_path_info ^(.+?\.php)(/.*)$;
    if (!-f $document_root$fastcgi_script_name) {
      return 404;
    }

    include fastcgi_params;
    fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    fastcgi_param PATH_INFO $fastcgi_path_info;
    fastcgi_param PATH_TRANSLATED $document_root$fastcgi_path_info;

    fastcgi_pass php-app:9000;
    fastcgi_index index.php;
  }
}
```

Déploiement sur Elastic Beanstalk

Votre dossier d'application contient maintenant les fichiers suivants :

```
### Dockerrun.aws.json
### php-app
#   ### index.php
#   ### static.html
### proxy
    ### conf.d
        ### default.conf
```

C'est tout ce dont vous avez besoin pour créer l'environnement Elastic Beanstalk. Créez une archive `.zip` des fichiers et dossiers ci-dessus (sans inclure le dossier de projet de niveau supérieur). Pour créer l'archive dans l'Explorateur Windows, sélectionnez le contenu du dossier de projet, effectuez un clic droit, sélectionnez Envoyer vers, puis cliquez sur Dossier compressé.

> **Note**
>
> Pour de plus amples informations sur la structure de fichiers requise et pour obtenir des instructions pour créer des archives dans d'autres environnements, veuillez consulter Création d'un bundle de sources d'applications Elastic Beanstalk

Ensuite, téléchargez le bundle de fichiers source sur Elastic Beanstalk et créez votre environnement. Pour Platform (Plateforme), sélectionnez Docker. Pour la branche Platform, sélectionnez ECS exécuté sur Amazon Linux 2023 64 bits.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk à l'aide de ce lien préconfiguré : console.aws.amazon.com/elasticbeanstalk/home#/newApplication? Nom de l'application = Tutoriels et type d'environnement = LoadBalanced
2. Pour Plateforme, sélectionnez la plateforme et la branche de plateforme qui correspondent à la langue utilisée par votre application, ou la plateforme Docker pour les applications basées sur des conteneurs.
3. Pour Application code (Code d'application), choisissez Upload your code (Charger votre code).
4. Choisissez Local file (Fichier local), Choose file (Choisir un fichier), puis ouvrez le bundle source.

5. Choisissez Vérifier et lancer.
6. Vérifiez les paramètres disponibles et choisissez Créer une application.

La console Elastic Beanstalk vous redirige vers le tableau de bord de gestion pour votre nouvel environnement. Cet écran présente l'état de l'environnement et la sortie d'événements par le service Elastic Beanstalk. Quand le statut est vert, cliquez sur l'URL à côté du nom de l'environnement pour voir votre nouveau site web.

Connexion à une instance de conteneur

Vous allez ensuite vous connecter à une EC2 instance Amazon dans votre environnement Elastic Beanstalk pour voir certains des éléments mobiles en action.

La manière la plus simple de se connecter à une instance dans votre environnement consiste à utiliser l'interface de ligne de commande (CLI) EB. Pour l'utiliser, installez l'interface de ligne de commande (CLI) EB, si vous ne l'avez pas déjà fait. Vous devrez également configurer votre environnement à l'aide d'une paire de clés Amazon EC2 SSH. Utilisez soit la page de configuration de la sécurité de la console, soit la commande eb init de l'interface de ligne de commande (CLI) EB. Pour vous connecter à une instance de l'environnement, utilisez la commande eb ssh de l'interface de ligne de commande (CLI) EB.

Maintenant que vous êtes connecté à une EC2 instance Amazon hébergeant vos conteneurs docker, vous pouvez voir comment les choses sont configurées. Exécutez `ls` sur `/var/app/current` :

```
[ec2-user@ip-10-0-0-117 ~]$ ls /var/app/current
Dockerrun.aws.json  php-app  proxy
```

Ce répertoire contient les fichiers du groupe source que vous avez téléchargés sur Elastic Beanstalk pendant la création de l'environnement.

```
[ec2-user@ip-10-0-0-117 ~]$ ls /var/log/containers
nginx-proxy    nginx-proxy-4ba868dbb7f3-stdouterr.log
php-app        php-app-dcc3b3c8522c-stdouterr.log       rotated
```

C'est à cet emplacement que les journaux sont créés sur l'instance de conteneur et collectés par Elastic Beanstalk. Elastic Beanstalk crée un volume dans ce répertoire pour chaque conteneur, que vous montez sur l'emplacement du conteneur où les journaux sont écrits.

Vous pouvez également regarder Docker pour voir les conteneurs en cours d'exécution avec `docker ps`.

```
[ec2-user@ip-10-0-0-117 ~]$ sudo docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED
  STATUS                  PORTS                                  NAMES

4ba868dbb7f3   nginx                                "/docker-entrypoint.…"   4 minutes ago
   Up 4 minutes            0.0.0.0:80->80/tcp, :::80->80/tcp   ecs-awseb-Tutorials-env-
dc2aywfjwg-1-nginx-proxy-acca84ef87c4aca15400
dcc3b3c8522c   php:fpm                              "docker-php-entrypoi…"   4 minutes ago
   Up 4 minutes            9000/tcp                               ecs-awseb-Tutorials-env-
dc2aywfjwg-1-php-app-b8d38ae288b7b09e8101
d9367c0baad6   amazon/amazon-ecs-agent:latest   "/agent"                 5 minutes ago
  Up 5 minutes (healthy)                                  ecs-agent
```

Vous pouvez y voir les deux conteneurs en cours d'exécution que vous avez déployés, ainsi que l'agent de conteneur Amazon ECS qui a coordonné le déploiement.

Mise à jour de l'agent du conteneur Amazon ECS

EC2 Les instances Amazon d'un environnement Docker géré par ECS sur Elastic Beanstalk exécutent un processus d'agent dans un conteneur Docker. Cet agent se connecte à Amazon ECS service afin de coordonner des déploiements de conteneurs. Ces déploiements sont exécutés comme des tâches dans Amazon ECS, qui sont configurées dans les fichiers de définition de tâche. Elastic Beanstalk crée ces fichiers de définition de tâche en se basant sur le fichier `Dockerrun.aws.json` que vous téléchargez dans un groupe de fichiers source.

Vérifiez le statut de l'agent de conteneur avec une demande get HTTP pour `http://localhost:51678/v1/metadata`:

```
[ec2-user@ip-10-0-0-117 ~]$ curl http://localhost:51678/v1/metadata
{
  "Cluster":"awseb-Tutorials-env-dc2aywfjwg",
  "ContainerInstanceArn":"arn:aws:ecs:us-west-2:123456789012:container-instance/awseb-
Tutorials-env-dc2aywfjwg/db7be5215cd74658aacfcb292a6b944f",
  "Version":"Amazon ECS Agent - v1.57.1 (089b7b64)"
}
```

Cette structure indique le nom du cluster Amazon ECS et l'ARN (Amazon Resource Name) de l'instance de cluster (l' EC2 instance Amazon à laquelle vous êtes connecté).

Pour plus d'informations, exécutez une requête get HTTP sur `http://localhost:51678/v1/tasks` :

```
[ec2-user@ip-10-0-0-117 ~]$ curl http://localhost:51678/v1/tasks
{
   "Tasks":[
      {
         "Arn":"arn:aws:ecs:us-west-2:123456789012:task/awseb-Tutorials-env-dc2aywfjwg/
bbde7ebe1d4e4537ab1336340150a6d6",
         "DesiredStatus":"RUNNING",
         "KnownStatus":"RUNNING",
         "Family":"awseb-Tutorials-env-dc2aywfjwg",
         "Version":"1",
         "Containers":[
            {

 "DockerId":"dcc3b3c8522cb9510b7359689163814c0f1453b36b237204a3fd7a0b445d2ea6",
               "DockerName":"ecs-awseb-Tutorials-env-dc2aywfjwg-1-php-app-
b8d38ae288b7b09e8101",
               "Name":"php-app",
               "Volumes":[
                  {
                     "Source":"/var/app/current/php-app",
                     "Destination":"/var/www/html"
                  }
               ]
            },
            {

 "DockerId":"4ba868dbb7f3fb3328b8afeb2cb6cf03e3cb1cdd5b109e470f767d50b2c3e303",
               "DockerName":"ecs-awseb-Tutorials-env-dc2aywfjwg-1-nginx-proxy-
acca84ef87c4aca15400",
               "Name":"nginx-proxy",
               "Ports":[
                  {
                     "ContainerPort":80,
                     "Protocol":"tcp",
                     "HostPort":80
                  },
                  {
                     "ContainerPort":80,
                     "Protocol":"tcp",
                     "HostPort":80
```

```
                }
            ],
            "Volumes":[
                {
                    "Source":"/var/app/current/php-app",
                    "Destination":"/var/www/html"
                },
                {
                    "Source":"/var/log/containers/nginx-proxy",
                    "Destination":"/var/log/nginx"
                },
                {
                    "Source":"/var/app/current/proxy/conf.d",
                    "Destination":"/etc/nginx/conf.d"
                }
            ]
        }
    ]
  }
}
```

Cette structure décrit la tâche qui est exécutée pour déployer les deux conteneurs Docker à partir du projet exemple de ce didacticiel. Les informations suivantes sont affichées :

- KnownStatus— L'`RUNNING`état indique que les conteneurs sont toujours actifs.
- Famille – Nom de la définition de tâche créée par Elastic Beanstalk à partir du fichier `Dockerrun.aws.json`.
- Version – Version de la définition de tâche. Elle est augmentée chaque fois que le fichier de définition de tâche est mis à jour.
- Containers – Informations sur les conteneurs exécutés sur l'instance.

Encore plus d'informations sont disponibles à partir d'Amazon ECS service lui-même, que vous pouvez appeler à l'aide de l AWS Command Line Interface. Pour obtenir des instructions sur l'utilisation AWS CLI avec Amazon ECS et des informations sur Amazon ECS en général, consultez le guide de l'utilisateur Amazon ECS.

Migration de votre application Elastic Beanstalk depuis AL1 Docker multi-conteneurs géré par ECS vers ECS sur Amazon Linux 2023

> **Note**
>
> Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. .

Cette rubrique vous guide dans la migration de vos applications depuis la branche de plate-forme obsolète Docker multi-conteneurs exécutée sur Amazon Linux 64 bits vers ECS fonctionnant sur AL2 64 bits 023. Cette branche de plateforme cible est à jour et prise en charge. Comme la branche Docker multi-conteneurs précédente, la nouvelle AL1 branche de la plateforme ECS AL2 023 utilise Amazon ECS pour coordonner le déploiement de plusieurs conteneurs Docker sur un cluster Amazon ECS dans un environnement Elastic Beanstalk. La nouvelle branche de plate-forme ECS AL2 023 prend en charge toutes les fonctionnalités de la précédente branche de plate-forme Docker AL1 multi-conteneurs. En outre, le même fichier `Dockerrun.aws.json` v2 est pris en charge.

Sections

- Migrer avec la console Elastic Beanstalk
- Migrez avec le AWS CLI

Migrer avec la console Elastic Beanstalk

Pour effectuer une migration à l'aide de la console Elastic Beanstalk, déployez le même code source dans un nouvel environnement basé sur la branche de plateforme ECS AL2 Running on 023. Aucune modification du code source n'est requise.

Pour migrer vers la branche de plateforme ECS s'exécutant sur Amazon Linux 2023

1. À l'aide de la source d'application déjà déployée dans l'ancien environnement, créez un groupe de sources d'application. Vous pouvez utiliser le même groupe de sources d'application et le même fichier `Dockerrun.aws.json` v2.
2. Créez un nouvel environnement à l'aide de la branche de plateforme ECS s'exécutant sur Amazon Linux 2023. Utilisez le groupe de source de l'étape précédente en tant qu'Application code (Code d'application). Pour obtenir des instructions plus détaillées, consultez Déploiement sur Elastic Beanstalk dans le Tutoriel Docker géré par ECS plus tôt dans ce chapitre.

Migrez avec le AWS CLI

Vous avez également la possibilité d'utiliser le AWS Command Line Interface (AWS CLI) pour migrer votre environnement Docker multi-conteneurs Amazon Linux Docker existant vers la nouvelle branche de la plateforme ECS AL2 023. Dans ce cas, vous n'avez pas besoin de créer un nouvel environnement ou de redéployer votre code source. Il suffit d'exécuter la commande AWS CLI update-environment. Elle effectuera une mise à jour de plateforme pour migrer votre environnement existant vers la branche de plateforme ECS Amazon Linux 2023.

Utilisez la syntaxe suivante pour migrer votre environnement vers la nouvelle branche de plateforme.

```
aws elasticbeanstalk update-environment \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 version running ECS" \
--region my-region
```

Voici un exemple de la commande de migration de l'environnement beta-101 vers la version 3.0.0 de la branche de plateforme ECS Amazon Linux 2023 dans la région us-east-1.

```
aws elasticbeanstalk update-environment \
--environment-name beta-101 \
--solution-stack-name "64bit Amazon Linux 2023 v4.0.0 running ECS" \
--region us-east-1
```

Le paramètre `solution-stack-name` fournit la branche de la plateforme et sa version. Utiliser la version de branche de plateforme la plus récente en spécifiant le bon nom de la pile de solutions. La version de chaque branche de plateforme est incluse dans le nom de la pile de solutions, comme illustré dans l'exemple ci-dessus. Pour obtenir la liste des piles de solutions les plus récentes pour la plateforme Docker, consultez Plateformes prises en charge dans le guide Plates-formes AWS Elastic Beanstalk .

> **Note**
>
> La list-available-solution-stackscommande fournit une liste des versions de plateforme disponibles pour votre compte dans une AWS région.
>
> ```
> aws elasticbeanstalk list-available-solution-stacks --region us-east-1 --query
>  SolutionStacks
> ```

Pour en savoir plus AWS CLI, consultez le guide de l'AWS Command Line Interface utilisateur. Pour plus d'informations sur AWS CLI les commandes pour Elastic Beanstalk, AWS CLI consultez le Command Reference for Elastic Beanstalk.

Authentification avec des référentiels d'images

Cette rubrique explique comment s'authentifier auprès de référentiels d'images en ligne avec Elastic Beanstalk. Pour les référentiels privés, Elastic Beanstalk doit s'authentifier avant de pouvoir extraire et déployer vos images. Pour Amazon ECR Public, l'authentification est facultative mais fournit des limites de débit plus élevées et une fiabilité améliorée.

Utilisation d'images à partir d'un référentiel Amazon ECR

Vous pouvez stocker vos images Docker personnalisées dans AWS Amazon Elastic Container Registry (Amazon ECR).

Lorsque vous stockez vos images Docker dans Amazon ECR, Elastic Beanstalk s'authentifie automatiquement auprès du registre Amazon ECR avec le profil d'instance de votre environnement. Par conséquent, vous devez autoriser vos instances à accéder aux images de votre référentiel Amazon ECR. Pour ce faire, ajoutez des autorisations au profil d'instance de votre environnement en attachant la politique EC2 ContainerRegistryReadOnly gérée par Amazon au profil d'instance. Cela fournit un accès en lecture seule à tous les référentiels Amazon ECR de votre compte. Vous avez également la possibilité de n'accéder qu'à un seul référentiel en utilisant le modèle suivant pour créer une politique personnalisée :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEbAuth",
            "Effect": "Allow",
            "Action": [
                "ecr:GetAuthorizationToken"
            ],
            "Resource": [
                "*"
            ]
```

```
        },
        {
            "Sid": "AllowPull",
            "Effect": "Allow",
            "Resource": [
                "arn:aws:ecr:us-east-2:111122223333:repository/repository-name"
            ],
            "Action": [
                "ecr:GetAuthorizationToken",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetRepositoryPolicy",
                "ecr:DescribeRepositories",
                "ecr:ListImages",
                "ecr:BatchGetImage"
            ]
        }
    ]
}
```

Remplacez le nom Amazon Resource Name (ARN) dans la stratégie ci-dessus par l'ARN de votre référentiel.

Vous devez spécifier les informations relatives à l'image dans votre `Dockerrun.aws.json` fichier. La configuration sera différente en fonction de la plateforme que vous utilisez.

Pour la plateforme Docker gérée par ECS, utilisez la `image` clé dans un objet de définition de conteneur :

```
"containerDefinitions": [
        {
        "name": "my-image",
        "image": "account-id.dkr.ecr.us-east-2.amazonaws.com/repository-name:latest",
```

Pour la plateforme Docker, reportez-vous à l'image par URL. L'URL se trouve dans la `Image` définition de votre `Dockerrun.aws.json` fichier :

```
  "Image": {
      "Name": "account-id.dkr.ecr.us-east-2.amazonaws.com/repository-name:latest",
      "Update": "true"
    },
```

En utilisant AWS Secrets Manager

Configurez Elastic Beanstalk pour qu'il s'authentifie auprès de votre référentiel privé avant le déploiement afin de permettre l'accès aux images de vos conteneurs.

Cette approche utilise la phase de préconstruction du processus de déploiement d'Elastic Beanstalk avec deux composants :

- ebextensions pour définir les variables d'environnement qui stockent les informations d'identification du référentiel
- scripts de crochet de plateforme à exécuter docker login avant d'extraire des images

Les scripts hook récupèrent un nom d'utilisateur et un mot de passe à partir de variables d'environnement renseignées à partir d'un AWS Secrets Manager secret unique au format JSON. Cette fonctionnalité nécessite les plateformes Docker Elastic Beanstalk et Docker gérées par ECS publiées le 13 janvier 2026 ou après cette date. Pour plus de détails, consultez la section Secrets environnementaux.

Pour configurer Elastic Beanstalk afin de s'authentifier auprès de votre dépôt privé avec AWS Secrets Manager

> **Note**
>
> Avant de continuer, assurez-vous d'avoir configuré vos informations d'identification AWS Secrets Manager et les autorisations IAM nécessaires. Pour plus de détails, reportez-vous à la section Conditions requises pour configurer les secrets en tant que variables d'environnement.

1. Créez la structure de répertoire suivante pour votre projet :

```
### .ebextensions
#   ### env.config
### .platform
#   ### confighooks
#   #   ### prebuild
#   #       ### 01login.sh
#   ### hooks
```

```
#       ### prebuild
#           ### 01login.sh
### Dockerfile
```

2. Utilisez-le AWS Secrets Managerpour enregistrer les informations d'identification de votre dépôt privé sous forme de secret au format JSON.

```
aws secretsmanager create-secret --name repo-credentials \
    --secret-string '{"username":"myuser","password":"mypassword"}'
```

3. Créez le fichier `env.config` suivant et placez-le dans le répertoire `.ebextensions` comme indiqué dans la structure de répertoire précédente. Cette configuration utilise l'espace de patte : tige de haricot élastique : application : secrets environnementaux noms avec extraction de clés JSON pour initialiser les variables d'environnement `USER` et Elastic `PASSWD` Beanstalk à partir de champs individuels du secret.

```
option_settings:
  aws:elasticbeanstalk:application:environmentsecrets:
    USER: arn:aws:secretsmanager:us-east-1:111122223333:secret:repo-credentials-
AbCd12:username
    PASSWD: arn:aws:secretsmanager:us-east-1:111122223333:secret:repo-credentials-
AbCd12:password
```

4. Créez le fichier de `01login.sh` script suivant et placez-le aux emplacements suivants (également indiqués dans la structure de répertoire précédente) :

   - `.platform/confighooks/prebuild/01login.sh`
   - `.platform/hooks/prebuild/01login.sh`

```
#!/bin/bash
echo $PASSWD | docker login -u $USER --password-stdin
```

   Le `01login.sh` script utilise les variables d'environnement configurées à l'étape 3 et transmet le mot de passe à docker login via`stdin`. Pour plus d'informations sur l'authentification Docker, consultez la section Docker login dans la documentation Docker.

> **Remarques**
>
> - La plate-forme Docker gérée par ECS utilise la syntaxe native ECS pour référencer les secrets. Pour plus d'informations, consultez la section Transmettre les secrets de Secrets Manager via les variables d'environnement Amazon ECS dans le manuel Amazon Elastic Container Service Developer Guide.
> - Pour plus d'informations sur les hooks de plateforme, consultez la section Extension the section called “Hooks de plateforme” des plateformes Linux Elastic Beanstalk.

Une fois l'authentification configurée, Elastic Beanstalk peut extraire et déployer des images depuis votre référentiel privé.

Utilisation du fichier **`Dockerrun.aws.json`**

Cette section décrit une autre approche pour authentifier Elastic Beanstalk auprès d'un référentiel privé. Avec cette approche, vous générez un fichier d'authentification avec la commande Docker, puis téléchargez le fichier d'authentification dans un compartiment Amazon S3. Vous devez également inclure les informations du compartiment dans votre fichier `Dockerrun.aws.json`.

Pour générer un fichier d'authentification et le fournir à Elastic Beanstalk

1. Générez un fichier d'authentification avec la commande docker login. Pour les référentiels sur Docker Hub, exécutez docker login:

   ```
   $ docker login
   ```

   Pour d'autres registres, incluez l'URL du serveur de registre :

   ```
   $ docker login registry-server-url
   ```

   > **Note**
   >
   > Si votre environnement Elastic Beanstalk utilise la version de plateforme Docker AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations dans the section called “Configuration Docker sur l'AMI Amazon Linux (antérieure à Amazon Linux 2)”.

Pour plus d'informations sur le fichier d'authentification, consultez Store images on Docker Hub et docker login sur le site web de Docker.

2. Chargez une copie du fichier d'authentification nommé `.dockercfg` dans un compartiment Amazon S3 sécurisé.

   - Le compartiment Amazon S3 doit être hébergé dans le même Région AWS environnement que celui qui l'utilise. Elastic Beanstalk ne peut pas télécharger de fichiers à partir d'un compartiment Amazon S3 hébergé dans d'autres régions.
   - Accordez des autorisations pour l'opération `s3:GetObject` au rôle IAM dans le profil d'instance. Pour plus d'informations, consultez Gestion des profils d'instance Elastic Beanstalk.

3. Incluez les informations sur le compartiment Amazon S3 dans le paramètre `Authentication` de votre fichier `Dockerrun.aws.json`.

   L'exemple suivant décrit comment utiliser un fichier d'authentification nommé `mydockercfg` dans un compartiment nommé `amzn-s3-demo-bucket` afin d'utiliser une image privée dans un registre tiers. Pour le numéro de version correct pour`AWSEBDockerrunVersion`, reportez-vous à la note qui suit l'exemple.

```
{
  "AWSEBDockerrunVersion": "version-no",
  "Authentication": {
    "Bucket": "amzn-s3-demo-bucket",
    "Key": "mydockercfg"
  },
  "Image": {
    "Name": "quay.io/johndoe/private-image",
    "Update": "true"
  },
  "Ports": [
    {
      "ContainerPort": "1234"
    }
  ],
  "Volumes": [
    {
      "HostDirectory": "/var/app/mydb",
      "ContainerDirectory": "/etc/mysql"
    }
  ],
```

```
  "Logging": "/var/log/nginx"
}
```

> **Versions `Dockerrun.aws.json`**
>
> Le paramètre `AWSEBDockerrunVersion` indique la version du fichier `Dockerrun.aws.json`.
>
> - Le Docker AL2 et AL2023 les plateformes utilisent les versions suivantes du fichier.
>   - `Dockerrun.aws.json v3`— environnements utilisant Docker Compose.
>   - `Dockerrun.aws.json v1`— environnements qui n'utilisent pas Docker Compose.
> - ECS exécuté sur Amazon Linux 2 et ECS exécuté sur Amazon AL2023 utilisent le `Dockerrun.aws.json v2` fichier. La plate-forme retirée ECS, l'AMI Multicontainer Docker Amazon Linux (AL1), utilisait également cette même version.

Une fois qu'Elastic Beanstalk peut s'authentifier auprès du registre en ligne qui héberge le référentiel privé, vos images peuvent être déployées et extraites.

Utilisation d'images provenant d'Amazon ECR Public

Amazon ECR Public est un registre de conteneurs public qui héberge des images Docker. Bien que les référentiels publics Amazon ECR soient accessibles au public, l'authentification fournit des limites de débit plus élevées et une meilleure fiabilité pour vos déploiements.

> **Note**
>
> L'authentification publique Amazon ECR n'est pas prise en charge dans les régions (`cn-*`) et les AWS GovCloud régions (`us-gov-*`) de Chine. Dans ces régions, Elastic Beanstalk utilisera des extractions non authentifiées.

Pour activer l'authentification Amazon ECR Public, ajoutez les autorisations suivantes au profil d'instance de votre environnement. Pour plus d'informations sur l'authentification Amazon ECR Public, consultez Authentification du registre dans Amazon ECR public dans le Guide de l'utilisateur public d'Amazon Elastic Container Registry :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
       {
          "Effect": "Allow",
          "Action": [
             "ecr-public:GetAuthorizationToken",
             "sts:GetServiceBearerToken"
          ],
          "Resource": "*"
       }
    ]
}
```

Une fois ces autorisations associées à votre profil d'instance, Elastic Beanstalk s'authentifie automatiquement auprès des registres publics Amazon ECR. Vous pouvez référencer des images Amazon ECR Public en utilisant le `public.ecr.aws/`*`registry-alias`*`/`*`repository-name:tag` format standard de votre `Dockerrun.aws.json` fichier ou Dockerfile.

Configuration des environnements Elastic Beanstalk Docker

Ce chapitre explique des informations de configuration supplémentaires pour toutes les branches de plate-forme Docker prises en charge, y compris la branche de plate-forme Docker gérée par ECS. À moins qu'une branche de plate-forme ou un composant de branche de plate-forme spécifique ne soit identifié dans une section, cela s'applique à tous les environnements qui exécutent des plateformes Docker gérées par Docker et ECS prises en charge.

> **Note**
>
> Si votre environnement Elastic Beanstalk utilise une version de plateforme Docker AMI Amazon Linux (antérieure à Amazon Linux 2), lisez impérativement les informations supplémentaires dans the section called “Configuration Docker sur l'AMI Amazon Linux (antérieure à Amazon Linux 2)”.

Sections

- Configuration des logiciels dans les environnements Docker
- Référencement de variables d'environnement dans les conteneurs
- Utilisation de la fonction d'interpolation pour les variables d'environnement avec Docker Compose
- Génération de journaux pour des rapports de santé améliorés avec Docker Compose
- Journalisation personnalisée des conteneurs Docker avec Docker Compose
- Images Docker
- Configuration des mises à jour gérées pour les environnements Docker
- Espaces de noms de la configuration Python
- Configuration Docker sur l'AMI Amazon Linux (antérieure à Amazon Linux 2)

Configuration des logiciels dans les environnements Docker

Vous pouvez utiliser la console Elastic Beanstalk pour configurer le logiciel s'exécutant sur les instances de votre environnement.

Pour configurer votre environnement Docker dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Effectuez les modifications de configuration nécessaires.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Pour de plus amples informations sur la configuration des paramètres logiciels dans n'importe quel environnement, veuillez consulter the section called “Variables d'environnement et paramètres logiciels”. Les sections suivantes couvrent des informations spécifiques à Docker.

Options du conteneur

La section Options du conteneur contient des options spécifiques à la plateforme. Pour les environnements Docker, elles vous permettent de choisir si votre environnement inclut ou non le serveur proxy Nginx.

Environnements avec Docker Compose

Si vous gérez votre environnement Docker avec Docker Compose, Elastic Beanstalk suppose que vous exécutez un serveur proxy en tant que conteneur. Par conséquent, sa valeur par défaut est None (Aucun) pour le paramètre du Proxy server (Serveur proxy) et Elastic Beanstalk ne fournit pas de configuration NGINX.

> **Note**
>
> Même si vous sélectionnez NGINX comme serveur proxy, ce paramètre est ignoré dans un environnement avec Docker Compose. Par défaut, le paramètre Proxy server (Serveur proxy) est toujours None (Aucun).

Étant donné que le proxy de serveur web NGINX est désactivé pour la plateforme Docker sur Amazon Linux 2 avec Docker Compose, vous devez suivre les instructions pour générer des journaux afin de profiter de rapports d'état améliorés. Pour de plus amples informations, veuillez consulter Génération de journaux pour des rapports de santé améliorés avec Docker Compose.

Propriétés de l'environnement (variables d'environnement)

Vous pouvez utiliser les propriétés d'environnement (également appelées variables d'environnement) pour transmettre des valeurs, telles que des points de terminaison, des paramètres de débogage et d'autres informations à votre application. La section Variables d'environnement de la console vous permet de spécifier des variables d'environnement sur les instances EC2 qui exécutent votre application. Les variables d'environnement sont transmises sous forme de paires clé-valeur à l'application.

Votre code d'application s'exécutant dans un conteneur peut faire référence à une variable d'environnement par son nom et lire sa valeur. Le code source qui lit ces variables d'environnement varie selon le langage de programmation. Pour obtenir des instructions pour lire les valeurs des variables d'environnement dans les langages de programmation pris en charge par les plateformes

gérées par Elastic Beanstalk, veuillez consulter la rubrique correspondante de chaque plateforme. Pour obtenir la liste des liens vers ces rubriques, veuillez consulter the section called “Variables d'environnement et paramètres logiciels”.

Secrets et paramètres des variables d'environnement Elastic Beanstalk

Elastic Beanstalk permet de AWS Systems Manager référencer et de stocker des données dans AWS Secrets Manager des variables d'environnement. Il s'agit d'une option sécurisée permettant à votre application d'accéder de manière native aux secrets et aux paramètres stockés par ces services sans avoir à gérer les appels d'API qui leur sont adressés. Vos plateformes Docker Elastic Beanstalk et Docker gérées par ECS doivent être une version publiée le 26 mars 2025 ou après cette date pour prendre en charge cette fonctionnalité. Pour plus d'informations sur l'utilisation de variables d'environnement pour référencer des secrets, consultezExtraction de secrets et de paramètres dans les variables d'environnement Elastic Beanstalk.

Environnements avec Docker Compose

Si vous gérez votre environnement Docker avec Docker Compose, vous devez effectuer une configuration supplémentaire pour récupérer les variables d'environnement dans les conteneurs. Pour que les exécutables lancés dans votre conteneur puissent accéder à ces variables d'environnement, vous devez les référencer dans le fichier `docker-compose.yml`. Pour plus d'informations, veuillez consulter Référencement de variables d'environnement dans les conteneurs.

Référencement de variables d'environnement dans les conteneurs

Si vous utilisez l'outil Docker Compose sur la plateforme Docker Amazon Linux 2, Elastic Beanstalk génère un fichier d'environnement Docker Compose appelé `.env` dans le répertoire racine de votre projet d'application. Ce fichier stocke les variables d'environnement que vous avez configurées pour Elastic Beanstalk.

> **Note**
>
> Si vous incluez un fichier `.env` dans le bundle de fichiers de votre application, Elastic Beanstalk ne générera pas de fichier `.env`.

Pour qu'un conteneur référence les variables d'environnement que vous définissez dans Elastic Beanstalk, vous devez suivre l'une de ces approches de configuration, ou les deux.

- Ajoutez le fichier `.env` généré par Elastic Beanstalk à l'option de configuration `env_file` dans le fichier `docker-compose.yml`.
- Définissez directement les variables d'environnement dans le fichier `docker-compose.yml`.

Les fichiers suivants fournissent un exemple. L'exemple de fichier `docker-compose.yml` illustre les deux approches.

- Si vous définissez les propriétés d'environnement `DEBUG_LEVEL=1` et `LOG_LEVEL=error`, Elastic Beanstalk génère le fichier `.env` suivant pour vous :

```
DEBUG_LEVEL=1
LOG_LEVEL=error
```

- Dans ce fichier `docker-compose.yml`, l'option de configuration `env_file` pointe vers le fichier `.env`, et elle définit également la variable d'environnement `DEBUG=1` directement dans le fichier `docker-compose.yml`.

```
services:
  web:
    build: .
    environment:
      - DEBUG=1
    env_file:
      - .env
```

> **Remarques**
>
> - Si vous définissez la même variable d'environnement dans les deux fichiers, la variable définie dans le fichier `docker-compose.yml` a une priorité plus élevée que la variable définie dans le fichier `.env`.
> - Veillez à ne pas laisser d'espace entre le signe égal (=) et la valeur attribuée à votre variable, afin d'empêcher l'ajout d'espaces à la chaîne.

Pour en savoir plus sur les variables d'environnement dans Docker Compose, veuillez consulter Environment variables in Compose

Utilisation de la fonction d'interpolation pour les variables d'environnement avec Docker Compose

À compter de la version de plateforme du 28 juillet 2023, la branche de plateforme Docker Amazon Linux 2 propose la fonctionnalité d'interpolation de Docker Compose. Grâce à cette fonctionnalité, les valeurs d'un fichier Compose peuvent être définies par des variables et interpolées au moment de l'exécution. Pour plus d'informations sur cette fonctionnalité, veuillez consulter la rubrique Interpolation (français non garanti) sur le site Web de documentation Docker.

> **Important**
>
> Si vous voulez utiliser cette fonctionnalité avec vos applications, sachez que vous devez mettre en œuvre une démarche qui utilise des hooks de plateforme.
> Cela est nécessaire en raison d'une atténuation que nous avons mise en œuvre dans le moteur de la plateforme. Cette atténuation garantit la rétrocompatibilité pour les clients qui ne sont pas au courant de la nouvelle fonctionnalité d'interpolation et dont les applications existantes utilisent des variables d'environnement avec le caractère $. Le moteur de plateforme mis à jour échappe à l'interpolation par défaut en remplaçant le caractère $ par des caractères $$.

Vous trouverez ci-dessous un exemple de script de hook de plateforme que vous pouvez configurer pour permettre l'utilisation de la fonctionnalité d'interpolation.

```
#!/bin/bash

: '
example data format in .env file
key1=value1
key2=value2
'
envfile="/var/app/staging/.env"
tempfile=$(mktemp)

while IFS= read -r line; do
  # split each env var string at '='
  split_str=(${line//=/ })
  if [ ${#split_str[@]} -eq 2 ]; then
    # replace '$$' with '$'
```

```
    replaced_str=${split_str[1]//\$\$/\$}
    # update the value of env var using ${replaced_str}
    line="${split_str[0]}=${replaced_str}"
  fi
  # append the updated env var to the tempfile
  echo "${line}" #"${tempfile}"
done < "${envfile}"
# replace the original .env file with the tempfile
mv "${tempfile}" "${envfile}"
```

Placez les hooks de plateforme dans ces deux répertoires :

- `.platform/confighooks/predeploy/`
- `.platform/hooks/predeploy/`

Pour plus d'informations, consultez Hooks de plateforme dans la rubrique Extension des plateformes Linux de ce guide.

Génération de journaux pour des rapports de santé améliorés avec Docker Compose

L'agent d'état fournit des métriques sur l'état du système d'exploitation et des applications pour les environnements Elastic Beanstalk. Il s'appuie sur des formats de journaux de serveur web qui transmettent les informations dans un format spécifique.

Elastic Beanstalk suppose que vous exécutez un proxy de serveur web en tant que conteneur. Par conséquent, le proxy du serveur web NGINX est désactivé pour les environnements Docker exécutant Docker Compose. Vous devez configurer votre serveur pour qu'il écrive les journaux à l'emplacement et au format utilisés par l'agent d'état Elastic Beanstalk. Cela vous permet d'utiliser pleinement les rapports d'intégrité améliorés, même si le proxy du serveur web est désactivé.

Pour obtenir des instructions sur la façon de procéder, veuillez consulter Configuration de journal de serveur web

Journalisation personnalisée des conteneurs Docker avec Docker Compose

Afin de résoudre efficacement les problèmes et de surveiller vos services conteneurisés, vous pouvez demander des journaux d'instance à Elastic Beanstalk à partir de la console de gestion de l'environnement ou de l'interface de ligne de commande (CLI) EB. Les journaux d'instance

sont composés de journaux de groupe et de journaux de fin, combinés et empaquetés pour vous permettre d'afficher les journaux et les événements récents de manière efficace et directe.

Elastic Beanstalk crée des répertoires de journaux sur l'instance de conteneur, un pour chaque service défini dans le fichier `docker-compose.yml`, à l'emplacement `/var/log/eb-docker/containers/<service name>`. Si vous utilisez la fonctionnalité Docker Compose sur la plateforme Docker Amazon Linux 2, vous pouvez monter ces répertoires à l'emplacement souhaité dans la structure du fichier de conteneur où les journaux sont écrits. Lorsque vous montez des répertoires de journaux pour écrire des données de journaux, Elastic Beanstalk peut collecter les données de journaux à partir de ces répertoires.

Pour configurer les fichiers journaux de votre service de manière à ce qu'ils soient des fichiers de queue et des journaux de bundle récupérables

1. Modifiez le fichier `docker-compose.yml`.
2. Sous la clé `volumes` de votre service, ajoutez un montage lié de la manière suivante :

   `"${EB_LOG_BASE_DIR}/<service name>:<log directory inside container>`

   Dans l'exemple de fichier `docker-compose.yml` ci-dessous :

   - `nginx-proxy` est *<service name>*
   - `/var/log/nginx` est *<log directory inside container>*

```
services:
  nginx-proxy:
    image: "nginx"
    volumes:
      - "${EB_LOG_BASE_DIR}/nginx-proxy:/var/log/nginx"
```

- Le répertoire `var/log/nginx` contient les journaux du service nginx-proxy dans le conteneur, et il sera mappé au répertoire `/var/log/eb-docker/containers/nginx-proxy` sur l'hôte.
- Tous les journaux de ce répertoire peuvent désormais être récupérés sous forme de journaux de processus et de groupe via la fonctionnalité de demande de journaux d'instance d'Elastic Beanstalk.

Remarques

- ${EB_LOG_BASE_DIR} est une variable d'environnement définie par Elastic Beanstalk avec la valeur `/var/log/eb-docker/containers`.
- Elastic Beanstalk crée automatiquement le répertoire `/var/log/eb-docker/containers/<service name>` pour chaque service dans le fichier `docker-compose.yml`.

Images Docker

Les branches Docker et ECS de la plateforme Docker gérée pour Elastic Beanstalk prennent en charge l'utilisation d'images Docker stockées dans un référentiel d'images en ligne public ou privé.

Spécifiez des images par nom dans `Dockerrun.aws.json`. Notez ces conventions :

- Les images dans les référentiels officiels sur Docker Hub utilisent un nom unique (par exemple, `ubuntu` ou `mongo`).
- Les images dans les autres référentiels sur Docker Hub sont qualifiées par un nom d'organisation (par exemple, `amazon/amazon-ecs-agent`).
- Les images dans les autres référentiels en ligne sont qualifiées par un nom de domaine (par exemple, `quay.io/assemblyline/ubuntu` ou *`account-id`*`.dkr.ecr.us-east-2.amazonaws.com/ubuntu:trusty`).

Pour les environnements utilisant la plateforme Docker uniquement, vous pouvez également créer votre propre image lors de la création d'environnement avec un fichier Dockerfile. Consultez Création d'images personnalisées avec un Dockerfile pour plus de détails. La plateforme Docker gérée par ECS ne prend pas en charge cette fonctionnalité.

Configuration des mises à jour gérées pour les environnements Docker

Avec les mises à jour gérées de la plateforme, vous pouvez configurer votre environnement afin qu'il se mette à jour automatiquement avec la dernière version d'une plateforme selon un calendrier défini.

Dans le cas des environnements Docker, vous pouvez déterminer si une mise à jour de la plateforme automatique doit être appliquée en cas de changement de version de Docker (lorsque la nouvelle version de plateforme inclut une nouvelle version de Docker). Elastic Beanstalk prend en charge

les mises à jour de plateformes gérées dans toutes les versions Docker lors de la mise à jour à partir d'un environnement exécutant une version de plateforme Docker antérieure à la version 2.9.0. Lorsqu'une nouvelle version de plateforme inclut une nouvelle version de Docker, Elastic Beanstalk incrémente le numéro de version de la mise à jour mineure. Par conséquent, pour autoriser les mises à jour de plateforme gérées sur différentes versions de Docker, activez les mises à jour gérées de la plateforme pour les mises à jour de version mineure et les correctifs. Pour empêcher les mises à jour de plateforme gérées sur différentes versions de Docker, activez les mises à jour gérées de la plateforme afin d'appliquer uniquement les mises à jour contenant des correctifs.

Par exemple, le fichier de configuration suivant active les mises à jour de plateforme gérées à 9 h UTC chaque mardi pour les mises à jour de version mineure et correctifs, permettant ainsi les mises à jour gérées sur plusieurs versions de Docker :

Example extensions .eb/ .config managed-platform-update

```
option_settings:
  aws:elasticbeanstalk:managedactions:
    ManagedActionsEnabled: true
    PreferredStartTime: "Tue:09:00"
  aws:elasticbeanstalk:managedactions:platformupdate:
    UpdateLevel: minor
```

Pour les environnements exécutant des versions de plateforme Docker 2.9.0 ou antérieures, Elastic Beanstalk n'effectue jamais de mises à jour des plateformes gérées si la nouvelle version de plateforme inclut une nouvelle version de Docker.

Espaces de noms de la configuration Python

Vous pouvez utiliser un fichier de configuration pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être spécifiques à la plate-forme ou s'appliquer à toutes les plateformes du service Elastic Beanstalk dans son ensemble. Les options de configuration sont organisées en espaces de noms.

> **Note**
>
> Ces informations s'appliquent uniquement à l'environnement Docker qui n'exécute pas Docker Compose. Cette option a un comportement différent avec les environnements Docker qui exécutent Docker Compose. Pour de plus amples informations sur les services proxy avec Docker Compose, veuillez consulter Options du conteneur.

La plateforme Docker prend en charge les options des espaces de noms suivants en plus des options prises en charge pour tous les environnements Elastic Beanstalk :

- `aws:elasticbeanstalk:environment:proxy` – Choisissez le serveur proxy pour votre environnement. Docker prend en charge l'exécution de Nginx ou aucun serveur proxy.

L'exemple de fichier de configuration suivant configure un environnement Docker de façon à ce qu'il n'exécute aucun serveur proxy.

Example.ebextensions/docker-settings.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: none
```

Configuration Docker sur l'AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Docker utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires de cette section.

Utilisation d'un fichier d'authentification pour un référentiel privé

Ces informations sont pertinentes pour vous si vous utilisez des images provenant d'un référentiel privé. En commençant par Docker version 1,7, la commande docker login a modifié le nom du fichier d'authentification et le format du fichier. Les versions de la plateforme Docker AMI Amazon Linux (antérieures à Amazon Linux 2) requièrent l'ancien fichier de configuration au format `~/.dockercfg`.

Avec Docker version 1.7 et les versions ultérieures, la commande docker login crée le fichier d'authentification dans `~/.docker/config.json` au format suivant.

```
{
    "auths":{
      "server":{
        "auth":"key"
      }
    }
  }
```

Avec Docker version 1.6.2 et les versions antérieures, la commande docker login crée le fichier d'authentification dans `~/.dockercfg` au format suivant.

```
{
    "server" :
    {
      "auth" : "auth_token",
      "email" : "email"
    }
  }
```

Pour convertir un fichier `config.json`, supprimez la clé `auths` extérieure, ajoutez une clé `email` et aplanissez le document JSON pour qu'il corresponde à l'ancien format.

Sur les versions de la plateforme Docker Amazon Linux 2, Elastic Beanstalk utilise le nom et le format de fichier d'authentification les plus récents. Si vous utilisez une version de la plateforme Docker Amazon Linux 2, vous pouvez utiliser le fichier d'authentification créé par la commande docker login sans aucune conversion.

Configuration de volumes de stockage supplémentaires

Pour améliorer les performances sur l'AMI Amazon Linux, Elastic Beanstalk configure deux volumes de stockage Amazon EBS pour les instances Amazon EC2 de votre environnement Docker. Outre le volume racine fourni pour tous les environnements Elastic Beanstalk, un deuxième volume de 12 Go nommé `xvdcz` est mis en service pour le stockage d'images sur les environnements Docker.

Si vous avez besoin de plus d'espace de stockage ou d'IOPS pour les images Docker, vous pouvez personnaliser le volume de stockage d'image à l'aide de l'option de configuration `BlockDeviceMapping` dans l'espace de noms aws:autoscaling:launchconfiguration.

Par exemple, le fichier de configuration suivant augmente la taille du volume stockage à 100 Go avec 500 IOPS provisionnées :

Example.ebextensions/blockdevice-xvdcz.config

```
option_settings:
  aws:autoscaling:launchconfiguration:
    BlockDeviceMappings: /dev/xvdcz=:100::io1:500
```

Si vous utilisez l'option `BlockDeviceMappings` pour configurer des volumes supplémentaires pour votre application, vous devez inclure un mappage pour `xvdcz` pour vous assurer de sa

création. L'exemple suivant configure deux volumes, le volume de stockage d'image `xvdcz` avec les paramètres par défaut et un volume d'application de 24 Go supplémentaires nommé `sdh` :

Example.ebextensions/blockdevice-sdh.config

```
option_settings:
  aws:autoscaling:launchconfiguration:
    BlockDeviceMappings: /dev/xvdcz=:12:true:gp2,/dev/sdh=:24
```

> **Note**
>
> Lorsque vous modifiez les paramètres dans cet espace de noms, Elastic Beanstalk remplace toutes les instances de votre environnement par des instances exécutant la nouvelle configuration. Consultez Configuration changes pour plus de détails.

Plateformes traditionnelles

Ce chapitre répertorie le contenu lié aux anciennes plateformes Docker qui ne sont plus prises en charge par AWS Elastic Beanstalk. Les sujets répertoriés ici restent dans ce document à titre de référence pour tous les clients qui utilisaient ces fonctionnalités ou composants avant leur départ à la retraite.

Rubriques

- Migration vers Elastic Beanstalk Docker exécuté sur Amazon Linux 2 depuis un Docker multi-conteneurs exécuté sur Amazon Linux
- GlassFish Conteneurs Docker préconfigurés sur Elastic Beanstalk

Migration vers Elastic Beanstalk Docker exécuté sur Amazon Linux 2 depuis un Docker multi-conteneurs exécuté sur Amazon Linux

Avant la sortie de la branche de plateforme ECS s'exécutant sur Amazon Linux 2 64 bits, Elastic Beanstalk a proposé un autre processus de migration vers Amazon Linux 2 pour les clients disposant d'environnements basés sur la branche de plateforme Docker multiconteneurs s'exécutant sur Amazon Linux 64 bits. Cette rubrique décrit ce processus de migration et reste dans ce document comme référence pour tous les clients ayant terminé ce processus de migration.

Nous recommandons désormais aux clients disposant d'environnements basés sur la branche de plateforme Docker multiconteneurs s'exécutant sur Amazon Linux 64 bits de migrer vers la branche de plateforme ECS s'exécutant sur Amazon Linux 2 64 bits. Contrairement au processus de migration alternatif, cette approche continue d'utiliser Amazon ECS pour coordonner les déploiements de conteneurs vers des environnements Docker gérés par ECS. Cet aspect permet une approche plus simple. Aucune modification du code source n'est requise, et le même `Dockerrun.aws.json` v2 est pris en charge. Pour de plus amples informations, veuillez consulter Migration de votre application Elastic Beanstalk depuis AL1 Docker multi-conteneurs géré par ECS vers ECS sur Amazon Linux 2023.

Migration héritée de Docker multiconteneurs sur Amazon Linux vers la branche de plateforme Docker Amazon Linux 2

Vous pouvez migrer vos applications qui s'exécutent sur la plateforme Docker multiconteneurs sur l'AMI Amazon Linux vers la plateforme Docker Amazon Linux 2. La plateforme Docker multiconteneurs sur l'AMI Amazon Linux nécessite de spécifier des images d'application prédéfinies à exécuter en tant que conteneurs. Après la migration, cette limitation ne s'applique plus, car la plateforme Docker Amazon Linux 2 permet également à Elastic Beanstalk de créer vos images de conteneur pendant le déploiement. Vos applications continueront à s'exécuter dans des environnements multiconteneurs tout en profitant des avantages supplémentaires que procure l'outil Docker Compose.

Docker Compose est un outil qui permet de définir et d'exécuter des applications Docker multiconteneurs. Pour en savoir plus sur Docker Compose et savoir comment l'installer, veuillez consulter les sites Docker Overview of Docker Compose et Install Docker Compose.

le fichier **`docker-compose.yml`** ;

L'outil Docker Compose utilise le fichier `docker-compose.yml` pour la configuration de vos services d'application. Ce fichier remplace votre fichier `Dockerrun.aws.json v2` dans votre répertoire de projet d'application et dans le bundle de fichiers source de l'application. Vous créez le fichier `docker-compose.yml` manuellement et il s'avérera utile de référencer votre fichier `Dockerrun.aws.json v2` pour la plupart des valeurs de paramètre.

Voici un exemple de fichier `docker-compose.yml` et le fichier `Dockerrun.aws.json v2` correspondant pour la même application. Pour de plus amples informations sur le fichier `docker-compose.yml`, veuillez consulter Compose file reference. Pour de plus amples informations sur le fichier `Dockerrun.aws.json v2`, veuillez consulter `Dockerrun.aws.json v2`.

docker-compose.yml

```
version: '2.4'
services:
  php-app:
    image: "php:fpm"
    volumes:
      - "./php-app:/var/www/html:ro
"
      - "${EB_LOG_BASE_DIR}/php-app
:/var/log/sample-app"
    mem_limit: 128m
    environment:
      Container: PHP
  nginx-proxy:
    image: "nginx"
    ports:
      - "80:80"
    volumes:
      - "./php-app:/var/www/html:ro
"
      - "./proxy/conf.d:/etc/nginx/
conf.d:ro"
      - "${EB_LOG_BASE_DIR}/nginx-p
roxy:/var/log/nginx"
    mem_limit: 128m
    links:
      - php-app
```

Dockerrun.aws.json v2

```
{
  "AWSEBDockerrunVersion": 2,
  "volumes": [
    {
      "name": "php-app",
      "host": {
        "sourcePath": "/var/app/
current/php-app"
      }
    },
    {
      "name": "nginx-proxy-conf",
      "host": {
        "sourcePath": "/var/app/
current/proxy/conf.d"
      }
    }
  ],
  "containerDefinitions": [
    {
      "name": "php-app",
      "image": "php:fpm",
      "environment": [
        {
          "name": "Container",
          "value": "PHP"
        }
      ],
      "essential": true,
      "memory": 128,
      "mountPoints": [
        {
          "sourceVolume": "php-app"
,
          "containerPath": "/var/www
/html",
          "readOnly": true
        }
      ]
    },
```

docker-compose.yml	Dockerrun.aws.json v2

```
    {
      "name": "nginx-proxy",
      "image": "nginx",
      "essential": true,
      "memory": 128,
      "portMappings": [
        {
          "hostPort": 80,
          "containerPort": 80
        }
      ],
      "links": [
        "php-app"
      ],
      "mountPoints": [
        {
          "sourceVolume": "php-app"
,
          "containerPath": "/var/www
/html",
          "readOnly": true
        },
        {
          "sourceVolume": "nginx-pr
oxy-conf",
          "containerPath": "/etc/ngi
nx/conf.d",
          "readOnly": true
        },
        {
          "sourceVolume": "awseb-lo
gs-nginx-proxy",
          "containerPath": "/var/log
/nginx"
        }
      ]
    }
  ]
}
```

Considérations supplémentaires sur la migration

La plateforme Docker Amazon Linux 2 et la plateforme AMI multiconteneurs Amazon Linux implémentent les propriétés d'environnement différemment. Ces deux plateformes ont également des répertoires de journaux différents créés par Elastic Beanstalk pour chacun de leurs conteneurs. Après avoir effectué la migration à partir de la plateforme Docker multi-conteneur AMI Amazon Linux, vous devez être conscient de ces différentes implémentations pour votre nouvel environnement de plateforme Docker Amazon Linux 2.

Area	Plateforme Docker sur Amazon Linux 2 avec Docker Compose	Plateforme Docker multiconteneurs sur l'AMI Amazon Linux
Propriétés de l'environnement	Pour que vos conteneurs puissent accéder aux propriétés de l'environnement, vous devez ajouter une référence au fichier `.env` dans le fichier `docker-compose.yml` . Elastic Beanstalk génère le fichier `.env`, répertoriant chaque propriété en tant que variable d'environnement. Pour plus d'informations, consultez Référencement de variables d'environnement dans les conteneurs.	Elastic Beanstalk peut transmettre directement les propriétés de l'environnement au conteneur. Votre code qui s'exécute dans le conteneur peut accéder à ces propriétés en tant que variables d'environnement sans configuration supplémentaire.
Répertoires de journaux	Pour chaque conteneur, Elastic Beanstalk crée un répertoire de journal appelé `/var/log/eb-docker/containers/` *`<service name>`* (ou `${EB_LOG_BASE_DIR}/<service name>`). Pour plus d'informations, consultez Journalisation personnalisée des conteneurs Docker avec Docker Compose.	Pour chaque conteneur, Elastic Beanstalk crée un répertoire de journal appelé `/var/log/containers/` *`<containername>`* . Pour de plus amples informations, veuillez consulter le champ `mountPoints` dans Format des définitions de conteneur.

Etapes de la migration

Pour migrer vers la plateforme Docker Amazon Linux 2

1. Créez le fichier `docker-compose.yml` de votre application, en fonction de son fichier `Dockerrun.aws.json` v2 existant. Pour de plus amples informations, veuillez consulter la section ci-dessu le fichier `docker-compose.yml` ;.
2. Dans le répertoire racine de votre dossier de projet d'application, remplacez le fichier `Dockerrun.aws.json` v2 par le fichier `docker-compose.yml` que vous venez de créer.

   La structure de votre répertoire doit être la suivante.

   ```
   ~/myApplication
   |-- docker-compose.yml
   |-- .ebextensions
   |-- php-app
   |-- proxy
   ```

3. Utilisez la commande eb init pour configurer votre répertoire local pour le déploiement vers Elastic Beanstalk.

   ```
   ~/myApplication$ eb init -p docker application-name
   ```

4. Utilisez la commande eb create pour créer un environnement et déployer votre image Docker.

   ```
   ~/myApplication$ eb create environment-name
   ```

5. Si votre application est une application web, après le lancement de votre environnement, utilisez la commande eb open pour l'afficher dans un navigateur web.

   ```
   ~/myApplication$ eb open environment-name
   ```

6. Vous pouvez afficher l'état de votre environnement nouvellement créé à l'aide de la commande eb status.

   ```
   ~/myApplication$ eb status environment-name
   ```

GlassFish Conteneurs Docker préconfigurés sur Elastic Beanstalk

> **Note**
>
> Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

La branche de la GlassFish plateforme Docker préconfigurée qui s'exécute sur l'AMI Amazon Linux (AL1) n'est plus prise en charge. Pour migrer votre GlassFish application vers une plateforme Amazon Linux 2023 prise en charge, déployez GlassFish et votre code d'application sur une image Docker Amazon Linux 2023. Pour de plus amples informations, veuillez consulter la rubrique the section called “Tutoriel - GlassFish sur Docker : chemin vers Amazon Linux 2023”.

Prise en main des conteneurs Docker préconfigurés – sur l'AMI Amazon Linux (antérieure à Amazon Linux 2)

Cette section décrit comment développer localement un exemple d'application, puis déployer l'application dans Elastic Beanstalk à l'aide d'un conteneur Docker préconfiguré.

Configuration de votre environnement de développement local

Pour cette présentation, nous utilisons un GlassFish exemple d'application.

Configuration de votre environnement

1. Créez un dossier pour l'exemple d'application.

```
~$ mkdir eb-preconf-example
~$ cd eb-preconf-example
```

2. Téléchargez le code de l'exemple d'application dans le nouveau dossier.

```
~$ wget https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/samples/docker-glassfish-v1.zip
~$ unzip docker-glassfish-v1.zip
~$ rm docker-glassfish-v1.zip
```

Développement et test en local

Pour développer un exemple d' GlassFish application

1. Ajoutez un `Dockerfile` au dossier racine de votre application. Dans le fichier, spécifiez l'image de base AWS Elastic Beanstalk Docker à utiliser pour exécuter votre conteneur Docker préconfiguré local. Vous déploierez ensuite votre application sur une version de plateforme Docker préconfigurée Elastic GlassFish Beanstalk. Choisissez l'image de base Docker que cette version de plateforme utilise. Pour déterminer l'image Docker actuelle de la version de plateforme, consultez la section Docker préconfiguré de la page Plateformes AWS Elastic Beanstalk prises en charge dans le guide Plateformes AWS Elastic Beanstalk .

   Example~/E/Dockerfile b-preconf-example

   ```
   # For Glassfish 5.0 Java 8
   FROM amazon/aws-eb-glassfish:5.0-al-onbuild-2.11.1
   ```

   Pour plus d'informations sur l'utilisation d'un `Dockerfile`, consultez Préparation de votre image Docker pour le déploiement sur Elastic Beanstalk.

2. Développez l'image Docker.

   ```
   ~/eb-preconf-example$ docker build -t my-app-image .
   ```

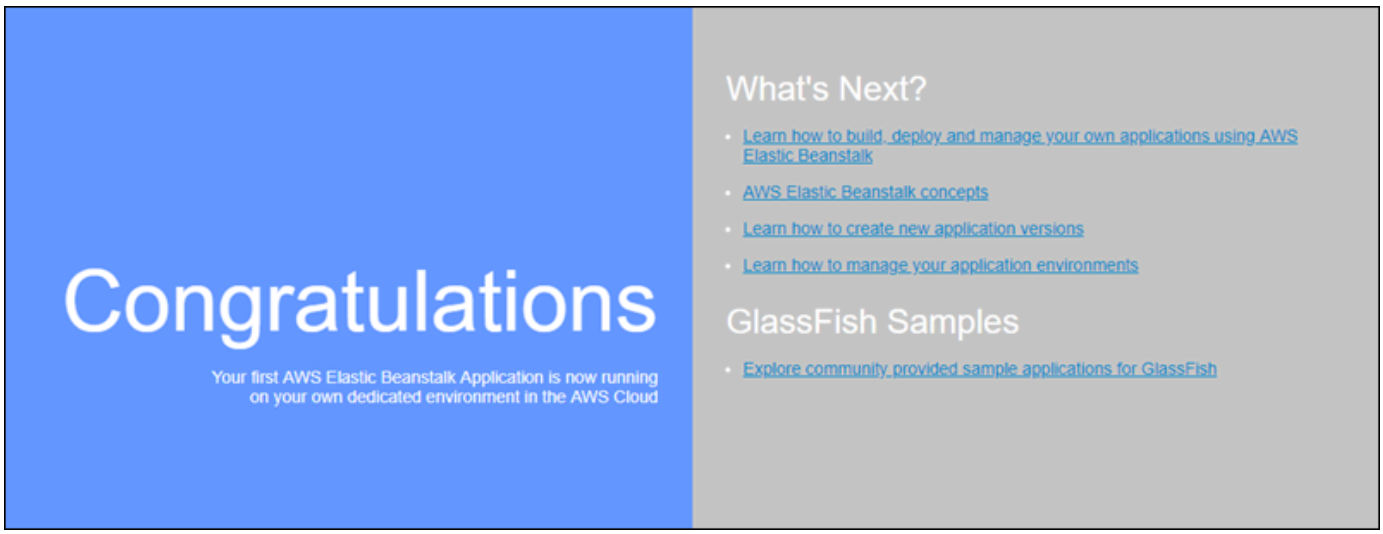
3. Exécutez le conteneur Docker à partir de l'image.

   > **Note**
   >
   > Vous devez inclure l'indicateur `-p` pour mapper le port 8080 sur le conteneur au port localhost 3000. Les conteneurs Docker Elastic Beanstalk exposent toujours l'application sur le port 8080 sur le conteneur. Les indicateurs `-it` exécutent l'image comme un processus interactif. L'indicateur `--rm` supprime le système de fichiers conteneur lorsque le conteneur s'arrête. Vous pouvez inclure le cas échéant l'indicateur `-d` pour exécuter l'image comme un démon.

   ```
   $ docker run -it --rm -p 3000:8080 my-app-image
   ```

4. Pour voir l'exemple d'application, tapez l'URL suivante dans votre navigateur web.

```
http://localhost:3000
```



Déploiement sur Elastic Beanstalk

Après avoir testé votre application, vous êtes prêt à la déployer dans Elastic Beanstalk.

Pour déployer votre application sur Elastic Beanstalk

1. Dans le dossier racine de votre application, renommez le `Dockerfile` `Dockerfile.local`. Cette étape est obligatoire pour qu'Elastic Beanstalk utilise le fichier `Dockerfile` qui contient les instructions correctes pour permettre à Elastic Beanstalk de développer une image Docker personnalisée sur chaque instance Amazon EC2 dans votre environnement Elastic Beanstalk.

   > **Note**
   >
   > Vous n'avez pas besoin d'effectuer cette étape si votre `Dockerfile` inclut des instructions qui modifient l'image Docker de base de la version de plateforme. Vous n'avez pas besoin d'utiliser un `Dockerfile` si votre `Dockerfile` contient seulement une ligne `FROM` pour spécifier l'image de base à partir de laquelle développer le conteneur. Dans ce cas, le `Dockerfile` est redondant.

2. Créez un groupe source d'application.

```
~/eb-preconf-example$ zip myapp.zip -r *
```

3. Ouvrez la console Elastic Beanstalk à l'aide de ce lien préconfiguré : console.aws.amazon.com/elasticbeanstalk/home#/newApplication? Nom de l'application = Tutoriels et type d'environnement = LoadBalanced
4. Pour Plateforme, sous Preconfigured – Docker (Préconfiguré – Docker), choisissez Glassfish.
5. Pour Code de l'application, choisissez Charger votre code puis Charger.
6. Choisissez Fichier local, Browse, puis ouvrez le groupe source d'application que vous venez de créer.
7. Choisissez Upload.
8. Choisissez Vérifier et lancer.
9. Vérifiez les paramètres disponibles et choisissez Créer une application.
10. Lorsque l'environnement est créé, vous pouvez afficher l'application déployée. Choisissez l'URL de l'environnement qui s'affiche en haut du tableau de bord de la console.

Déploiement d'une GlassFish application sur la plateforme Docker : une voie de migration vers Amazon Linux 2023

L'objectif de ce didacticiel est de fournir aux clients utilisant la GlassFish plateforme Docker préconfigurée (basée sur l'AMI Amazon Linux) un chemin de migration vers Amazon Linux 2023. Vous pouvez migrer votre GlassFish application vers Amazon Linux 2023 en GlassFish déployant le code de votre application sur une image Docker Amazon Linux 2023.

Ce didacticiel explique comment utiliser la plateforme AWS Elastic Beanstalk Docker pour déployer une application basée sur le serveur d'applications Java EE dans un GlassFish environnement Elastic Beanstalk.

Nous illustrons deux approches de création d'une image Docker :

- Simple : fournissez le code source de votre GlassFish application et laissez Elastic Beanstalk créer et exécuter une image Docker dans le cadre du provisionnement de votre environnement. Ceci est facile à configurer, au prix d'une augmentation du temps de mise en service des instances.
- Avancé – Créez une image Docker personnalisée contenant votre code d'application et vos dépendances, et fournissez-la à Elastic Beanstalk afin de l'utiliser dans votre environnement. Cette approche est légèrement plus impliquée et réduit le temps de mise en service des instances dans votre environnement.

Conditions préalables

Ce tutoriel suppose que vous ayez quelques connaissances des opérations Elastic Beanstalk de base, de l'interface de ligne de commande Elastic Beanstalk (EB CLI) et de Docker. Si ce n'est pas déjà fait, suivez les instructions dans Découvrez comment démarrer avec Elastic Beanstalk pour lancer votre premier environnement Elastic Beanstalk. Ce tutoriel utilise l'interface de ligne de commande (CLI) EB, mais vous pouvez également créer des environnements et télécharger des applications à l'aide de la console Elastic Beanstalk.

Pour suivre ce tutoriel, vous aurez également besoin des composants Docker suivants :

- Une installation locale en état de fonctionnement de Docker. Pour de plus amples informations, veuillez consulter Get Docker (Obtenir Docker) sur le site web de documentation de Docker.
- Accès à Docker Hub. Vous devez créer un ID Docker pour accéder au Docker Hub. Pour de plus amples informations, veuillez consulter Share the application (Partager l'application) sur le site Web de documentation de Docker.

Pour en savoir plus sur la configuration des environnements Docker sur des plateformes Elastic Beanstalk, consultez Préparation de votre image Docker pour le déploiement sur Elastic Beanstalk dans ce même chapitre.

Exemple simple : fournissez votre code d'application

Il s'agit d'un moyen simple de déployer votre GlassFish application. Vous fournissez au code source de votre application le fichier `Dockerfile` inclus dans ce tutoriel. Elastic Beanstalk crée une image Docker qui inclut votre application et la pile logicielle. GlassFish Ensuite, Elastic Beanstalk exécute l'image sur les instances de votre environnement.

Cette approche présente un problème, à savoir qu'Elastic Beanstalk crée l'image Docker localement chaque fois qu'il crée une instance pour votre environnement. La génération de l'image augmente le temps de mise en service de l'instance. Cet impact n'est pas limité à la création initiale de l'environnement. Il se produit également lors des actions de montée en charge.

Pour lancer un environnement avec un exemple d' GlassFish application

1. Téléchargez l'exemple `docker-glassfish-al2-v1.zip`, puis développez le fichier `.zip` dans un répertoire de votre environnement de développement.

   ```
   ~$ curl https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/samples/docker-glassfish-al2-v1.zip --output docker-glassfish-al2-v1.zip
   ```

```
~$ mkdir glassfish-example
~$ cd glassfish-example
~/glassfish-example$ unzip ../docker-glassfish-al2-v1.zip
```

La structure de votre répertoire doit être la suivante.

```
~/glassfish-example
|-- Dockerfile
|-- Dockerrun.aws.json
|-- glassfish-start.sh
|-- index.jsp
|-- META-INF
|   |-- LICENSE.txt
|   |-- MANIFEST.MF
|   `-- NOTICE.txt
|-- robots.txt
`-- WEB-INF
    `-- web.xml
```

Les fichiers suivants sont essentiels à la création et à l'exécution d'un conteneur Docker dans votre environnement :

- `Dockerfile` – Fournit des instructions que Docker utilise pour créer une image avec votre application et les dépendances requises.
- `glassfish-start.sh` – Script shell exécuté par l'image Docker pour démarrer votre application.
- `Dockerrun.aws.json`— Fournit une clé de journalisation pour inclure le journal du serveur GlassFish d'applications dans les requêtes du fichier journal. Si les GlassFish journaux ne vous intéressent pas, vous pouvez omettre ce fichier.

2. Configurez votre répertoire local pour le déploiement sur Elastic Beanstalk.

```
~/glassfish-example$ eb init -p docker glassfish-example
```

3. (Facultatif) Utilisez la commande eb local run pour créer et exécuter votre conteneur en local.

```
~/glassfish-example$ eb local run --port 8080
```

> **Note**
>
> Pour en savoir plus sur la commande eb local, consultez the section called “eb local”. La commande n'est pas prise en charge sur Windows. Vous pouvez également créer et exécuter votre conteneur avec les commandes docker build et docker run. Pour plus d'informations, consultez la documentation Docker.

4. (Facultatif) Lorsque votre conteneur est en cours d'exécution, utilisez la commande eb local open pour afficher votre application dans un navigateur web. Vous pouvez également ouvrir http://localhost:8080/ dans un navigateur web.

```
~/glassfish-example$ eb local open
```

5. Utilisez la commande eb create pour créer un environnement et déployer votre application.

```
~/glassfish-example$ eb create glassfish-example-env
```

6. Après le lancement de votre environnement, utilisez la commande eb open pour l'afficher dans un navigateur web.

```
~/glassfish-example$ eb open
```

Lorsque vous avez terminé l'exemple, arrêtez l'environnement et supprimez les ressources associées.

```
~/glassfish-example$ eb terminate --all
```

Exemple avancé : fournir une image Docker préconstruite

Il s'agit d'une méthode plus avancée pour déployer votre GlassFish application. Sur la base du premier exemple, vous créez une image Docker contenant le code de votre application et la pile GlassFish logicielle, puis vous la transférez vers Docker Hub. Après avoir effectué cette étape unique, vous pouvez lancer des environnements Elastic Beanstalk basés sur votre image personnalisée.

Lorsque vous lancez un environnement et que vous fournissez votre image Docker, les instances de votre environnement téléchargent et utilisent cette image directement et n'ont pas besoin de créer une image Docker. Par conséquent, le temps de mise en service de l'instance est réduit.

> **Remarques**
>
> - Les étapes suivantes créent une image Docker disponible publiquement.
> - Vous utiliserez les commandes Docker de votre installation Docker locale, ainsi que vos informations d'identification Docker Hub. Pour plus d'informations, consultez la section Prérequis précédente de cette rubrique.

Pour lancer un environnement avec une image Docker d' GlassFish application prédéfinie

1. Téléchargez et développez l'exemple `docker-glassfish-al2-v1.zip` comme dans l'exemple simple précédent. Si vous avez terminé cet exemple, vous pouvez utiliser le répertoire que vous possédez déjà.
2. Créez une image Docker et poussez-la vers Docker Hub. Entrez votre identifiant Docker pour vous *`docker-id`* connecter à Docker Hub.

   ```
   ~/glassfish-example$ docker build -t docker-id/beanstalk-glassfish-example:latest .
   ~/glassfish-example$ docker push docker-id/beanstalk-glassfish-example:latest
   ```

   > **Note**
   >
   > Avant de transmettre votre image, il se peut que vous deviez exécuter docker login. Vous serez invité à saisir vos informations d'identification Docker Hub si vous exécutez la commande sans paramètres.

3. Créez un répertoire supplémentaire.

   ```
   ~$ mkdir glassfish-prebuilt
   ~$ cd glassfish-prebuilt
   ```

4. Copiez l'exemple suivant dans un fichier nommé `Dockerrun.aws.json`.

Example~/glassfish-prebuilt/Dockerrun.aws.json

```
{
  "AWSEBDockerrunVersion": "1",
  "Image": {
    "Name": "docker-username/beanstalk-glassfish-example"
  },
  "Ports": [
    {
      "ContainerPort": 8080,
      "HostPort": 8080
    }
  ],
  "Logging": "/usr/local/glassfish5/glassfish/domains/domain1/logs"
}
```

5. Configurez votre répertoire local pour le déploiement sur Elastic Beanstalk.

```
~/glassfish-prebuilt$ eb init -p docker glassfish-prebuilt$
```

6. (Facultatif) Utilisez la commande eb local run pour exécuter votre conteneur localement.

```
~/glassfish-prebuilt$ eb local run --port 8080
```

7. (Facultatif) Lorsque votre conteneur est en cours d'exécution, utilisez la commande eb local open pour afficher votre application dans un navigateur web. Vous pouvez également ouvrir http://localhost:8080/ dans un navigateur web.

```
~/glassfish-prebuilt$ eb local open
```

8. Utilisez la commande eb create pour créer un environnement et déployer votre image Docker.

```
~/glassfish-prebuilt$ eb create glassfish-prebuilt-env
```

9. Après le lancement de votre environnement, utilisez la commande eb open pour l'afficher dans un navigateur web.

```
~/glassfish-prebuilt$ eb open
```

Lorsque vous avez terminé l'exemple, arrêtez l'environnement et supprimez les ressources associées.

```
~/glassfish-prebuilt$ eb terminate --all
```

Surveillance des environnements dans Elastic Beanstalk

Grâce à la surveillance de l'état de santé d'Elastic Beanstalk, vous pouvez vérifier la disponibilité des applications et créer des alertes qui s'activent lorsque les indicateurs dépassent vos seuils. Vous pouvez utiliser la surveillance de l'état d'Elastic Beanstalk à la fois dans la console et sur la ligne de commande pour suivre l'état de votre environnement.

Rubriques

- Surveillance de l'état de l'environnement dans la console AWS de gestion
- Utilisation de l'interface de ligne de commande EB pour surveiller l'intégrité de l'environnement
- Création de rapports d'intégrité de base
- Rapports et surveillance de l'état de santé améliorés dans Elastic Beanstalk
- Gestion des alarmes
- Affichage de l'historique des modifications d'un environnement Elastic Beanstalk
- Affichage du flux d'événements d'un environnement Elastic Beanstalk
- Affichage de la liste des instances de serveur et connexion à ces instances
- Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk
- Afficher les journaux de déploiement pour un environnement Elastic Beanstalk

Surveillance de l'état de l'environnement dans la console AWS de gestion

Vous pouvez accéder aux informations opérationnelles concernant votre application depuis la console Elastic Beanstalk. La console affiche l'état de votre environnement et l'intégrité de l'application en un coup de œil. Dans la page Environments (Environnements) de la console et dans la page de chaque application, les environnements de la liste sont codés par couleur pour indiquer leur état.

Pour surveiller un environnement dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

3. Dans le panneau de navigation, choisissez Surveillance.

La page Monitoring vous montre les statistiques globales sur votre environnement, telles que l'utilisation de l'UC et la latence moyenne. Outre les statistiques globales, vous pouvez afficher les graphiques de surveillance qui affichent l'utilisation des ressources sur la durée. Vous pouvez cliquer sur n'importe lequel des graphiques pour afficher des informations plus détaillées.

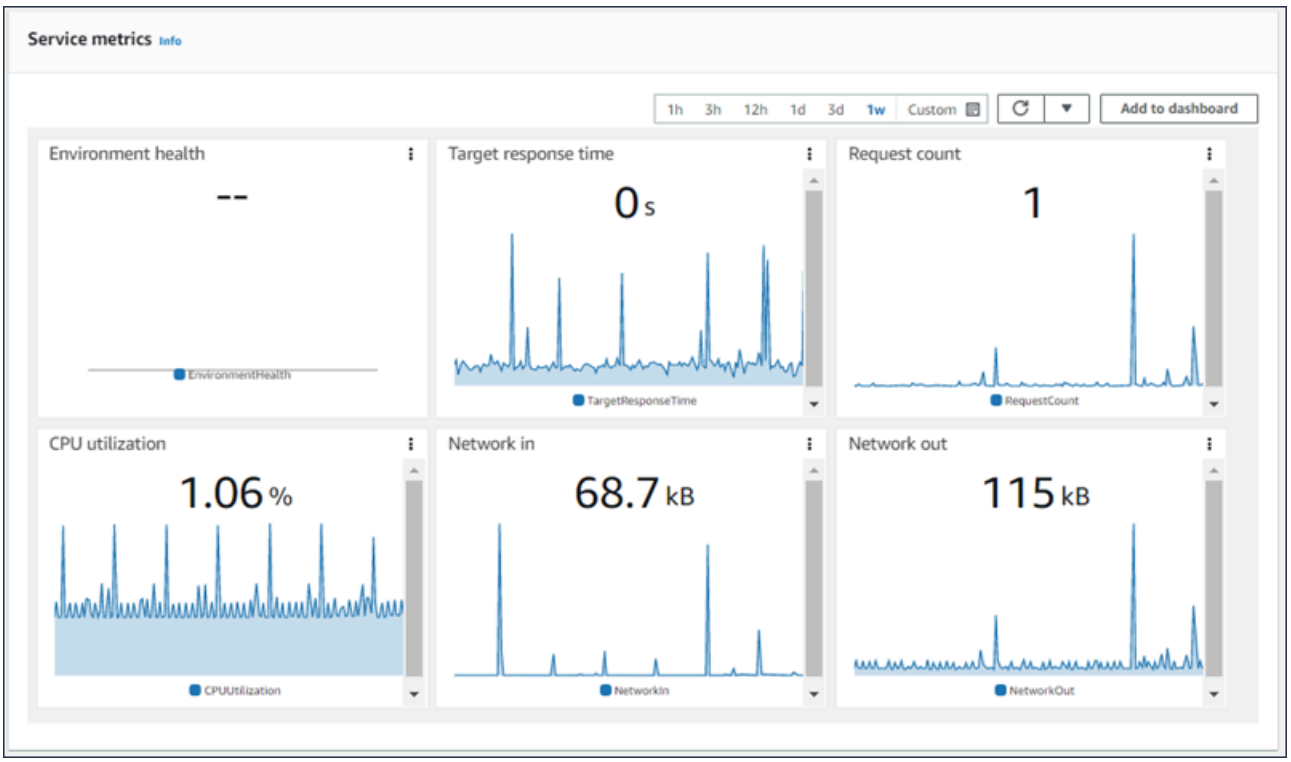
> **Note**
>
> Par défaut, seules CloudWatch les métriques de base sont activées, qui renvoient les données par périodes de cinq minutes. Vous pouvez activer des mesures plus détaillées d'une minute en modifiant CloudWatch les paramètres de configuration de votre environnement.

Graphiques de surveillance

La page Surveillance présente une vue d'ensemble des métriques liées à l'état de votre environnement. Cela inclut l'ensemble de mesures par défaut fourni par Elastic Load Balancing et Amazon EC2, ainsi que des graphiques illustrant l'évolution de l'état de l'environnement au fil du temps.

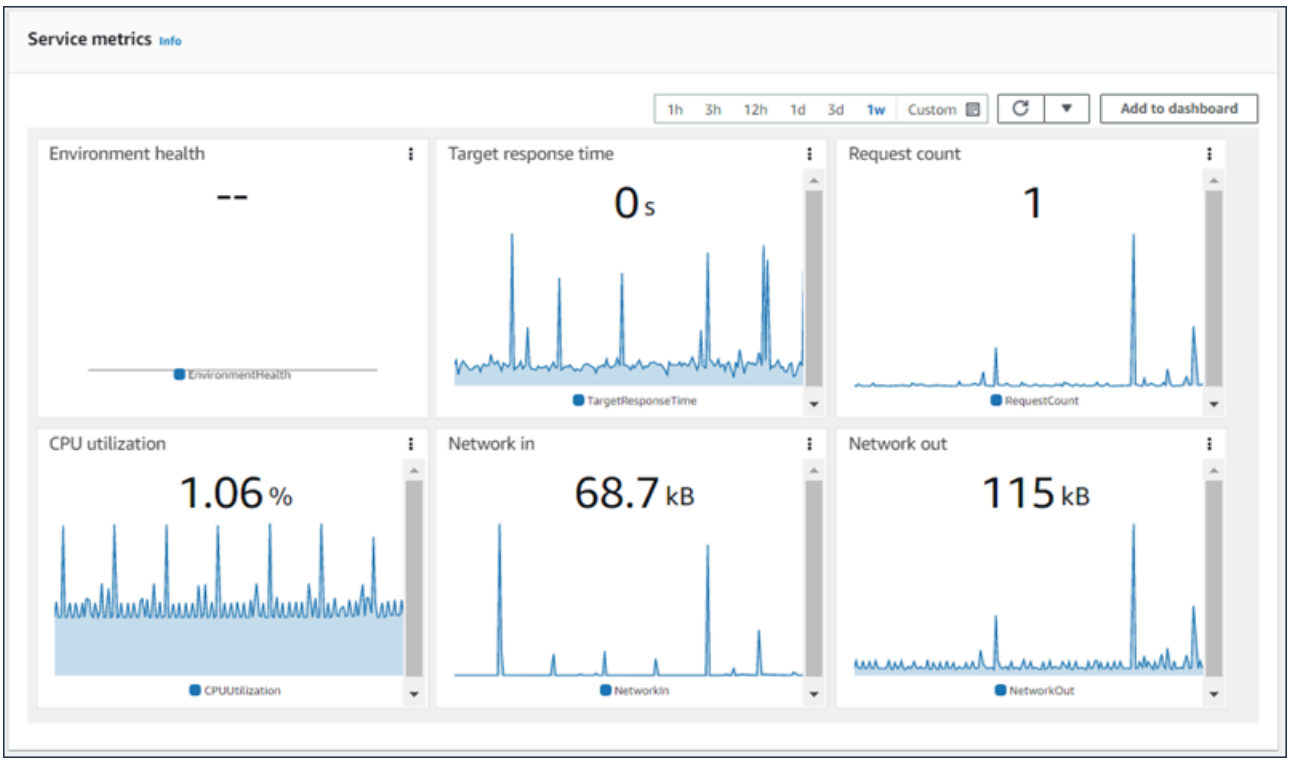
La barre située au-dessus des graphiques propose différents intervalles de temps que vous pouvez sélectionner. Par exemple, sélectionnez 1w pour afficher les informations couvrant la semaine dernière. Vous pouvez également sélectionner 3h pour afficher les informations couvrant les trois dernières heures.

Pour une plus grande variété de sélections d'intervalles de temps, choisissez Personnalisé. Vous disposez alors de deux options d'intervalle : Absolue ou Relative. L'option Absolue vous permet de spécifier une plage de dates spécifique, par exemple du 1er janvier 2023 au 30 juin 2023. L'option Relative vous permet de sélectionner un nombre entier avec une unité de temps spécifique : Minutes, Heures, Jours, Semaines ou Mois. Les exemples incluent 10 heures, 10 jours et 10 mois.



Personnalisation de la console de surveillance

Pour créer et consulter des statistiques personnalisées, vous devez utiliser Amazon CloudWatch. CloudWatch Vous pouvez ainsi créer des tableaux de bord personnalisés pour surveiller vos ressources dans une vue unique. Sélectionnez Ajouter au tableau de bord pour accéder à la CloudWatch console Amazon depuis la page de surveillance. Amazon vous CloudWatch offre la possibilité de créer un nouveau tableau de bord ou d'en sélectionner un existant. Pour plus d'informations, consultez la section Utilisation CloudWatch des tableaux de bord Amazon dans le guide de CloudWatch l'utilisateur Amazon.



Les EC2 métriques Elastic Load Balancing et Amazon sont activées pour tous les environnements.

Lorsque l'état de santé est amélioré, la EnvironmentHealth métrique est activée et un graphique est automatiquement ajouté à la console de surveillance. L'état amélioré ajoute également la page Health à la console de gestion. Pour obtenir une liste des métriques d'état amélioré disponibles, veuillez consulter Publication de métriques CloudWatch personnalisées Amazon pour un environnement.

Utilisation de l'interface de ligne de commande EB pour surveiller l'intégrité de l'environnement

L'interface de ligne de commande Elastic Beanstalk (EB CLI) est un outil de ligne de commande permettant de gérer les environnements. AWS Elastic Beanstalk Vous pouvez également utiliser l'interface de ligne de commande EB pour surveiller l'intégrité de votre environnement en temps réel et avec davantage de granularité que celle qui est actuellement disponible dans la console Elastic Beanstalk.

Après avoir installé et configuré l'EB CLI, vous pouvez lancer un nouvel environnement et y déployer votre code à l'aide de la eb create commande. Si vous avez déjà un environnement que vous avez créé dans la console Elastic Beanstalk, vous pouvez y attacher l'interface de ligne de commande EB

en exécutant eb init dans un dossier de projet et en suivant les invites (le dossier du projet peut être vide).

> **Important**
>
> Assurez-vous que vous utilisez la dernière version de l'interface de ligne de commande EB en exécutant `pip install` avec l'option `--upgrade` :
>
> ```
> $ sudo pip install --upgrade awsebcli
> ```
>
> Pour des instructions complètes d'installation de l'interface de ligne de commande EB, veuillez consulter Installation de la CLI EB avec un script de configuration (recommandé).

Pour utiliser l'interface de ligne de commande EB pour surveiller l'intégrité de votre environnement, vous devez tout d'abord configurer un dossier local du projet en exécutant eb init puis en suivant les invites. Pour obtenir des instructions complètes, veuillez consulter Configuration de l'interface de ligne de commande EB.

Si un environnement s'exécute déjà dans Elastic Beanstalk et que vous souhaitez utiliser l'interface de ligne de commande EB pour surveiller son état, procédez comme suit pour l'attacher à l'environnement existant.

Pour attacher l'interface de ligne de commande EB à un environnement existant

1. Ouvrez une terminal de ligne de commande et accédez à votre dossier utilisateur.
2. Créez et ouvrez un dossier pour votre environnement.
3. Exécutez la commande eb init, puis choisissez l'application et l'environnement dont vous souhaitez analyser l'intégrité. Si un seul environnement exécute l'application que vous avez choisie, l'interface de ligne de commande EB le sélectionne automatiquement et vous n'avez pas à choisir l'environnement, comme illustré dans l'exemple suivant.

```
~/project$ eb init
Select an application to use
1) elastic-beanstalk-example
2) [ Create new Application ]
(default is 2): 1
Select the default environment.
You can change this later by typing "eb use [environment_name]".
```

```
1) elasticBeanstalkEx2-env
2) elasticBeanstalkExa-env
(default is 1): 1
```

Pour surveillez l'intégrité en utilisant l'interface de ligne de commande EB

1. Ouvrez une ligne de commande et accédez au dossier de votre projet.
2. Exécutez la commande eb health pour afficher l'état d'intégrité des instances dans votre environnement. Dans cet exemple, il y a cinq instances en cours d'exécution dans un environnement Linux.

```
~/project $ eb health
 elasticBeanstalkExa-env                                    Ok
 2015-07-08 23:13:20
WebServer
    Ruby 2.1 (Puma)
  total      ok    warning  degraded  severe    info   pending  unknown
    5        5        0        0        0        0        0        0

  instance-id   status     cause
                                          health
    Overall     Ok
  i-d581497d    Ok
  i-d481497c    Ok
  i-136e00c0    Ok
  i-126e00c1    Ok
  i-8b2cf575    Ok

  instance-id   r/sec    %2xx   %3xx   %4xx   %5xx      p99      p90      p75
 p50     p10                                requests
    Overall     671.8   100.0    0.0    0.0    0.0    0.003    0.002    0.001
 0.001   0.000
  i-d581497d    143.0    1430      0      0      0    0.003    0.002    0.001
 0.001   0.000
  i-d481497c    128.8    1288      0      0      0    0.003    0.002    0.001
 0.001   0.000
  i-136e00c0    125.4    1254      0      0      0    0.004    0.002    0.001
 0.001   0.000
  i-126e00c1    133.4    1334      0      0      0    0.003    0.002    0.001
 0.001   0.000
```

```
  i-8b2cf575    141.2    1412      0      0      0    0.003    0.002    0.001
 0.001   0.000

  instance-id   type       az   running     load 1  load 5      user%  nice%
 system%  idle%   iowait%                              cpu
  i-d581497d    t2.micro   1a   12 mins        0.0    0.04        6.2    0.0
 1.0   92.5       0.1
  i-d481497c    t2.micro   1a   12 mins       0.01    0.09        5.9    0.0
 1.6   92.4       0.1
  i-136e00c0    t2.micro   1b   12 mins       0.15    0.07        5.5    0.0
 0.9   93.2       0.0
  i-126e00c1    t2.micro   1b   12 mins       0.17    0.14        5.7    0.0
 1.4   92.7       0.1
  i-8b2cf575    t2.micro   1c   1 hour        0.19    0.08        6.5    0.0
 1.2   92.1       0.1

  instance-id   status     id   version              ago
                                        deployments
  i-d581497d    Deployed   1    Sample Application   12 mins
  i-d481497c    Deployed   1    Sample Application   12 mins
  i-136e00c0    Deployed   1    Sample Application   12 mins
  i-126e00c1    Deployed   1    Sample Application   12 mins
  i-8b2cf575    Deployed   1    Sample Application   1 hour
```

Dans cet exemple, il y a une seule instance en cours d'exécution dans un environnement Windows.

```
~/project $ eb health
 WindowsSampleApp-env                                Ok
      2018-05-22 17:33:19
WebServer                                              IIS 10.0 running on 64bit
 Windows Server 2016/2.2.0
  total      ok    warning  degraded  severe    info   pending  unknown
    1        1        0        0        0        0        0        0

  instance-id           status     cause
                                          health
    Overall             Ok
  i-065716fba0e08a351   Ok

  instance-id           r/sec    %2xx   %3xx   %4xx   %5xx      p99      p90
 p75     p50     p10                         requests
```

```
    Overall              13.7   100.0    0.0    0.0    0.0    1.403    0.970
0.710   0.413   0.079
 i-065716fba0e08a351     2.4   100.0    0.0    0.0    0.0    1.102*   0.865
0.601   0.413   0.091

 instance-id           type       az   running     % user time    % privileged
time  % idle time                                  cpu
 i-065716fba0e08a351   t2.large   1b   4 hours             0.2
0.1         99.7

 instance-id           status     id   version              ago
                                       deployments
 i-065716fba0e08a351   Deployed   2    Sample Application   4 hours
```

Lecture du résultat

Le résultat affiche le nom de l'environnement, l'intégrité globale de l'environnement et la date actuelle en haut de l'écran.

```
elasticBeanstalkExa-env                                  Ok
 2015-07-08 23:13:20
```

Les trois lignes suivantes indiquent le type d'environnement (» WebServer » dans ce cas), la configuration (Ruby 2.1 avec Puma) et une ventilation du nombre d'instances dans chacun des sept états.

```
WebServer
 Ruby 2.1 (Puma)
  total      ok    warning  degraded  severe    info   pending  unknown
    5        5        0        0        0        0        0        0
```

Le reste du résultat est divisé en quatre sections. La première affiche l'état et la cause de l'état de l'environnement dans l'ensemble, puis pour chaque instance. L'exemple suivant montre deux instances de l'environnement avec un état `Info` et une cause indiquant qu'un déploiement a commencé.

```
  instance-id    status     cause
                                        health
    Overall      Ok
  i-d581497d     Info       Performing application deployment (running for 3 seconds)
```

```
  i-d481497c     Info       Performing application deployment (running for 3 seconds)
  i-136e00c0     Ok
  i-126e00c1     Ok
  i-8b2cf575     Ok
```

Pour de plus amples informations sur les couleurs et les états d'intégrité, veuillez consulter Couleurs et états utilisés dans les rapports d'intégrité.

La section requests (demandes) affiche des informations à partir des journaux du serveur web sur chaque instance. Dans cet exemple, chaque instance accepte les demandes normalement et il n'y a pas d'erreurs.

```
  instance-id    r/sec    %2xx   %3xx   %4xx   %5xx      p99      p90      p75      p50
    p10                                requests
    Overall      13.7    100.0    0.0    0.0    0.0    1.403    0.970    0.710   0.413
0.079
  i-d581497d     2.4     100.0    0.0    0.0    0.0    1.102*   0.865    0.601   0.413
0.091
  i-d481497c     2.7     100.0    0.0    0.0    0.0    0.842*   0.788    0.480   0.305
0.062
  i-136e00c0     4.1     100.0    0.0    0.0    0.0    1.520*   1.088    0.883   0.524
0.104
  i-126e00c1     2.2     100.0    0.0    0.0    0.0    1.334*   0.791    0.760   0.344
0.197
  i-8b2cf575     2.3     100.0    0.0    0.0    0.0    1.162*   0.867    0.698   0.477
0.076
```

La section cpu illustre les métriques du système d'exploitation pour chaque instance. La sortie diffère selon le système d'exploitation. Voici la sortie pour les environnements Linux.

```
  instance-id   type       az   running     load 1  load 5      user%  nice%  system%
idle%   iowait%                              cpu
  i-d581497d    t2.micro   1a   12 mins       0.0    0.03        0.2    0.0      0.0
99.7       0.1
  i-d481497c    t2.micro   1a   12 mins       0.0    0.03        0.3    0.0      0.0
99.7       0.0
  i-136e00c0    t2.micro   1b   12 mins       0.0    0.04        0.1    0.0      0.0
99.9       0.0
  i-126e00c1    t2.micro   1b   12 mins      0.01    0.04        0.2    0.0      0.0
99.7       0.1
  i-8b2cf575    t2.micro   1c   1 hour        0.0    0.01        0.2    0.0      0.1
99.6       0.1
```

Voici la sortie pour les environnements Windows.

```
  instance-id           type       az   running     % user time    % privileged time  %
idle time
  i-065716fba0e08a351   t2.large   1b   4 hours             0.2                 0.0
     99.8
```

Pour de plus amples informations sur les métriques du serveur et du système d'exploitation affichées, veuillez consulter Métriques des instances.

La section finale, deployments (déploiements), présente l'état de déploiement de chaque instance. En cas d'échec d'un déploiement propagé, vous pouvez utiliser l'ID, le statut et l'étiquette de version indiqués pour identifier les instances de votre environnement exécutant la mauvaise version.

```
  instance-id   status     id   version              ago
                                  deployments
  i-d581497d    Deployed   1    Sample Application   12 mins
  i-d481497c    Deployed   1    Sample Application   12 mins
  i-136e00c0    Deployed   1    Sample Application   12 mins
  i-126e00c1    Deployed   1    Sample Application   12 mins
  i-8b2cf575    Deployed   1    Sample Application   1 hour
```

Vue d'intégrité interactive

La commande eb health affiche un instantané de l'intégrité de votre environnement. Pour actualiser les informations affichées toutes les 10 secondes, utilisez l'option `--refresh`.

```
$ eb health --refresh
 elasticBeanstalkExa-env                          Ok
 2015-07-09 22:10:04 (1 secs)
WebServer
          Ruby 2.1 (Puma)
  total      ok    warning  degraded  severe    info   pending  unknown
    5        5        0        0        0        0        0        0

  instance-id   status     cause
                                      health
    Overall     Ok
  i-bb65c145    Ok         Application deployment completed 35 seconds ago and took 26
 seconds
  i-ba65c144    Ok         Application deployment completed 17 seconds ago and took 25
 seconds
```

```
 i-f6a2d525    Ok         Application deployment completed 53 seconds ago and took 26
seconds
 i-e8a2d53b    Ok         Application deployment completed 32 seconds ago and took 31
seconds
 i-e81cca40    Ok

 instance-id   r/sec    %2xx   %3xx   %4xx   %5xx      p99      p90      p75     p50
  p10                              requests
   Overall     671.8   100.0    0.0    0.0    0.0    0.003    0.002    0.001   0.001
0.000
 i-bb65c145    143.0    1430      0      0      0    0.003    0.002    0.001   0.001
0.000
 i-ba65c144    128.8    1288      0      0      0    0.003    0.002    0.001   0.001
0.000
 i-f6a2d525    125.4    1254      0      0      0    0.004    0.002    0.001   0.001
0.000
 i-e8a2d53b    133.4    1334      0      0      0    0.003    0.002    0.001   0.001
0.000
 i-e81cca40    141.2    1412      0      0      0    0.003    0.002    0.001   0.001
0.000

 instance-id   type       az   running     load 1  load 5      user%  nice%  system%
idle%   iowait%                              cpu
 i-bb65c145    t2.micro   1a   12 mins        0.0    0.03        0.2    0.0      0.0
99.7       0.1
 i-ba65c144    t2.micro   1a   12 mins        0.0    0.03        0.3    0.0      0.0
99.7       0.0
 i-f6a2d525    t2.micro   1b   12 mins        0.0    0.04        0.1    0.0      0.0
99.9       0.0
 i-e8a2d53b    t2.micro   1b   12 mins       0.01    0.04        0.2    0.0      0.0
99.7       0.1
 i-e81cca40    t2.micro   1c   1 hour         0.0    0.01        0.2    0.0      0.1
99.6       0.1

 instance-id   status     id   version              ago
                                 deployments
 i-bb65c145    Deployed   1    Sample Application   12 mins
 i-ba65c144    Deployed   1    Sample Application   12 mins
 i-f6a2d525    Deployed   1    Sample Application   12 mins
 i-e8a2d53b    Deployed   1    Sample Application   12 mins
 i-e81cca40    Deployed   1    Sample Application   1 hour

(Commands: Help,Quit, # # # #)
```

Cet exemple illustre un environnement qui a récemment été monté en puissance d'une à cinq instances. L'opération de mise à l'échelle a abouti, et toutes les instances transmettent maintenant des vérifications de l'état et sont prêtes à prendre des demandes. En mode interactif, l'état d'intégrité est mis à jour toutes les 10 secondes. Dans l'angle supérieur droit, un minuteur assure le décompte jusqu'à la prochaine mise à jour.

Dans l'angle inférieur gauche, le rapport affiche une liste d'options. Pour quitter le mode interactif, appuyez sur Q. Pour faire défiler la page, appuyez sur les touches fléchées. Pour afficher une liste de commandes supplémentaires, appuyez sur H.

Options d'affichage d'intégrité interactif

Lors de l'affichage d'intégrité de l'environnement de manière interactive, vous pouvez utiliser les touches du clavier pour ajuster la vue et dire à Elastic Beanstalk de remplacer ou de redémarrer des instances individuelles. Pour afficher la liste des commandes disponibles lors de la consultation du rapport d'intégrité en mode interactif, appuyez sur H.

```
  up,down,home,end   Scroll vertically
  left,right         Scroll horizontally
  F                  Freeze/unfreeze data
  X                  Replace instance
  B                  Reboot instance
  <,>                Move sort column left/right
  -,+                Sort order descending/ascending
  P                  Save health snapshot data file
  Z                  Toggle color/mono mode
  Q                  Quit this program

  Views
  1                  All tables/split view
  2                  Status Table
  3                  Request Summary Table
  4                  CPU%/Load Table
  H                  This help menu


(press Q or ESC to return)
```

Création de rapports d'intégrité de base

Cette rubrique décrit les fonctionnalités proposées par Elastic Beanstalk Basic Health.

AWS Elastic Beanstalk utilise des informations provenant de sources multiples pour déterminer si votre environnement est disponible et pour traiter les demandes provenant d'Internet. L'état de santé d'un environnement est représenté par l'une des quatre couleurs et s'affiche sur la page de présentation de l'environnement de la console Elastic Beanstalk. Il est également disponible depuis l'DescribeEnvironmentsAPI et en appelant eb status avec l'EB CLI.

Le système de création de rapports de base sur l'état fournit des informations sur l'état des instances dans un environnement Elastic Beanstalk en fonction des vérifications de l'état effectuées par Elastic Load Balancing pour les environnements à charge équilibrée ou Amazon Elastic Compute Cloud pour des environnements instance unique.

Outre le contrôle de l'état de vos EC2 instances, Elastic Beanstalk surveille également les autres ressources de votre environnement et signale les ressources manquantes ou mal configurées qui peuvent rendre votre environnement inaccessible aux utilisateurs.

Les métriques collectées par les ressources de votre environnement sont publiées sur Amazon toutes CloudWatch les cinq minutes. Cela inclut les métriques du système d'exploitation provenant d' EC2Elastic Load Balancing et les métriques des requêtes. Vous pouvez consulter des graphiques basés sur ces CloudWatch mesures sur la page de surveillance de la console d'environnement. Pour l'intégrité de base, ces métriques ne sont pas utilisées pour déterminer une intégrité de l'environnement.

Rubriques

- Couleurs de l'intégrité
- Vérifications de l'état Elastic Load Balancing
- Vérifications de l'état d'un environnement à instance unique et d'un environnement de travail
- Contrôles supplémentaires
- CloudWatch Métriques Amazon

Couleurs de l'intégrité

Elastic Beanstalk signale l'état d'un environnement de serveur web en fonction de la façon dont l'application qui s'y exécute répond à la vérification de l'état. Elastic Beanstalk utilise l'une des quatre couleurs pour décrire l'état, comme illustré dans le tableau suivant :

Couleur	Description
Gris	Votre environnement est en cours de mise à jour.
Vert	Votre environnement a passé avec succès la vérification de l'état la plus récente. Au moins une instance dans votre environnement est disponible et accepte des demandes.
Jaune	Votre environnement a échoué à une ou plusieurs vérifications de l'état. Certaines demandes à votre environnement sont en cours d'échec.
Rouge	Votre environnement a échoué à au moins trois vérifications de l'état, ou une ressource d'environnement est devenue indisponible. Les demandes échouent systématiquement.

Ces descriptions s'appliquent uniquement aux environnements utilisant la création de rapports d'intégrité de base. Veuillez consulter Couleurs et états utilisés dans les rapports d'intégrité pour obtenir des détails sur l'état améliore.

Vérifications de l'état Elastic Load Balancing

Dans un environnement à charge équilibrée, Elastic Load Balancing envoie une demande à chaque instance dans un environnement toutes les 10 secondes afin de confirmer que les instances sont saines. Par défaut, l'équilibreur de charge est configuré pour ouvrir une connexion TCP sur le port 80. Si l'instance reconnaît la connexion, elle est considérée comme saine.

Vous pouvez choisir remplacer ce paramètre en spécifiant une ressource existante dans votre application. Si vous spécifiez un chemin d'accès, tel que `/health`, l'URL de vérification de l'état est définie sur `HTTP:80/health`. L'URL de vérification de l'état doit être définie sur un chemin d'accès qui est toujours desservi par votre application. Si elle est définie sur une page statique qui est desservie ou mise en cache par le serveur web devant votre application, les vérifications de l'état ne révéleront pas de problèmes avec le serveur d'applications ou le conteneur web. Pour obtenir des instructions sur la modification de votre URL de vérification de l'état, consultez Surveillance de l'état.

Si une URL de vérification de l'état est configurée, Elastic Load Balancing attend une demande GET qu'il soumet pour renvoyer une réponse `200 OK`. L'application échoue à la vérification de l'état en cas de défaut de réponse dans les 5 secondes, ou si la réponse est un code d'état autre que HTTP. Après 5 échecs consécutifs de vérification de l'état, Elastic Load Balancing suspend l'instance.

Pour de plus amples informations sur les vérifications de l'état Elastic Load Balancing, veuillez consulter Vérification de l'état dans le Guide de l'utilisateur Elastic Load Balancing.

> **Note**
>
> La configuration d'une URL de vérification de l'état ne modifie pas le comportement de vérification de l'état du groupe Auto Scaling d'un environnement. Une instance défectueuse est supprimée de l'équilibreur de charge, mais n'est pas automatiquement remplacée par Amazon EC2 Auto Scaling, sauf si vous configurez Amazon EC2 Auto Scaling pour utiliser le bilan de santé d'Elastic Load Balancing comme base pour le remplacement des instances. Pour configurer Amazon EC2 Auto Scaling afin de remplacer les instances qui échouent à un test de santé d'Elastic Load Balancing, consultezParamètre de vérification de l'état Auto Scaling pour votre environnement Elastic Beanstalk.

Vérifications de l'état d'un environnement à instance unique et d'un environnement de travail

Dans un environnement à instance unique ou au niveau du travailleur, Elastic Beanstalk détermine l'état de santé de l'instance en surveillant le statut de son instance Amazon. EC2 Les paramètres d'intégrité d'Elastic Load Balancing, y compris le contrôle de santé HTTP URLs, ne peuvent pas être utilisés dans ces types d'environnement.

Pour plus d'informations sur les contrôles de statut des EC2 instances Amazon, consultez la section Monitoring Instances with Status Checks dans le guide de EC2 l'utilisateur Amazon.

Contrôles supplémentaires

Outre les vérifications de l'état Elastic Load Balancing, Elastic Beanstalk surveille les ressources de votre environnement et l'état devient rouge si les ressources ne parviennent pas à se déployer, ne sont pas correctement configurées ou deviennent indisponibles. Ces contrôles confirment ce qui suit :

- Le groupe Auto Scaling de l'environnement est disponible et possède au minimum une instance.
- Le groupe de sécurité de l'environnement est disponible et est configuré pour autoriser le trafic entrant sur le port 80.
- L'environnement CNAME existe et pointe vers l'équilibreur de charge approprié.
- Dans un environnement de travail, la file d'attente Amazon Simple Queue Service (Amazon SQS) est interrogée au moins une fois toutes les trois minutes.

CloudWatch Métriques Amazon

En ce qui concerne les rapports de santé de base, le service Elastic Beanstalk ne publie aucun indicateur sur Amazon. CloudWatch Les CloudWatch métriques utilisées pour produire des graphiques sur la page de surveillance de la console d'environnement sont publiées par les ressources de votre environnement.

Par exemple, EC2 publie les métriques suivantes pour les instances du groupe Auto Scaling de votre environnement :

`CPUUtilization`

Pourcentage d'unités de calcul actuellement en cours d'utilisation.

`DiskReadBytes`, `DiskReadOps`, `DiskWriteBytes`, `DiskWriteOps`

Nombre d'octets lus et écrits et nombre d'opérations de lecture et d'écriture.

`NetworkIn`, `NetworkOut`

Nombre d'octets envoyés et reçus.

Elastic Load Balancing publie les métriques suivantes pour l'équilibreur de charge de votre environnement :

`BackendConnectionErrors`

Nombre d'échecs de connexion entre l'équilibreur de charge et les instances d'environnement.

`HTTPCode_Backend_2XX`, `HTTPCode_Backend_4XX`

Nombre de codes de réponses aboutis (2XX) et d'erreur client (4XX) générés par des instances dans votre environnement.

`Latency`

Nombre de secondes entre le moment où l'équilibreur de charge relaie une demande à une instance et celui de la réception de la réponse.

`RequestCount`

Nombre de demandes terminées.

Ces listes ne sont pas complètes. Pour obtenir la liste complète des statistiques pouvant être signalées pour ces ressources, consultez les rubriques suivantes du manuel Amazon CloudWatch Developer Guide :

Métriques

Namespace	Rubrique
AWS::ElasticLoadBalancing::LoadBalancer	Métriques et ressources Elastic Load Balancing
AWS::AutoScaling::AutoScalingGroupe	Métriques et ressources Amazon Elastic Compute Cloud
AWS::SQS::Queue	Métriques et ressources Amazon SQS
AWS : :RDS : : DBInstance	Dimensions et métriques Amazon RDS

Métrique d'intégrité d'environnement de travail

Pour les environnements de travail uniquement, le démon SQS publie une métrique personnalisée pour l'état de l'environnement CloudWatch, où la valeur 1 est verte. Vous pouvez consulter les données des indicateurs CloudWatch de santé de votre compte à l'aide de l'espace de `ElasticBeanstalk/SQSD` noms. La dimension de métrique est `EnvironmentName`, et le nom de métrique est `Health`. Toutes les instances publient leurs métriques sur le même espace de noms.

Pour permettre au démon de publier des métriques, le profil d'instance de l'environnement doit avoir l'autorisation d'appeler `cloudwatch:PutMetricData`. Cette autorisation est incluse dans le profil d'instance par défaut. Pour plus d'informations, consultez Gestion des profils d'instance Elastic Beanstalk.

Rapports et surveillance de l'état de santé améliorés dans Elastic Beanstalk

Cette section explique les fonctionnalités de la fonctionnalité Elastic Beanstalk Enhanced Health.

Les rapports de santé améliorés sont une fonctionnalité que vous pouvez activer dans votre environnement AWS Elastic Beanstalk pour recueillir des informations supplémentaires sur les ressources de votre environnement. Elastic Beanstalk analyse les informations recueillies pour fournir

un meilleur aperçu de l'état global de l'environnement et permettre l'identification des problèmes pouvant entraîner une indisponibilité de votre application.

En plus des modifications dans le mode de fonctionnement des couleurs d'état, l'état amélioré ajoute un descripteur statut qui fournit un indicateur de la gravité des problèmes observés lorsqu'un environnement est jaune ou rouge. Lorsque davantage d'informations sont disponibles sur l'état actuel, vous pouvez choisir le bouton Causes pour afficher des informations d'état détaillées sur la page d'état.

Pour fournir des informations détaillées sur l'état des instances Amazon EC2 s'exécutant dans votre environnement, Elastic Beanstalk inclut un agent de vérification de l'état dans l'AMI (Amazon Machine Image) pour chaque version de plateforme qui prend en charge les rapports améliorés sur l'état. L'agent de vérification de l'état surveille les journaux de serveur web et les métriques système et les transmet au service Elastic Beanstalk. Elastic Beanstalk analyse ces métriques et ces données issues d'Elastic Load Balancing et d'Amazon EC2 Auto Scaling pour fournir un aperçu global de l'état d'un environnement.

Outre la collecte et la présentation des informations relatives aux ressources de votre environnement, Elastic Beanstalk surveille les ressources de votre environnement pour plusieurs conditions d'erreur et fournit des notifications pour vous aider à éviter les défaillances et à résoudre les problèmes de configuration. Les facteurs susceptibles d'influer sur l'état de votre environnement incluent les résultats de chaque demande de votre application, les métriques du système d'exploitation de vos instances et l'état du déploiement le plus récent.

Vous pouvez afficher l'état de santé en temps réel en utilisant la page de présentation de l'environnement de la console Elastic Beanstalk ou la commande eb health de l'interface de ligne de commande Elastic Beanstalk. Pour enregistrer et suivre l'état de santé de l'environnement et de l'instance au fil du temps, vous pouvez configurer votre environnement pour publier les informations recueillies par Elastic Beanstalk afin d'améliorer les rapports de santé sur Amazon sous forme de CloudWatch métriques personnalisées. CloudWatch les frais relatifs aux mesures personnalisées s'appliquent à toutes les mesures autres que celles `EnvironmentHealth` qui sont gratuites.

> **Remarques sur la plateforme Windows**
>
> Lorsque vous activez les rapports améliorés sur l'état de santé pour un environnement Windows Server, ne modifiez pas la configuration de la journalisation IIS. Pour que la surveillance améliorée de l'état de santé fonctionne correctement, la journalisation IIS doit être configurée avec le format W3C et les destinations d'événements de journal ETW event only ou Both log file and ETW event.

> Par ailleurs, ne désactivez pas ou n'arrêtez pas le service Windows de l'agent de vérification de l'état Elastic Beanstalk sur les instances de votre environnement. Pour collecter et signaler des informations d'état de santé améliorées sur une instance, ce service doit être activé et en cours d'exécution.

La première fois que vous créez un environnement, Elastic Beanstalk vous invite à créer les rôles requis et permet d'améliorer les rapports de santé par défaut. Poursuivez votre lecture pour en savoir plus sur le fonctionnement des rapports améliorés sur l'état, ou consultez Activation des rapports améliorés sur l'état Elastic Beanstalk pour commencer à utiliser immédiatement ces rapports.

Rubriques

- Agent de vérification de l'état Elastic Beanstalk
- Facteurs de détermination de l'intégrité de l'environnement et de l'instance
- Personnalisation d'une règle de vérification de l'état
- Rôles d'intégrité améliorée
- Autorisation de santé améliorée
- Événements d'intégrité améliorée
- Comportement de la création de rapports d'intégrité améliorée au cours des mises à jour, des déploiements et de la mise à l'échelle
- Activation des rapports améliorés sur l'état Elastic Beanstalk
- Surveillance améliorée de l'état avec la console de gestion de l'environnement
- Couleurs et états utilisés dans les rapports d'intégrité
- Métriques des instances
- Configuration de règles d'intégrité améliorée pour un environnement
- Publication de métriques CloudWatch personnalisées Amazon pour un environnement
- Utilisation des rapports améliorés sur l'état à l'aide de l'API Elastic Beanstalk
- Format de journal d'intégrité améliorée
- Notifications et dépannage
- Analyse de l'environnement basée sur l'IA

Agent de vérification de l'état Elastic Beanstalk

L'agent de vérification de l'état Elastic Beanstalk est un processus démon (ou un service, dans les environnements Windows) qui s'exécute sur chaque instance Amazon EC2 de votre environnement, en surveillant les métriques d'état au niveau de l'application et du système d'exploitation et en signalant les problèmes à Elastic Beanstalk. L'agent d'état est inclus dans toutes les versions de plateforme Linux à partir de la version 2.0 de chaque plateforme.

L'agent de santé fournit des statistiques similaires à celles publiées CloudWatch par Amazon EC2 Auto Scaling et Elastic Load Balancing dans le cadre des rapports de santé de base, notamment la charge du processeur, les codes HTTP et la latence. Toutefois, l'agent de vérification de l'état rapporte les métriques directement à Elastic Beanstalk, avec une granularité et une fréquence supérieures aux rapports de base sur l'état.

Pour l'intégrité de base, ces métriques sont publiées toutes les cinq minutes et peuvent être contrôlées avec des graphiques dans la console de gestion d'environnement. Avec les rapports améliorés sur l'état, l'agent de vérification de l'état Elastic Beanstalk rapporte des métriques à Elastic Beanstalk toutes les 10 secondes. Elastic Beanstalk utilise les métriques fournies par l'agent de vérification de l'état pour déterminer l'état de santé de chaque instance dans l'environnement et, combinées à d'autres facteurs, pour déterminer l'état global de l'environnement.

L'état général de l'environnement peut être consulté en temps réel sur la page de présentation de l'environnement de la console Elastic Beanstalk et est publié CloudWatch par Elastic Beanstalk toutes les 60 secondes. Vous pouvez consulter en temps réel les métriques détaillées indiquées par l'agent d'état en utilisant la commande eb health dans l'interface de ligne de commande EB.

Moyennant des frais supplémentaires, vous pouvez choisir de publier des métriques individuelles au niveau de l'instance et de l'environnement CloudWatch toutes les 60 secondes. Les métriques publiées sur CloudWatch peuvent ensuite être utilisées pour créer des graphiques de surveillance dans la console de gestion d'environnement.

La création de rapports d'intégrité améliorée n'implique un coût que si vous choisissez de publier des métriques d'intégrité améliorée sur CloudWatch. Lorsque vous utilisez l'intégrité améliorée, vous obtenez encore les métriques d'intégrité de base publiées gratuitement, même si vous ne choisissez pas de publier des métriques d'intégrité améliorée.

Pour obtenir des détails sur les métriques rapportées par l'agent de vérification de l'état, veuillez consulter Métriques des instances. Pour plus de détails sur la publication de statistiques de santé

améliorées sur CloudWatch, voirPublication de métriques CloudWatch personnalisées Amazon pour un environnement.

Facteurs de détermination de l'intégrité de l'environnement et de l'instance

Outre les vérifications système des rapports de base sur l'état, notamment les Vérifications de l'état Elastic Load Balancing et la surveillance des ressources, les rapports améliorés sur l'état Elastic Beanstalk collectent des données supplémentaires sur l'état des instances de votre environnement. Sont incluses les métriques du système d'exploitation, les journaux de serveur et l'état des opérations d'environnement en cours, telles que les déploiements et les mises à jour. Le service de rapports sur l'état Elastic Beanstalk associe des informations issues de toutes les sources disponibles et les analyse pour évaluer l'état global de l'environnement.

Opérations et commandes

Lorsque vous effectuez une opération dans votre environnement, telle que le déploiement d'une nouvelle version d'une application, Elastic Beanstalk apporte plusieurs modifications qui affectent l'état de santé de l'environnement.

Par exemple, lorsque vous déployez une nouvelle version d'une application dans un environnement qui exécute plusieurs instances, des messages similaires au suivant s'affichent lorsque vous surveillez l'état de santé de l'environnement avec l'interface de ligne de commande EB.

```
  id              status     cause
    Overall       Info       Command is executing on 3 out of 5 instances
  i-bb65c145      Pending    91 % of CPU is in use. 24 % in I/O wait
                             Performing application deployment (running for 31 seconds)
  i-ba65c144      Pending    Performing initialization (running for 12 seconds)
  i-f6a2d525      Ok         Application deployment completed 23 seconds ago and took 26
seconds
  i-e8a2d53b      Pending    94 % of CPU is in use. 52 % in I/O wait
                             Performing application deployment (running for 33 seconds)
  i-e81cca40      Ok
```

Dans cet exemple, l'état général de l'environnement est Ok et la cause de cet état est que la commande s'exécute sur 3 des 5 instances. Trois des instances dans l'environnement ont le statut Pending, indiquant qu'une opération est en cours.

Lorsqu'une opération est terminée, Elastic Beanstalk rapporte des informations complémentaires sur l'opération. A titre d'exemple, Elastic Beanstalk affiche les informations suivantes sur une instance qui a déjà été mise à jour avec la nouvelle version de l'application :

```
i-f6a2d525     Ok        Application deployment completed 23 seconds ago and took 26
 seconds
```

Les informations sur l'intégrité de l'instance incluent également des détails sur le déploiement le plus récent sur chaque instance dans votre environnement. Chaque instance indique un état et un ID de déploiement. L'ID de déploiement est un nombre entier qui augmente d'un niveau chaque fois que vous déployez une nouvelle version de votre application ou que vous modifiez les paramètres des options de configuration des instances, telles que les variables d'environnement. Vous pouvez utiliser les informations de déploiement pour identifier les instances qui exécutent la mauvaise version de votre application après un échec de déploiement propagé.

Dans la colonne de cause, Elastic Beanstalk inclut des messages d'information sur la réussite des opérations et d'autres états sains pour différentes vérifications de l'état. Ces messages ne sont toutefois pas conservés indéfiniment. Les causes des statuts d'environnement défectueux sont conservées jusqu'à ce que l'environnement renvoie un état sain.

Expiration de commande

Elastic Beanstalk applique un délai d'expiration de commande à partir du moment où une opération commence à autoriser une instance à effectuer la transition vers un état sain. Cette expiration de la commande est définie dans la configuration de déploiement et de mise à jour de votre environnement (dans l'espace de noms aws:elasticbeanstalk:command) et est paramétrée par défaut sur 10 minutes.

Les mises à jour propagées sont l'occasion pour Elastic Beanstalk d'appliquer un délai d'expiration distinct à chaque lot dans l'opération. Cette expiration est définie dans le cadre de la configuration des mises à jour propagées de l'environnement (dans l'espace de noms aws:autoscaling:updatepolicy:rollingupdate). Si toutes les instances dans le lot sont saines dans le délai de mise à jour continue, l'opération se poursuit et passe au lot suivant. Dans le cas contraire, l'opération échoue.

> **Note**
>
> Si votre application ne réussit pas les vérifications de l'état avec le statut OK, mais qu'elle est stable à un autre niveau, vous pouvez définir l'option `HealthCheckSuccessThreshold`

dans l'espace de noms `aws:elasticbeanstalk:command namespace` afin de modifier le niveau auquel Elastic Beanstalk considère une instance comme étant saine.

Pour qu'un environnement de serveur web soit considéré comme sain, chaque instance dans l'environnement ou le lot chaque instance doit réussir 12 vérifications de l'état consécutives en deux minutes. Dans un environnement de travail, chaque instance doit réussir 18 vérifications de l'état. Avant l'expiration de la commande, Elastic Beanstalk n'abaisse pas l'état de santé d'un environnement lorsque les vérifications de l'état échouent. Si les instances de l'environnement deviennent saines avant l'expiration de la commande, l'opération réussit.

Requêtes HTTP

Lorsqu'aucune opération n'est en cours sur un environnement, la source principale d'informations sur l'intégrité de l'instance et de l'environnement repose sur les journaux de serveur web pour chaque instance. Pour déterminer l'état d'une instance et l'état global de l'environnement, Elastic Beanstalk prend en compte le nombre de demandes, le résultat de chaque demande et la vitesse à laquelle chaque demande a été résolue.

Sur les plateformes Linux, Elastic Beanstalk lit et analyse les journaux des serveurs web pour obtenir des informations sur les demandes HTTP. Sur la plateforme Windows Server, Elastic Beanstalk reçoit directement ces informations du serveur web IIS.

Votre environnement peut ne pas avoir de serveur web actif. Par exemple, la plateforme Docker multi-conteneurs n'inclut pas de serveur web. Les autres plateformes comprennent un serveur web, et votre application peut le désactiver. Dans ces cas-là, votre environnement exige une configuration supplémentaire pour fournir à l'agent de vérification de l'état Elastic Beanstalk les journaux au format dont il a besoin pour transmettre les informations d'état au service Elastic Beanstalk. Consultez Format de journal d'intégrité améliorée pour plus de détails.

Métriques du système d'exploitation

Elastic Beanstalk surveille les métriques du système d'exploitation rapportées par l'agent de vérification de l'état pour identifier les instances qui sont constamment à court de ressources système.

Pour obtenir des détails sur les métriques rapportées par l'agent de vérification de l'état, veuillez consulter Métriques des instances.

Personnalisation d'une règle de vérification de l'état

Le rapport sur l'intégrité améliorée d'Elastic Beanstalk s'appuie sur un ensemble de règles qui déterminent l'intégrité de votre environnement. Certaines de ces règles peuvent ne pas être adaptées à votre application. Un cas courant est une application qui renvoie de fréquence erreurs HTTP 4xx en raison de sa conception. Elastic Beanstalk utilise l'une de ses règles par défaut pour conclure que quelque chose ne fonctionne pas correctement, puis modifie l'état de santé de votre environnement de OK à Avertissement, Dégradé ou Grave, en fonction du taux d'erreur. Pour gérer ce cas correctement, Elastic Beanstalk vous permet de configurer cette règle et d'ignorer les erreurs HTTP 4xx de l'application. Pour plus d'informations, consultez Configuration de règles d'intégrité améliorée pour un environnement.

Rôles d'intégrité améliorée

Les rapports améliorés sur l'état exigent deux rôles : un rôle de service pour Elastic Beanstalk et un profil d'instance pour l'environnement. Le rôle de service permet à Elastic Beanstalk d'interagir AWS avec d'autres services en votre nom afin de recueillir des informations sur les ressources de votre environnement. Le profil d'instance permet aux instances de votre environnement d'écrire des journaux dans Amazon S3 et de communiquer des informations améliorées sur l'état au service Elastic Beanstalk.

Lorsque vous créez un environnement Elastic Beanstalk à l'aide de la console Elastic Beanstalk ou de l'interface de ligne de commande EB, Elastic Beanstalk crée un rôle de service par défaut et attache les stratégies gérées requises à un profil d'instance par défaut pour votre environnement.

Si vous utilisez l'API, un SDK ou le AWS CLI pour créer des environnements, vous devez créer ces rôles à l'avance et les spécifier lors de la création de l'environnement afin d'améliorer l'intégrité. Pour obtenir des instructions sur la création de rôles appropriés pour vos environnements, veuillez consulter Rôles d'Elastic Beanstalk Service, profils d'instance et politiques utilisateur.

Nous vous recommandons d'utiliser des stratégies gérées pour votre profil d'instance et votre rôle de service. Les politiques gérées sont des politiques Gestion des identités et des accès AWS (IAM) gérées par Elastic Beanstalk. L'utilisation de stratégies gérées garantit que votre environnement dispose de toutes les autorisations nécessaires pour fonctionner correctement.

Pour le profil d'instance, vous pouvez utiliser la stratégie `AWSElasticBeanstalkWebTier` ou `AWSElasticBeanstalkWorkerTier` gérée, pour un environnement de niveau serveur web ou de niveau de travail, respectivement. Pour de plus amples informations sur ces deux stratégies de profil d'instance gérées, veuillez consulter the section called “Profils d'instance”.

Autorisation de santé améliorée

Les stratégies gérées du profil d'instance Elastic Beanstalk contiennent des autorisations pour l'action `elasticbeanstalk:PutInstanceStatistics`. Cette action ne fait pas partie de l'API Elastic Beanstalk. Elle fait partie d'une autre API que les instances d'environnement utilisent en interne pour communiquer des informations améliorées sur l'état au service Elastic Beanstalk. Vous n'appelez pas cette API directement.

Lorsque vous créez un environnement, l'autorisation de l'action `elasticbeanstalk:PutInstanceStatistics` est activée par défaut. Pour renforcer la sécurité de votre environnement et prévenir l'usurpation des données d'état de santé en votre nom, nous vous recommandons de garder l'autorisation de cette action activée. Si vous utilisez des stratégies gérées pour votre profil d'instance, cette fonction est disponible pour votre nouvel environnement sans aucune autre configuration. Si vous utilisez un profil d'instance personnalisé au lieu d'une stratégie gérée, votre environnement peut afficher un état de santé Aucune donnée. Cela se produit car les instances ne sont pas autorisées pour l'action qui communique des données d'intégrité améliorées au service.

Pour autoriser l'action, incluez l'instruction suivante dans votre profil d'instance.

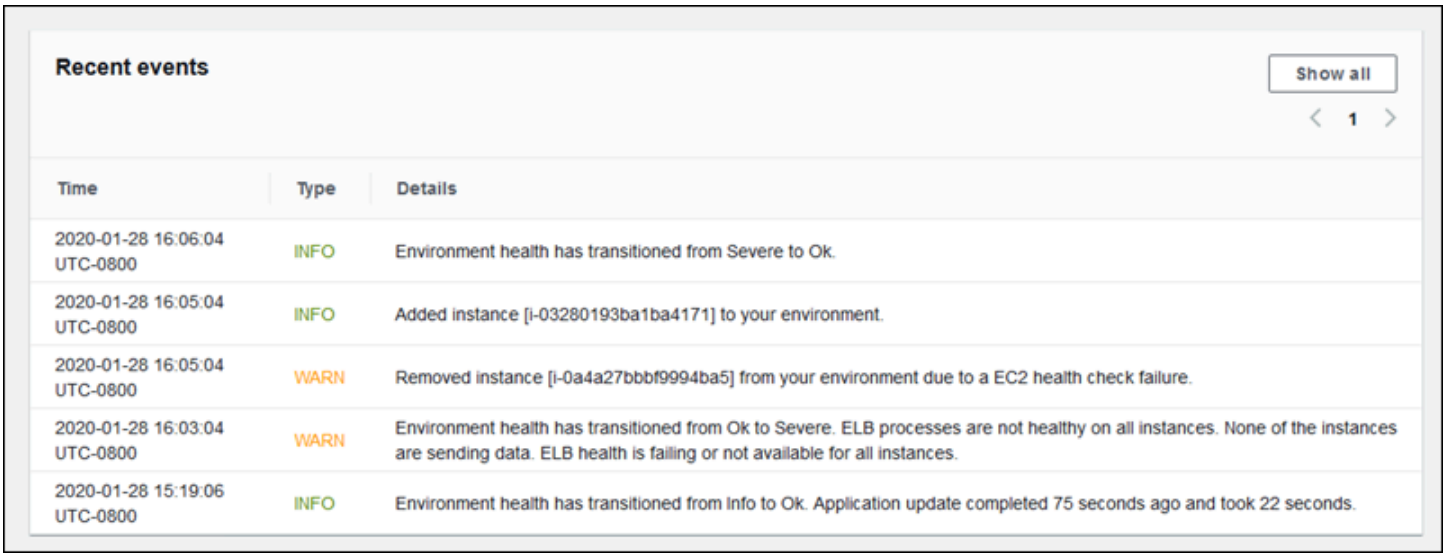
```
{
  "Sid": "ElasticBeanstalkHealthAccess",
  "Action": [
    "elasticbeanstalk:PutInstanceStatistics"
  ],
  "Effect": "Allow",
  "Resource": [
    "arn:aws:elasticbeanstalk:*:*:application/*",
    "arn:aws:elasticbeanstalk:*:*:environment/*"
  ]
}
```

Si vous ne souhaitez pas utiliser l'autorisation d'état de santé améliorée pour le moment, désactivez-la en définissant l'option `EnhancedHealthAuthEnabled` dans l'espace de noms the section called “aws:elasticbeanstalk:healthreporting:system” sur `false`. Si cette option est désactivée, les autorisations décrites précédemment ne sont pas requises. Vous pouvez les supprimer du profil d'instance pour accorder un accès sur la base du moindre privilège à vos applications et environnements.

> **Note**
>
> Auparavant, le paramètre par défaut pour `EnhancedHealthAuthEnabled` était `false`, ce entraînait la désactivation par défaut de l'autorisation de l'action `elasticbeanstalk:PutInstanceStatistics`. Pour activer cette action pour un environnement existant, définissez l'option `EnhancedHealthAuthEnabled` dans l'espace de noms the section called "aws:elasticbeanstalk:healthreporting:system" sur `true`. Vous pouvez configurer cette option à l'aide d'un paramètre d'option dans un fichier de configuration.

Événements d'intégrité améliorée

Le système d'intégrité améliorée génère des événements lorsqu'un environnement effectue la transition entre différents états. L'exemple suivant illustre la sortie d'événements provenant d'un environnement effectuant une transition entre les états Infos, OK et Grave.



Lors du passage à un état dégradé, l'événement de rapport amélioré sur l'état de santé inclut un message indiquant la cause de la transition.

Les changements de statut au niveau de l'instance ne conduisent pas tous Elastic Beanstalk à émettre un événement. Pour éviter les fausses alarmes, Elastic Beanstalk ne génère d'événement d'état que si un problème persiste dans plusieurs vérifications.

Des informations d'état au niveau de l'environnement en temps réel, y compris le statut, la couleur et la cause, sont disponibles sur la page de présentation de l'environnement de la console Elastic

Beanstalk et de l'interface de ligne de commande EB. En associant l'interface de ligne de commande EB à votre environnement et en exécutant la commande eb health, vous pouvez aussi afficher les statuts en temps réel de chacune des instances dans votre environnement.

Comportement de la création de rapports d'intégrité améliorée au cours des mises à jour, des déploiements et de la mise à l'échelle

Activer la création de rapports d'intégrité améliorée peut affecter le comportement de votre environnement au cours des déploiements et des mises à jour de configuration. Elastic Beanstalk ne termine pas un lot de mises à jour tant que toutes les instances n'ont pas réussi les vérifications de l'état. Par ailleurs, étant donné que les rapports améliorés sur l'état de santé appliquent un standard supérieur et surveille plusieurs facteurs, les instances qui réussissent la vérification de l'état ELB des rapports basiques sur l'état de santé ne réussissent pas forcément dans les rapports améliorés sur l'état de santé. Consultez les rubriques sur rolling configuration updates et rolling deployments pour plus d'informations sur la façon dont les vérifications de l'état affectent le processus de mise à jour.

Les rapports améliorés sur l'état peuvent également mettre en évidence la nécessité de définir une URL de vérification de l'état correcte pour Elastic Load Balancing. Lorsque votre environnement s'adapte pour répondre à la demande, de nouvelles instances commencent à accepter des demandes dès qu'elles réussissent suffisamment de vérifications de l'état ELB. Si aucune URL de vérification de l'état n'est configurée, le délai après qu'une nouvelle instance soit capable d'accepter une connexion TCP peut se réduire à seulement 20 secondes.

Si votre application n'a pas fini de démarrer au moment où l'équilibreur de charge la déclare suffisamment saine pour recevoir du trafic, vous verrez un flot de demandes ayant échoué, et votre environnement commencera à échouer à des vérifications de l'état. Une URL de vérification de l'état qui atteint un chemin servi par votre application peut éviter ce problème. Les vérifications de l'état ELB ne réussissent pas tant qu'une demande GET adressée à l'URL de vérification de l'état n'a pas renvoyé un code de statut 200.

Activation des rapports améliorés sur l'état Elastic Beanstalk

Cette rubrique explique comment les rapports de santé améliorés sont activés. Il fournit des procédures vous permettant d'activer la fonctionnalité d'intégrité améliorée pour votre environnement à l'aide de la console Elastic Beanstalk, de l'EB CLI et d'une configuration .ebextensions.

Les nouveaux environnements créés avec les dernières versions de la plateforme incluent l'agent AWS Elastic Beanstalk de santé, qui permet d'améliorer les rapports de santé. Si vous créez votre

environnement dans la console Elastic Beanstalk ou avec l'interface de ligne de commande EB, les rapports améliorés sur l'état sont activés par défaut. Vous pouvez également définir vos préférences relatives aux rapports améliorés sur l'état dans le code source de votre application, à l'aide des fichiers de configuration.

Les rapports améliorés sur l'état nécessitent un profil d'instance et un rôle de service incluant l'ensemble standard d'autorisations. Lorsque vous créez un environnement dans la console Elastic Beanstalk, Elastic Beanstalk crée automatiquement les rôles nécessaires. Pour obtenir des instructions sur la création de votre premier environnement, consultez Découvrez comment démarrer avec Elastic Beanstalk.

Rubriques

- Activation des rapports améliorés sur l'état à l'aide de la console Elastic Beanstalk
- Activation des rapports d'intégrité améliorée via l'interface de ligne de commande EB
- Activation des rapports d'intégrité améliorées via un fichier de configuration

Activation des rapports améliorés sur l'état à l'aide de la console Elastic Beanstalk

Pour activer les rapports améliorés sur l'état dans un environnement en cours d'exécution à l'aide de la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Monitoring (Surveillance), choisissez Edit (Modifier).
5. Dans la section Rapport sur l'état de santé, choisissez Amélioré dans le champ Présentation.

> **Note**
>
> Les options relatives aux rapports améliorés sur l'état de santé ne s'affichent pas si vous utilisez une plateforme ou une version non prise en charge.

6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

La console Elastic Beanstalk active par défaut les rapports améliorés sur l'état lorsque vous créez un environnement avec la version 2 (v2) de la plateforme. Vous pouvez désactiver les rapports améliorés sur l'état en modifiant l'option des rapports sur l'état lors de la création de l'environnement.

Pour désactiver les rapports améliorés sur l'état lors de la création d'un environnement à l'aide de la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Créez une application ou sélectionnez une application existante.
3. Créez un environnement. Sur la page Créer un nouvel environnement, avant de choisir Créer un environnement, choisissez Configurer plus d'options.
4. Dans la catégorie de configuration Monitoring (Surveillance), choisissez Edit (Modifier).
5. Dans la section Rapport sur l'état de santé, choisissez Basique dans le champ Présentation.
6. Choisissez Enregistrer.

Activation des rapports d'intégrité améliorée via l'interface de ligne de commande EB

Lorsque vous créez un environnement avec la commande eb create, l'interface de ligne de commande EB active les rapports améliorés sur l'état par défaut, et applique le rôle de service et le profil d'instance par défaut.

Vous pouvez spécifier un autre rôle de service par nom en utilisant l'option `--service-role`.

Si votre environnement est exécuté dans une version v2 de la plateforme avec des rapports basiques sur l'état de santé et que vous souhaitez passer aux rapports améliorés sur l'état de santé, procédez comme suit.

Pour activer les rapports améliorés sur l'état dans un environnement en cours d'exécution via l'interface de ligne de commande EB

1. Utilisez la commande eb config pour ouvrir le fichier de configuration dans l'éditeur de texte par défaut.

```
~/project$ eb config
```

2. Recherchez l'espace de noms `aws:elasticbeanstalk:environment` dans la section des paramètres. Assurez-vous que la valeur de `ServiceRole` n'est pas nulle et qu'elle correspond au nom de votre rôle de service.

```
  aws:elasticbeanstalk:environment:
    EnvironmentType: LoadBalanced
    ServiceRole: aws-elasticbeanstalk-service-role
```

3. Sous l'espace de noms `aws:elasticbeanstalk:healthreporting:system:`, remplacez la valeur `SystemType` par **enhanced**.

```
  aws:elasticbeanstalk:healthreporting:system:
    SystemType: enhanced
```

4. Enregistrez le fichier de configuration et fermez l'éditeur de texte.

5. L'interface de ligne de commande EB lance une mise à jour de l'environnement pour appliquer les modifications apportées à la configuration. Attendez la fin de l'opération ou appuyez sur Ctrl +C pour quitter l'interface en toute sécurité.

```
~/project$ eb config
Printing Status:
INFO: Environment update is starting.
INFO: Health reporting type changed to ENHANCED.
INFO: Updating environment no-role-test's configuration settings.
```

Activation des rapports d'intégrité améliorées via un fichier de configuration

Vous pouvez activer les rapports améliorés sur l'état en incluant un fichier de configuration dans votre bundle source. L'exemple suivant présente un fichier de configuration qui active les rapports améliorés sur l'état et affecte le rôle de service et le profil d'instance par défaut à l'environnement :

Example.ebextensions/enhanced-health.config

```
option_settings:
  aws:elasticbeanstalk:healthreporting:system:
    SystemType: enhanced
  aws:autoscaling:launchconfiguration:
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role
  aws:elasticbeanstalk:environment:
```

```
    ServiceRole: aws-elasticbeanstalk-service-role
```

Si vous avez créé votre propre rôle de service ou profil d'instance, remplacez le texte en surbrillance par les noms de ces rôles.

Surveillance améliorée de l'état avec la console de gestion de l'environnement

Lorsque vous activez les rapports de santé améliorés dans AWS Elastic Beanstalk, vous pouvez surveiller l'état de l'environnement dans la console de gestion de l'environnement.

Rubriques

- Aperçu de l'environnement
- Page « Health (Santé) » de l'environnement
- Page « Monitoring (Surveillance) »

Aperçu de l'environnement

La présentation de l'environnement affiche les statuts d'état de l'environnement et répertorie les événements qui apportent des informations sur l'évolution récente des statuts d'état.

Pour afficher la présentation de l'environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Pour plus d'informations sur l'état de santé de l'environnement actuel, ouvrez la page Santé en choisissant Causes. Sinon, dans le panneau de navigation, sélectionnez Health (Santé).

Page « Health (Santé) » de l'environnement

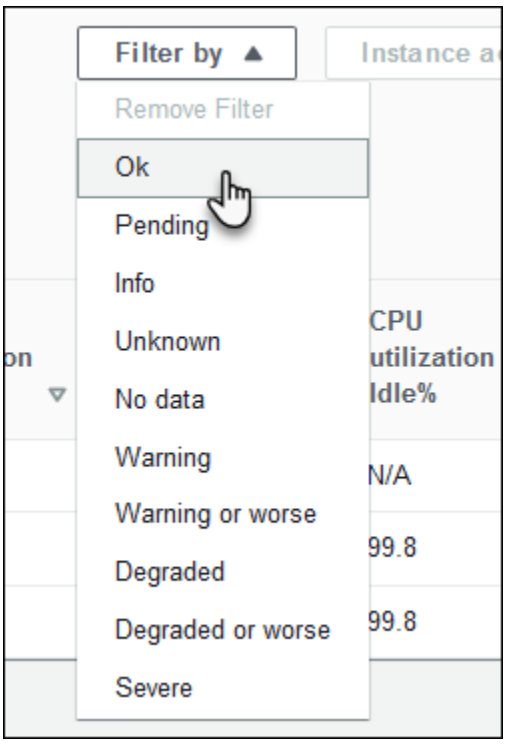
La page Health affiche l'état de santé, les métriques et les causes de l'environnement et de chaque EC2 instance Amazon de l'environnement.

> **Note**
>
> Elastic Beanstalk affiche la page Health (État) uniquement si vous avez activé la surveillance améliorée de l'état pour l'environnement.

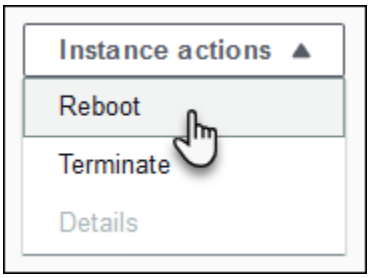
L'image suivante montre la page Health (Santé) pour un environnement Linux.

L'image suivante montre la page Health (Santé) pour un environnement Windows. Notez que les métriques d'UC sont différentes de celles d'un environnement Linux.

En haut de la page, vous pouvez voir le nombre total d'instances d'environnement, ainsi que le nombre d'instances par état. Pour afficher uniquement les instances associées à un état particulier, choisissez Filter By (Filtrer par), puis choisissez un état.



Pour redémarrer ou suspendre une instance défectueuse, choisissez Actions d'instance, puis Reboot (Redémarrer) ou Terminate (Résilier).

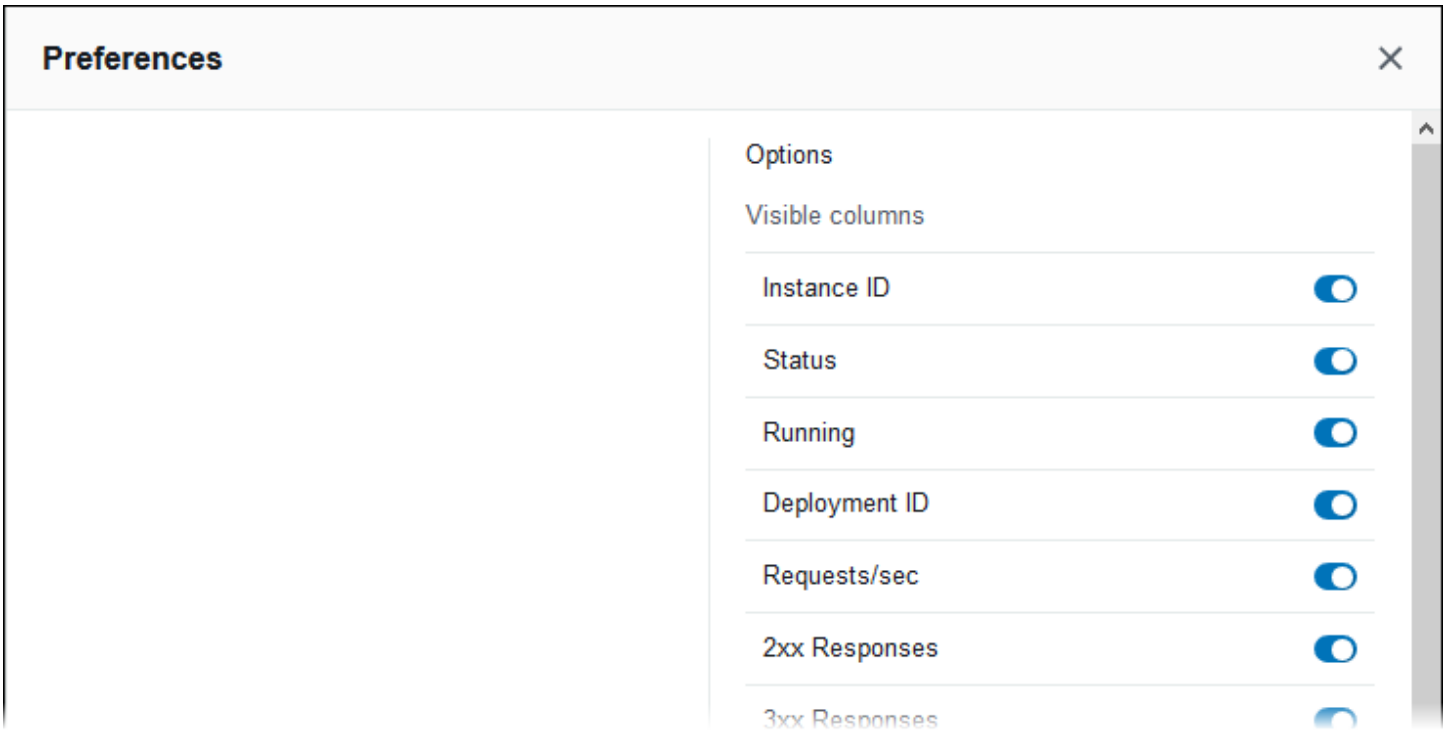


Elastic Beanstalk met à jour la page Health (État) toutes les 10 secondes. Il fournit des informations sur la santé de l'environnement et de l'instance.

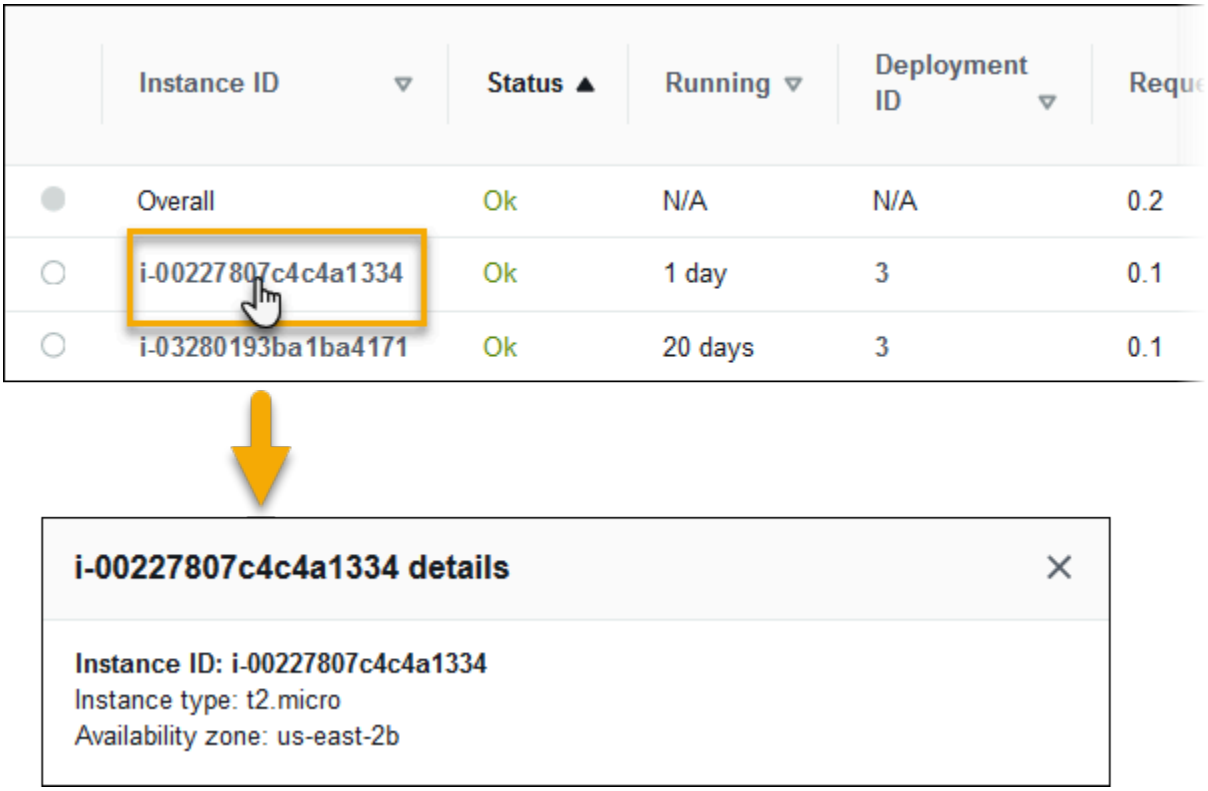
Pour chaque EC2 instance Amazon de l'environnement, la page affiche l'ID et le statut de l'instance, le temps écoulé depuis le lancement de l'instance, l'identifiant du dernier déploiement exécuté sur l'instance, les réponses et la latence des demandes traitées par l'instance, ainsi que les informations relatives à la charge et à l'utilisation du processeur. La ligne Overall (Globale) affiche des informations sur la réponse moyenne et la latence pour l'ensemble de l'environnement.

La page affiche de nombreux détails dans un très grand tableau. Pour masquer certaines colonnes, choisissez

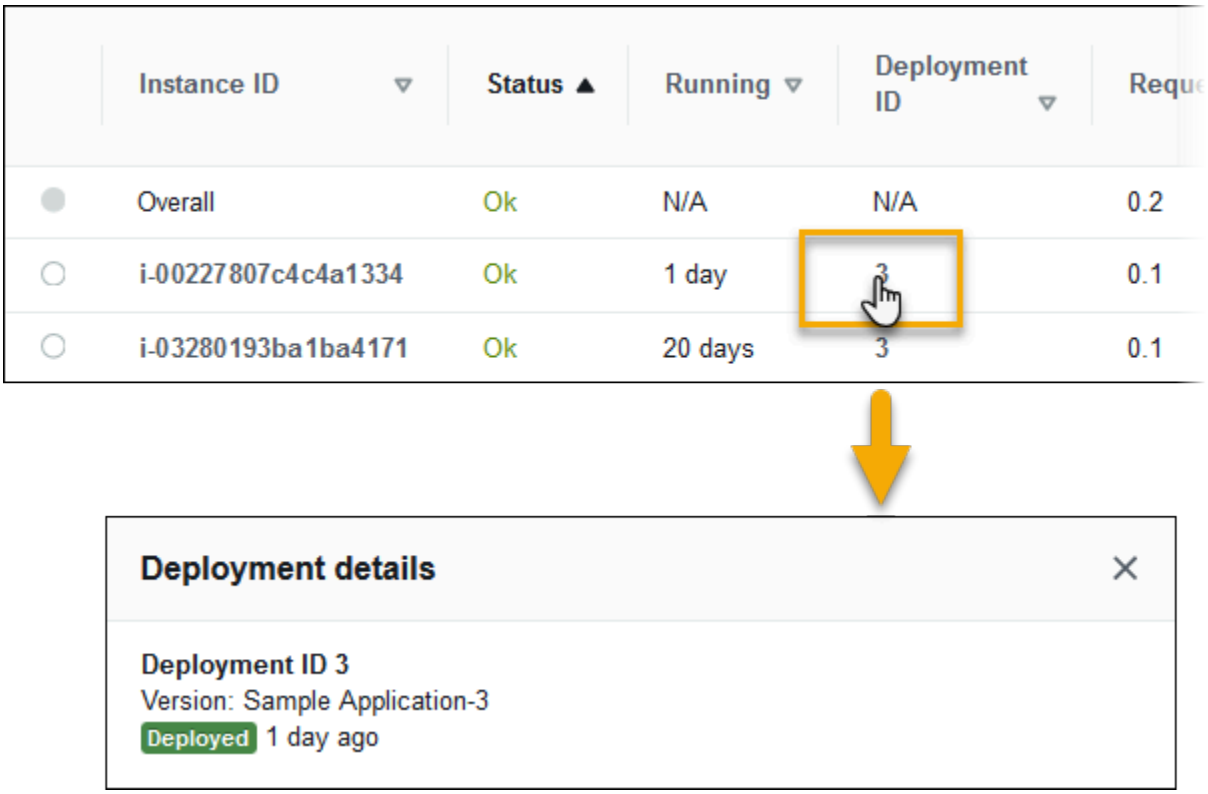
(Preferences (Préférences)). Sélectionnez ou effacez les noms de colonne, puis choisissez Confirme (Confirmer).



Choisissez l'Instance ID (ID d'instance) de n'importe quelle instance pour afficher plus d'informations sur cette dernière, y compris sa zone de disponibilité et le type d'instance.



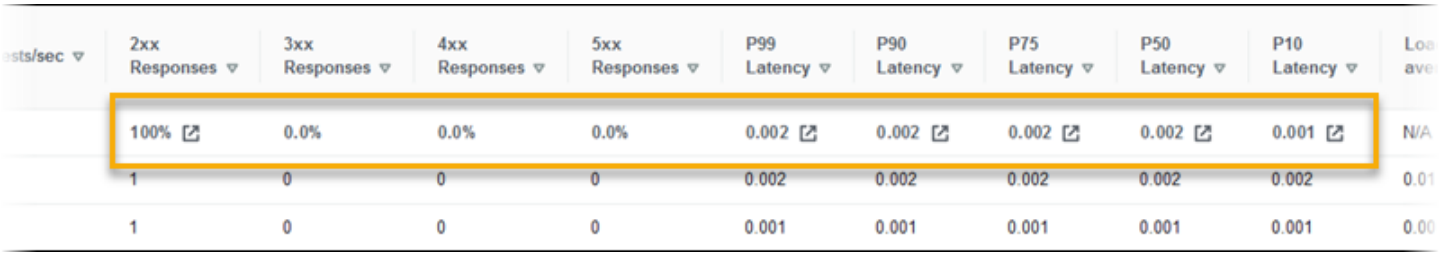
Choisissez l'Deployment ID (ID de déploiement) d'une instance pour afficher des informations sur le dernier déploiement sur l'instance.



Les informations sur le déploiement incluent les éléments suivants :

- ID de déploiement — Identifiant unique du déploiement. IDs Le déploiement commence à 1 et augmente d'une unité chaque fois que vous déployez une nouvelle version de l'application ou que vous modifiez les paramètres de configuration qui affectent le logiciel ou le système d'exploitation exécuté sur les instances de votre environnement.
- Version — Libellé de version du code source de l'application utilisé dans le déploiement.
- Statut — Statut du déploiement (`In Progress`, `Deployed` ou `Failed`).
- Durée écoulée — Durée écoulée depuis le début du déploiement (pour les déploiements en cours) ou durée écoulée depuis la fin du déploiement (pour les déploiements terminés).

Si vous activez l'intégration de X-Ray dans votre environnement et que vous instrumentez votre application avec le AWS X-Ray SDK, la page Health ajoute des liens vers la AWS X-Ray console dans la ligne de présentation.



sts/sec	2xx Responses	3xx Responses	4xx Responses	5xx Responses	P99 Latency	P90 Latency	P75 Latency	P50 Latency	P10 Latency	Loa ave
	100%	0.0%	0.0%	0.0%	0.002	0.002	0.002	0.002	0.001	N/A
	1	0	0	0	0.002	0.002	0.002	0.002	0.002	0.01
	1	0	0	0	0.001	0.001	0.001	0.001	0.001	0.00

Choisissez un lien pour afficher les traces associées aux statistiques mises en évidence dans la AWS X-Ray console.

Page « Monitoring (Surveillance) »

La page de surveillance affiche des statistiques récapitulatives et des graphiques pour les CloudWatch métriques Amazon personnalisées générées par le système de rapports de santé amélioré. Pour savoir comment ajouter des graphiques et des statistiques à cette page, veuillez consulter Surveillance de l'état de l'environnement dans la console AWS de gestion.

Couleurs et états utilisés dans les rapports d'intégrité

Les rapports améliorés sur l'état de santé indiquent l'état de santé des instances et de l'environnement global à l'aide de quatre couleurs, comme dans les rapports basiques sur l'état de santé. Les rapports améliorés sur l'état de santé incluent également sept statuts d'état de santé. Ce sont des termes descriptifs, composés d'un seul mot, qui vous permettent de mieux comprendre l'état de santé de votre environnement.

État de l'instance et état de l'environnement

Chaque fois qu'Elastic Beanstalk vérifie l'état de votre environnement, les rapports améliorés sur l'état vérifient l'état de chaque instance de votre environnement, en analysant l'ensemble des données disponibles. En cas d'échec de l'une des vérifications de niveau inférieur, Elastic Beanstalk rétrograde l'état de l'instance.

Elastic Beanstalk affiche les informations sur l'état pour l'environnement global (couleur, statut et cause) dans la console de gestion de l'environnement. Ces informations sont également disponibles dans l'interface de ligne de commande EB. Le statut d'état de santé et les messages d'explication sont mis à jour toutes les 10 secondes pour chaque instance, et sont disponibles via l'interface de ligne de commande EB lorsque vous affichez le statut d'état de santé avec eb health.

Elastic Beanstalk tire parti des changements d'état des instances pour évaluer l'état de l'environnement, mais ne change pas immédiatement l'état de santé de l'environnement. Lorsque la

vérification de l'état d'une instance échoue au moins trois fois dans un laps de temps d'une minute, Elastic Beanstalk peut rétrograder l'état de l'environnement. En fonction du nombre d'instances de l'environnement et du problème identifié, une instance défaillante peut conduire Elastic Beanstalk à afficher un message d'information ou à changer l'état de santé de l'environnement en le faisant passer de vert (OK) à jaune (Avertissement) ou rouge (Dégradé ou Grave).

OK (vert)

Ce statut s'affiche lorsque :

- L'instance réussit les vérifications de l'état et l'agent d'état ne signale aucun problème.
- La plupart des instances de l'environnement réussissent les vérifications de l'état et l'agent d'état ne signale aucun problème majeur.
- L'instance réussit les vérifications de l'état et traite les demandes normalement.

Exemple : votre environnement a été déployé récemment et accepte les demandes normalement. 5 % de demandes renvoient des erreurs de la série 400. Le déploiement s'est terminé normalement sur chaque instance.

Message (instance) : Application deployment completed 23 seconds ago and took 26 seconds.

Avertissement (jaune)

Ce statut s'affiche lorsque :

- L'agent d'état signale un nombre modéré d'échecs de la demande ou d'autres problèmes pour une instance ou un environnement.
- Une opération est en cours sur une instance et prend beaucoup de temps.

Exemple : une instance de l'environnement a le statut Grave.

Message (environnement) : Impaired services on 1 out of 5 instances.

Dégradé (rouge)

Ce statut s'affiche lorsque l'agent d'état signale un nombre élevé d'échecs de la demande ou d'autres problèmes pour une instance ou un environnement.

Exemple : l'environnement est en cours de mise à l'échelle ascendante vers 5 instances.

Message (environnement) : 4 active instances is below Auto Scaling group minimum size 5.

Grave (rouge)

Ce statut s'affiche lorsque l'agent d'état signale un nombre très élevé d'échecs de la demande ou d'autres problèmes pour une instance ou un environnement.

Exemple : Elastic Beanstalk ne parvient pas à contacter l'équilibreur de charge pour obtenir l'état de l'instance.

Message (environnement) : ELB health is failing or not available for all instances. None of the instances are sending data. Impossible d'assumer le rôle « arn:aws:iam : :123456789012:role/ ». aws-elasticbeanstalk-service-role Verify that the role exists and is configured correctly.

Message (Instances) : Instance ELB health has not been available for 37 minutes. Pas de données. Last seen 37 minutes ago.

Info (vert)

Ce statut s'affiche lorsque :

- Une opération est en cours sur une instance.
- Une opération est en cours sur plusieurs instances d'un environnement.

Exemple : une nouvelle version d'application est en cours de déploiement dans les instances en cours d'exécution.

Message (environnement) : Command is executing on 3 out of 5 instances.

Message (instance) : Performing application deployment (running for 3 seconds).

En attente (gris)

Ce statut s'affiche lorsqu'une opération est en cours sur une instance et que le délai d'attente de la commande n'est pas dépassé.

Exemple : vous avez créé l'environnement récemment et des instances sont en cours d'amorçage.

Message : Performing initialization (running for 12 seconds).

Inconnu (gris)

Ce statut s'affiche lorsqu'Elastic Beanstalk et l'agent de vérification de l'état signalent une quantité de données insuffisante sur une instance.

Exemple : aucune donnée n'est reçue.

Suspendu (gris)

Ce statut s'affiche lorsqu'Elastic Beanstalk cesse de surveiller l'état de l'environnement. L'environnement peut ne pas fonctionner correctement. Certaines conditions d'état graves, si elles persistent, conduisent Elastic Beanstalk à faire passer l'environnement au statut Suspendu.

Exemple : Elastic Beanstalk ne parvient pas à accéder au rôle de service de l'environnement.

Exemple : le groupe Auto Scaling créé par Elastic Beanstalk pour l'environnement a été supprimé.

Message : Environment health has transitioned from OK to Severe. Il n'y a aucune instance. La capacité souhaitée du groupe Auto Scaling est définie sur 1.

Métriques des instances

Des métriques d'instance fournissent des informations sur l'intégrité d'instances dans votre environnement. L'agent de vérification de l'état Elastic Beanstalk s'exécute sur chaque instance. Il rassemble et transmet à Elastic Beanstalk des métriques relatives aux instances. Elastic Beanstalk analyse ensuite ces métriques pour déterminer l'état des instances dans vos environnements.

L'agent de vérification de l'état Elastic Beanstalk sur instance recueille des métriques sur les instances à partir de serveurs web et du système d'exploitation. Pour obtenir des informations sur les serveurs web sur les plateformes Linux, Elastic Beanstalk lit et analyse les journaux des serveurs web. Sur la plateforme Windows Server, Elastic Beanstalk reçoit directement ces informations du serveur web IIS. Les serveurs web fournissent des informations sur les demandes HTTP entrantes : le nombre de requêtes qui sont entrées, combien ont généré des erreurs et le délai qui a été nécessaire à leur résolution. Le système d'exploitation fournit des informations d'instantané sur l'état des ressources des instances : la charge de l'UC et la distribution du temps consacré à chaque type de processus.

L'agent de vérification de l'état recueille des métriques de système d'exploitation et de serveur web et les transmet à Elastic Beanstalk toutes les 10 secondes. Elastic Beanstalk analyse les données et utilise les résultats pour mettre à jour l'état de santé de chaque instance et de l'environnement.

Rubriques

- Métriques de serveur web
- Métriques du système d'exploitation
- Capture des métriques du serveur web dans IIS sous Windows Server

Métriques de serveur web

Sur les plateformes Linux, l'agent de vérification de l'état Elastic Beanstalk lit les métriques de serveur web à partir des journaux générés par le conteneur web ou le serveur qui traite les demandes sur chaque instance de votre environnement. Les plateformes Elastic Beanstalk sont configurées pour générer deux journaux : un au format lisible par l'utilisateur et un au format lisible par la machine. L'agent de vérification de l'état transmet les journaux lisibles par la machine à Elastic Beanstalk toutes les 10 secondes.

Pour plus d'informations sur le format de journal utilisé par Elastic Beanstalk, consultez Format de journal d'intégrité améliorée.

Sur la plateforme Windows Server, Elastic Beanstalk ajoute un module au pipeline de demandes du serveur web IIS et capture les métriques relatives aux délais des demandes HTTP et aux codes de réponse. Le module envoie ces métriques à l'agent d'état de l'instance à l'aide d'un canal de communication inter-processus (IPC) hautes performances. Pour plus d'informations sur l'implémentation, consultez Capture des métriques du serveur web dans IIS sous Windows Server.

Métriques de serveur web signalées

`RequestCount`

Nombre de requêtes gérées par le serveur web par seconde au cours des 10 dernières secondes. Affiché comme un `r/sec` (demandes par seconde) moyen dans l'interface de ligne de commande EB et sur la Page « Health (Santé) » de l'environnement.

`Status2xx`, `Status3xx`, `Status4xx`, `Status5xx`

Nombre de requêtes ayant abouti sur chaque type de code de statut au cours des 10 dernières secondes. Par exemple, les demandes ayant abouti renvoient 200 OK, les redirections renvoient 301 et une erreur 404 est renvoyée si l'URL saisie ne correspond à aucune ressource de l'application.

L'interface de ligne de commande EB et la Page « Health (Santé) » de l'environnement affichent ces métrique sous la forme d'un nombre brut de demandes pour les instances et sous la forme d'un pourcentage des demandes globales pour les environnements.

`p99.9`, `p99`, `p95`, `p90`, `p85`, `p75`, `p50`, `p10`

Moyenne de latence pour le pourcentage de requêtes x le plus lent au cours des 10 dernières secondes, où x est la différence entre le nombre et 100. Par exemple, `p99 1.403` indique que les demandes faisant partie des 1 % les plus lents au cours des 10 dernières secondes avaient une latence moyenne de 1 351 secondes.

Métriques du système d'exploitation

L'agent de vérification de l'état Elastic Beanstalk rapporte les métriques de système d'exploitation suivantes. Elastic Beanstalk utilise ces métriques pour identifier les instances qui subissent une charge lourde constante. Les métriques diffèrent selon le système d'exploitation.

Métriques de système d'exploitation signalées (Linux)

`Running`

Le temps qui s'est écoulé depuis le lancement de l'instance.

`Load 1`, `Load 5`

Charge moyenne au cours des dernières périodes de 1 minute et de 5 minutes. Affiché comme une valeur décimale indiquant le nombre moyen de processus qui s'exécutent pendant cette durée. Si le nombre affiché est supérieur au nombre de v CPUs (threads) disponibles, le reste correspond au nombre moyen de processus en attente.

Par exemple, si votre type d'instance comporte quatre V CPUs et que la charge est de 4,5, il y a eu en moyenne 0,5 processus en attente pendant cette période, soit l'équivalent d'un processus en attente 50 % du temps.

`User %`, `Nice %`, `System %`, `Idle %`, `I/O Wait %`

Pourcentage de temps que l'UC a consacré à chaque état au cours des 10 dernières secondes.

Métriques de système d'exploitation signalées (Windows)

`Running`

Le temps qui s'est écoulé depuis le lancement de l'instance.

`% User Time`, `% Privileged Time`, `% Idle Time`

Pourcentage de temps que l'UC a consacré à chaque état au cours des 10 dernières secondes.

Capture des métriques du serveur web dans IIS sous Windows Server

Sur la plateforme Windows Server, Elastic Beanstalk ajoute un module au pipeline de demandes du serveur web IIS et capture les métriques relatives aux délais des demandes HTTP et aux codes de réponse. Le module envoie ces métriques à l'agent d'état de l'instance à l'aide d'un canal de communication inter-processus (IPC) hautes performances. L'agent de vérification de l'état regroupe ces métriques, les combine avec celles du système d'exploitation et les envoie au service Elastic Beanstalk.

Détails de l'implémentation

Pour capturer les métriques provenant d'IIS, Elastic Beanstalk implémente une interface `IHttpModule` gérée et s'abonne aux événements `BeginRequest` et `EndRequest`. Cela permet au module de signaler la latence des requêtes HTTP et les codes de réponse pour toutes les requêtes web gérées par IIS. Pour ajouter le module au pipeline de demandes d'IIS, Elastic Beanstalk enregistre le module dans la section `<modules>` du fichier de configuration IIS `%windir%\System32\inetsrv\config\applicationHost.config`.

Le module Elastic Beanstalk dans IIS envoie les métriques de demandes web capturées à l'agent de vérification de l'état sur instance, qui est un service Windows appelé `HealthD`. Pour envoyer ces données, le module utilise `NetNamedPipeBinding`, qui fournit une liaison sécurisée et fiable, optimisée pour les communications machine.

Configuration de règles d'intégrité améliorée pour un environnement

AWS Elastic Beanstalk les rapports de santé améliorés reposent sur un ensemble de règles visant à déterminer la santé de votre environnement. Certaines de ces règles peuvent ne pas être adaptées à votre application. Voici quelques exemples courants :

- Vous utilisez des outils de test côté client. Dans ce cas, des erreurs fréquentes de client HTTP (4xx) sont attendues.

- Vous utilisez AWS WAF avec Application Load Balancer de votre environnement pour bloquer le trafic entrant indésirable. Dans ce cas, Application Load Balancer renvoie l'erreur HTTP 403 pour chaque message entrant rejeté.

Par défaut, Elastic Beanstalk inclut toutes les erreurs HTTP 4xx de l'application lors de la détermination de l'état de l'environnement. L'état de santé de votre environnement passe de OK à Warning (Avertissement), Degraded (Dégradé) ou Severe (Grave), en fonction du taux d'erreur. Pour gérer correctement les cas mentionnés dans les exemples précédents, Elastic Beanstalk vous permet de configurer certaines règles d'état améliorées. Vous pouvez choisir d'ignorer les erreurs HTTP 4xx de l'application sur les instances de l'environnement ou d'ignorer les erreurs HTTP 4xx renvoyées par l'équilibreur de charge de l'environnement. Cette rubrique décrit comment effectuer ces modifications de configuration.

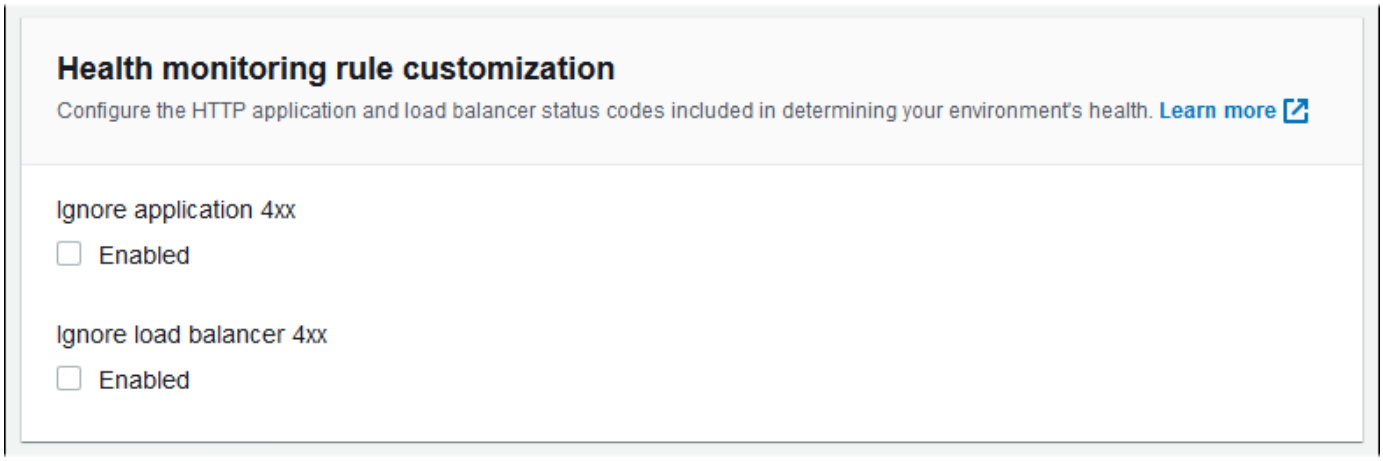
> **Note**
>
> Actuellement, il s'agit de la seule personnalisation de règle d'intégrité améliorée disponible. Vous ne pouvez pas configurer l'intégrité améliorée pour ignorer d'autres erreurs HTTP en plus de 4xx.

Configuration des règles d'état améliorées à l'aide de la console Elastic Beanstalk

Vous pouvez utiliser la console Elastic Beanstalk pour configurer les règles d'état améliorées dans votre environnement.

Pour configurer la vérification des codes d'état HTTP 4xx à l'aide de la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Monitoring (Surveillance), choisissez Edit (Modifier).
5. Sous Personnalisation de la règle de surveillance de l'intégrité, activez ou désactivez les options Ignorer souhaitées.



6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Configuration des règles d'intégrité améliorée à l'aide de l'interface de ligne de commande EB

Vous pouvez utiliser l'interface de ligne de commande EB pour configurer les règles d'état améliorées en enregistrant la configuration de votre environnement en local, en ajoutant une entrée qui configure les règles d'état améliorées, puis en chargeant la configuration dans Elastic Beanstalk. Vous pouvez appliquer la configuration enregistrée à un environnement pendant ou après la création.

Pour configurer la vérification du code d'état HTTP 4xx à l'aide de l'interface de ligne de commande EB et des configurations enregistrées

1. Initialisez votre dossier de projet avec eb init.
2. Créez un environnement en exécutant la commande eb create.
3. Enregistrez un modèle de configuration localement en exécutant la commande eb config save. L'exemple suivant utilise l'option `--cfg` pour spécifier le nom de la configuration.

```
$ eb config save --cfg 01-base-state
Configuration saved at: ~/project/.elasticbeanstalk/saved_configs/01-base-
state.cfg.yml
```

4. Ouvrez le fichier de configuration enregistrée dans un éditeur de texte.
5. Sous `OptionSettings` > `aws:elasticbeanstalk:healthreporting:system:`, ajoutez une clé `ConfigDocument` pour lister chaque règle d'intégrité améliorée à configurer. Le `ConfigDocument` suivant désactive la vérification des codes d'état HTTP 4xx de l'application, tout en conservant la vérification du code HTTP 4xx de l'équilibreur de charge activé.

```
OptionSettings:
  ...
  aws:elasticbeanstalk:healthreporting:system:
    ConfigDocument:
      Rules:
        Environment:
          Application:
            ApplicationRequests4xx:
              Enabled: false
          ELB:
            ELBRequests4xx:
              Enabled: true
      Version: 1
    SystemType: enhanced
...
```

> **Note**
>
> Vous pouvez combiner `Rules` et `CloudWatchMetrics` dans le même paramètre d'option `ConfigDocument`. Le paramètre `CloudWatchMetrics` est décrit dans Publication de métriques CloudWatch personnalisées Amazon pour un environnement. Si vous avez précédemment activé `CloudWatchMetrics`, le fichier de configuration que vous récupérez à l'aide de la commande eb config save possède déjà une clé `ConfigDocument` avec une section `CloudWatchMetrics`. Ne la supprimez pas : ajoutez une section `Rules` dans la même valeur d'option `ConfigDocument`.

6. Enregistrez le fichier de configuration et fermez l'éditeur de texte. Dans cet exemple, le fichier de configuration mis à jour est enregistré avec un nom qui est différent (`02-cloudwatch-enabled.cfg.yml`) de celui du fichier de configuration téléchargé. Cela crée une configuration enregistrée distincte lorsque le fichier est téléchargé. Vous pouvez utiliser le même nom que le fichier téléchargé pour remplacer la configuration existante sans en créer une.
7. Utilisez la commande eb config put pour charger le fichier de configuration mis à jour dans Elastic Beanstalk.

```
$ eb config put 02-cloudwatch-enabled
```

Lorsque vous utilisez les commandes eb config `get` et `put` avec des configurations enregistrés, n'incluez pas l'extension de nom de fichier.

8. Appliquez la configuration enregistrée à votre environnement en cours d'exécution.

```
$ eb config --cfg 02-cloudwatch-enabled
```

L'option `--cfg` spécifie un fichier de configuration nommé qui est appliqué à l'environnement. Vous pouvez enregistrer le fichier de configuration en local ou dans Elastic Beanstalk. Si un fichier de configuration avec le nom spécifié existe dans les deux emplacements, l'interface de ligne de commande EB utilise le fichier local.

Configuration des règles d'intégrité améliorée à l'aide d'un document de configuration

Le document de configuration pour les règles d'intégrité améliorée est un document JSON qui répertorie les règles à configurer.

L'exemple suivant montre un document de configuration qui désactive la vérification des codes d'état HTTP 4xx de l'application et active la vérification des codes d'état HTTP 4xx de l'équilibreur de charge.

```
{
  "Rules": {
    "Environment": {
      "Application": {
        "ApplicationRequests4xx": {
          "Enabled": false
        }
      },
      "ELB": {
        "ELBRequests4xx": {
          "Enabled": true
        }
      }
    }
  },
  "Version": 1
}
```

Pour le AWS CLI, vous transmettez le document en tant que valeur de `Value` clé dans un argument de paramètres d'option, qui est lui-même un objet JSON. Dans ce cas, vous devez utiliser des

guillemets d'échappement dans le document intégré. La commande suivante vérifie si les paramètres de configuration sont valides.

```
$ aws elasticbeanstalk validate-configuration-settings --application-name my-app --
environment-name my-env --option-settings '[
    {
        "Namespace": "aws:elasticbeanstalk:healthreporting:system",
        "OptionName": "ConfigDocument",
        "Value": "{\"Rules\": { \"Environment\": { \"Application\":
 { \"ApplicationRequests4xx\": { \"Enabled\": false } }, \"ELB\": { \"ELBRequests4xx\":
 {\"Enabled\": true } } } }, \"Version\": 1 }"
    }
]'
```

Pour un fichier de configuration `.ebextensions` au format YAML, vous pouvez fournir le document JSON en l'état.

```
  option_settings:
    - namespace: aws:elasticbeanstalk:healthreporting:system
      option_name: ConfigDocument
      value: {
  "Rules": {
    "Environment": {
      "Application": {
        "ApplicationRequests4xx": {
          "Enabled": false
        }
      },
      "ELB": {
        "ELBRequests4xx": {
          "Enabled": true
        }
      }
    }
  },
  "Version": 1
}
```

Publication de métriques CloudWatch personnalisées Amazon pour un environnement

Vous pouvez publier les données recueillies par le biais de rapports de santé AWS Elastic Beanstalk améliorés sur Amazon CloudWatch sous forme de statistiques personnalisées. La publication de métriques vous CloudWatch permet de surveiller l'évolution des performances des applications au fil du temps et d'identifier les problèmes potentiels en suivant l'évolution de l'utilisation des ressources et de la latence des demandes en fonction de la charge.

En publiant des métriques sur CloudWatch, vous les rendez également disponibles pour une utilisation avec des graphiques de surveillance et des alarmes. Une métrique gratuite est automatiquement activée lorsque vous utilisez des rapports de santé améliorés. EnvironmentHealth Mesures personnalisées autres que celles liées EnvironmentHealthaux CloudWatch frais standard.

Pour publier des métriques CloudWatch personnalisées pour un environnement, vous devez d'abord activer les rapports de santé améliorés sur l'environnement. Pour obtenir des instructions, consultez Activation des rapports améliorés sur l'état Elastic Beanstalk.

Rubriques

- Métriques de création de rapports d'intégrité améliorés
- Configuration des CloudWatch métriques à l'aide de la console Elastic Beanstalk
- Configuration de métriques CloudWatch personnalisées à l'aide de l'interface de ligne de commande EB
- Fourniture des documents de configuration des métriques personnalisées

Métriques de création de rapports d'intégrité améliorés

Lorsque vous activez les rapports de santé améliorés dans votre environnement, le système de rapports de santé améliorés publie automatiquement une métrique CloudWatch personnalisée, EnvironmentHealth. Pour publier des métriques supplémentaires CloudWatch, configurez votre environnement avec ces métriques à l'aide de la console Elastic Beanstalk, de l'EB CLI ou de .ebextensions.

Vous pouvez publier les indicateurs de santé améliorés suivants depuis votre environnement vers CloudWatch.

Métriques disponibles (toutes les plateformes)

`EnvironmentHealth`

Environnement uniquement. Il s'agit de la seule CloudWatch métrique publiée par le système de rapports de santé améliorés, sauf si vous configurez des métriques supplémentaires. L'état de l'environnement est représentée par un des sept statuts. Dans la CloudWatch console, ces statuts correspondent aux valeurs suivantes :

- 0 – OK
- 1 – Info
- 5 – Inconnu
- 10 – Pas de données
- 15 – Avertissement
- 20 – Dégradé
- 25 – Grave

`InstancesSevere`, `InstancesDegraded`, `InstancesWarning`, `InstancesInfo`, `InstancesOk`, `InstancesPending`, `InstancesUnknown`, `InstancesNoData`

Environnement uniquement. Ces métriques indiquent le nombre d'instances dans les environnement avec chaque état de santé. `InstancesNoData` indique le nombre d'instances pour lesquelles aucune donnée ne sera reçue.

`ApplicationRequestsTotal`, `ApplicationRequests5xx`, `ApplicationRequests4xx`, `ApplicationRequests3xx`, `ApplicationRequests2xx`

Instance et environnement. Indique le nombre total de requêtes terminées par l'instance ou l'environnement et le nombre de requêtes ayant abouti avec chaque catégorie de code d'état.

`ApplicationLatencyP10`, `ApplicationLatencyP50`, `ApplicationLatencyP75`, `ApplicationLatencyP85`, `ApplicationLatencyP90`, `ApplicationLatencyP95`, `ApplicationLatencyP99`, `ApplicationLatencyP99.9`

Instance et environnement. Indique la quantité moyenne de temps, en secondes, nécessaire pour terminer le pourcentage x le plus rapide de requêtes.

`InstanceHealth`

Instance uniquement. Indique l'état d'intégrité actuel de l'instance. L'état d'instance est représentée par un statut (sur sept statuts au total). Dans la CloudWatch console, ces statuts correspondent aux valeurs suivantes :

- 0 – OK
- 1 – Info
- 5 – Inconnu
- 10 – Pas de données
- 15 – Avertissement
- 20 – Dégradé
- 25 – Grave

Métriques disponibles (Linux)

`CPUIrq`, `CPUIdle`, `CPUUser`, `CPUSystem`, `CPUSoftirq`, `CPUIowait`, `CPUNice`

Instance uniquement. Indique le pourcentage de temps que l'UC a passé dans chaque état au cours de la dernière minute.

`LoadAverage1min`

Instance uniquement. La charge d'UC moyenne de l'instance au cours de la dernière minute.

`RootFilesystemUtil`

Instance uniquement. Indique le pourcentage d'espace disque en cours d'utilisation.

Métriques disponibles (Windows)

`CPUIdle`, `CPUUser`, `CPUPrivileged`

Instance uniquement. Indique le pourcentage de temps que l'UC a passé dans chaque état au cours de la dernière minute.

Configuration des CloudWatch métriques à l'aide de la console Elastic Beanstalk

Vous pouvez utiliser la console Elastic Beanstalk pour configurer votre environnement afin de publier des CloudWatch indicateurs de santé améliorés et de les rendre disponibles pour une utilisation avec des graphiques de surveillance et des alarmes.

Pour configurer CloudWatch des métriques personnalisées dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Monitoring (Surveillance), choisissez Edit (Modifier).
5. Sous Rapport sur l'état de santé, sélectionnez les métriques d'instance et d'environnement que vous voulez publier dans CloudWatch. Pour sélectionner plusieurs métriques, appuyez sur la touche Ctrl tout en choisissant.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

L'activation des métriques CloudWatch personnalisées les ajoute à la liste des métriques disponibles sur la page Monitoring.

Configuration de métriques CloudWatch personnalisées à l'aide de l'interface de ligne de commande EB

Vous pouvez utiliser l'interface de ligne de commande EB pour configurer des métriques personnalisées en enregistrant la configuration de votre environnement en local, en ajoutant une entrée qui définit les métriques à publier, puis en chargeant la configuration dans Elastic Beanstalk. Vous pouvez appliquer la configuration enregistrée à un environnement pendant ou après la création.

Pour configurer des métriques CloudWatch personnalisées à l'aide de l'interface de ligne de commande EB et des configurations enregistrées

1. Initialisez votre dossier de projet avec eb init.
2. Créez un environnement en exécutant la commande eb create.
3. Enregistrez un modèle de configuration localement en exécutant la commande eb config save. L'exemple suivant utilise l'option `--cfg` pour spécifier le nom de la configuration.

   ```
   $ eb config save --cfg 01-base-state
   Configuration saved at: ~/project/.elasticbeanstalk/saved_configs/01-base-
   state.cfg.yml
   ```

4. Ouvrez le fichier de configuration enregistrée dans un éditeur de texte.

5. Sous `OptionSettings` >`aws:elasticbeanstalk:healthreporting:system:`, ajoutez une `ConfigDocument` clé pour activer chacune des CloudWatch mesures souhaitées. Par exemple, le `ConfigDocument` suivant publie des métriques `ApplicationRequests5xx` et `ApplicationRequests4xx` au niveau de l'environnement et des métriques `ApplicationRequestsTotal` au niveau de l'instance.

```
OptionSettings:
  ...
  aws:elasticbeanstalk:healthreporting:system:
    ConfigDocument:
      CloudWatchMetrics:
        Environment:
          ApplicationRequests5xx: 60
          ApplicationRequests4xx: 60
        Instance:
          ApplicationRequestsTotal: 60
      Version: 1
    SystemType: enhanced
...
```

Dans l'exemple, 60 indique le nombre de secondes entre les mesures. C'est la seule valeur actuellement prise en charge.

> **Note**
>
> Vous pouvez combiner `CloudWatchMetrics` et `Rules` dans le même paramètre d'option `ConfigDocument`. Le paramètre `Rules` est décrit dans Configuration de règles d'intégrité améliorée pour un environnement.
> Si vous avez précédemment utilisé `Rules` pour configurer les règles d'intégrité améliorée, le fichier de configuration que vous récupérez à l'aide de la commande eb config save possède déjà une clé `ConfigDocument` avec une section `Rules`. Ne la supprimez pas : ajoutez une section `CloudWatchMetrics` dans la même valeur d'option `ConfigDocument`.

6. Enregistrez le fichier de configuration et fermez l'éditeur de texte. Dans cet exemple, le fichier de configuration mis à jour est enregistré avec un nom qui est différent (`02-cloudwatch-enabled.cfg.yml`) de celui du fichier de configuration téléchargé. Cela crée une configuration enregistrée distincte lorsque le fichier est téléchargé. Vous pouvez utiliser le même nom que le fichier téléchargé pour remplacer la configuration existante sans en créer une.

7. Utilisez la commande eb config put pour charger le fichier de configuration mis à jour dans Elastic Beanstalk.

   ```
   $ eb config put 02-cloudwatch-enabled
   ```

   Lorsque vous utilisez les commandes eb config `get` et `put` avec des configurations enregistrées, n'incluez pas l'extension de fichier.

8. Appliquez la configuration enregistrée à votre environnement en cours d'exécution.

   ```
   $ eb config --cfg 02-cloudwatch-enabled
   ```

   L'option `--cfg` spécifie un fichier de configuration nommé qui est appliqué à l'environnement. Vous pouvez enregistrer le fichier de configuration en local ou dans Elastic Beanstalk. Si un fichier de configuration avec le nom spécifié existe dans les deux emplacements, l'interface de ligne de commande EB utilise le fichier local.

Fourniture des documents de configuration des métriques personnalisées

Le document de configuration (config) pour les métriques CloudWatch personnalisées Amazon est un document JSON qui répertorie les métriques à publier au niveau de l'environnement et de l'instance. L'exemple suivant illustre un document de configuration qui active toutes les métriques personnalisées disponibles.

```
{
  "CloudWatchMetrics": {
    "Environment": {
      "ApplicationLatencyP99.9": 60,
      "InstancesSevere": 60,
      "ApplicationLatencyP90": 60,
      "ApplicationLatencyP99": 60,
      "ApplicationLatencyP95": 60,
      "InstancesUnknown": 60,
      "ApplicationLatencyP85": 60,
      "InstancesInfo": 60,
      "ApplicationRequests2xx": 60,
      "InstancesDegraded": 60,
      "InstancesWarning": 60,
      "ApplicationLatencyP50": 60,
```

```
      "ApplicationRequestsTotal": 60,
      "InstancesNoData": 60,
      "InstancesPending": 60,
      "ApplicationLatencyP10": 60,
      "ApplicationRequests5xx": 60,
      "ApplicationLatencyP75": 60,
      "InstancesOk": 60,
      "ApplicationRequests3xx": 60,
      "ApplicationRequests4xx": 60
    },
    "Instance": {
      "ApplicationLatencyP99.9": 60,
      "ApplicationLatencyP90": 60,
      "ApplicationLatencyP99": 60,
      "ApplicationLatencyP95": 60,
      "ApplicationLatencyP85": 60,
      "CPUUser": 60,
      "ApplicationRequests2xx": 60,
      "CPUIdle": 60,
      "ApplicationLatencyP50": 60,
      "ApplicationRequestsTotal": 60,
      "RootFilesystemUtil": 60,
      "LoadAverage1min": 60,
      "CPUIrq": 60,
      "CPUNice": 60,
      "CPUIowait": 60,
      "ApplicationLatencyP10": 60,
      "LoadAverage5min": 60,
      "ApplicationRequests5xx": 60,
      "ApplicationLatencyP75": 60,
      "CPUSystem": 60,
      "ApplicationRequests3xx": 60,
      "ApplicationRequests4xx": 60,
      "InstanceHealth": 60,
      "CPUSoftirq": 60
    }
  },
  "Version": 1
}
```

Pour le AWS CLI, vous transmettez le document en tant que valeur de `Value` clé dans un argument de paramètres d'option, qui est lui-même un objet JSON. Dans ce cas, vous devez utiliser des guillemets d'échappement dans le document intégré.

```
$ aws elasticbeanstalk validate-configuration-settings --application-name my-app --
environment-name my-env --option-settings '[
    {
        "Namespace": "aws:elasticbeanstalk:healthreporting:system",
        "OptionName": "ConfigDocument",
        "Value": "{\"CloudWatchMetrics\": {\"Environment\":
 {\"ApplicationLatencyP99.9\": 60,\"InstancesSevere\": 60,\"ApplicationLatencyP90\":
 60,\"ApplicationLatencyP99\": 60,\"ApplicationLatencyP95\": 60,\"InstancesUnknown
\": 60,\"ApplicationLatencyP85\": 60,\"InstancesInfo\": 60,\"ApplicationRequests2xx
\": 60,\"InstancesDegraded\": 60,\"InstancesWarning\": 60,\"ApplicationLatencyP50\":
 60,\"ApplicationRequestsTotal\": 60,\"InstancesNoData\": 60,\"InstancesPending
\": 60,\"ApplicationLatencyP10\": 60,\"ApplicationRequests5xx\": 60,
\"ApplicationLatencyP75\": 60,\"InstancesOk\": 60,\"ApplicationRequests3xx\": 60,
\"ApplicationRequests4xx\": 60},\"Instance\": {\"ApplicationLatencyP99.9\": 60,
\"ApplicationLatencyP90\": 60,\"ApplicationLatencyP99\": 60,\"ApplicationLatencyP95\":
 60,\"ApplicationLatencyP85\": 60,\"CPUUser\": 60,\"ApplicationRequests2xx\":
 60,\"CPUIdle\": 60,\"ApplicationLatencyP50\": 60,\"ApplicationRequestsTotal\":
 60,\"RootFilesystemUtil\": 60,\"LoadAverage1min\": 60,\"CPUIrq\": 60,\"CPUNice
\": 60,\"CPUIowait\": 60,\"ApplicationLatencyP10\": 60,\"LoadAverage5min\": 60,
\"ApplicationRequests5xx\": 60,\"ApplicationLatencyP75\": 60,\"CPUSystem\": 60,
\"ApplicationRequests3xx\": 60,\"ApplicationRequests4xx\": 60,\"InstanceHealth\": 60,
\"CPUSoftirq\": 60}},\"Version\": 1}"
    }
]'
```

Pour un fichier de configuration `.ebextensions` au format YAML, vous pouvez fournir le document JSON en l'état.

```
  option_settings:
    - namespace: aws:elasticbeanstalk:healthreporting:system
      option_name: ConfigDocument
      value: {
  "CloudWatchMetrics": {
    "Environment": {
      "ApplicationLatencyP99.9": 60,
      "InstancesSevere": 60,
      "ApplicationLatencyP90": 60,
      "ApplicationLatencyP99": 60,
      "ApplicationLatencyP95": 60,
      "InstancesUnknown": 60,
      "ApplicationLatencyP85": 60,
      "InstancesInfo": 60,
      "ApplicationRequests2xx": 60,
```

```
      "InstancesDegraded": 60,
      "InstancesWarning": 60,
      "ApplicationLatencyP50": 60,
      "ApplicationRequestsTotal": 60,
      "InstancesNoData": 60,
      "InstancesPending": 60,
      "ApplicationLatencyP10": 60,
      "ApplicationRequests5xx": 60,
      "ApplicationLatencyP75": 60,
      "InstancesOk": 60,
      "ApplicationRequests3xx": 60,
      "ApplicationRequests4xx": 60
    },
    "Instance": {
      "ApplicationLatencyP99.9": 60,
      "ApplicationLatencyP90": 60,
      "ApplicationLatencyP99": 60,
      "ApplicationLatencyP95": 60,
      "ApplicationLatencyP85": 60,
      "CPUUser": 60,
      "ApplicationRequests2xx": 60,
      "CPUIdle": 60,
      "ApplicationLatencyP50": 60,
      "ApplicationRequestsTotal": 60,
      "RootFilesystemUtil": 60,
      "LoadAverage1min": 60,
      "CPUIrq": 60,
      "CPUNice": 60,
      "CPUIowait": 60,
      "ApplicationLatencyP10": 60,
      "LoadAverage5min": 60,
      "ApplicationRequests5xx": 60,
      "ApplicationLatencyP75": 60,
      "CPUSystem": 60,
      "ApplicationRequests3xx": 60,
      "ApplicationRequests4xx": 60,
      "InstanceHealth": 60,
      "CPUSoftirq": 60
    }
  },
  "Version": 1
}
```

Utilisation des rapports améliorés sur l'état à l'aide de l'API Elastic Beanstalk

Étant donné que les rapports d'état AWS Elastic Beanstalk améliorés ont des exigences en matière de rôles et de solutions, vous devez mettre à jour les scripts et le code que vous utilisiez avant la publication des rapports d'état améliorés avant de pouvoir les utiliser. Pour assurer la rétrocompatibilité, les rapports améliorés sur l'état ne sont pas activés par défaut lorsque vous créez un environnement à l'aide de l'API Elastic Beanstalk.

Vous configurez des rapports de santé améliorés en définissant le rôle de service, le profil d'instance et les options CloudWatch de configuration Amazon pour votre environnement. Vous pouvez le faire de trois façons : en définissant les options de configuration dans le dossier `.ebextensions`, avec des configurations enregistrées ou en les configurant directement dans le paramètre `create-environment` de l'appel `option-settings`.

Pour utiliser l'API ou l'interface de ligne de AWS commande (CLI) afin de créer un environnement prenant en charge l'amélioration de la santé, vous devez : SDKs

- Créez un rôle de service et un profil d'instance avec les autorisations appropriées.
- Créez un nouvel environnement avec une nouvelle version de plateforme
- Définissez les options de configuration du type de système d'état, du profil d'instance et du rôle de service.

Utilisez les options de configuration suivantes dans les espaces de noms `aws:elasticbeanstalk:healthreporting:system`, `aws:autoscaling:launchconfiguration` et `aws:elasticbeanstalk:environment` afin de configurer votre environnement pour les rapports améliorés sur l'état.

Options de configuration des rapports améliorés sur l'état

SystemType

Espace de nom : `aws:elasticbeanstalk:healthreporting:system`

Pour activer les rapports améliorés sur l'état, définissez l'option sur **enhanced**.

IamInstanceProfile

Espace de nom : `aws:autoscaling:launchconfiguration`

Choisissez le nom d'un profil d'instance configuré pour être utilisé avec Elastic Beanstalk.

ServiceRole

Espace de nom : `aws:elasticbeanstalk:environment`

Choisissez le nom d'un rôle de service configuré pour être utilisé avec Elastic Beanstalk.

ConfigDocument (facultatif)

Espace de nom : `aws:elasticbeanstalk:healthreporting:system`

Document JSON qui définit les métriques d'instance et d'environnement sur lesquelles publier CloudWatch. Par exemple :

```
{
  "CloudWatchMetrics":
    {
    "Environment":
      {
      "ApplicationLatencyP99.9":60,
      "InstancesSevere":60
      }
    "Instance":
      {
      "ApplicationLatencyP85":60,
      "CPUUser": 60
      }
    }
  "Version":1
}
```

> **Note**
>
> Les documents de configuration peuvent exiger une mise en forme spéciale, comme des guillemets d'échappement, en fonction de la façon dont vous les fournissez à Elastic Beanstalk. Pour obtenir des exemples, consultez Fourniture des documents de configuration des métriques personnalisées.

Format de journal d'intégrité améliorée

AWS Elastic Beanstalk les plateformes utilisent un format de journal de serveur Web personnalisé pour transmettre efficacement les informations relatives aux requêtes HTTP au système amélioré de rapports de santé. Le système analyse les journaux, identifie les problèmes et définit en conséquence l'état de santé de l'instance et de l'environnement. Si vous désactivez le proxy de serveur web dans votre environnement et que vous traitez les demandes directement depuis le conteneur web, vous pouvez toujours utiliser pleinement les rapports améliorés sur l'état en configurant votre serveur de sorte à générer des journaux à l'emplacement et au format utilisés par l'agent de vérification de l'état Elastic Beanstalk.

> **Note**
>
> Les informations de cette page concernent uniquement les plateformes Linux. Sur la plateforme Windows Server, Elastic Beanstalk reçoit les informations sur les demandes HTTP directement à partir du serveur web IIS. Pour en savoir plus, consultez Capture des métriques du serveur web dans IIS sous Windows Server.

Configuration de journal de serveur web

Les plateformes Elastic Beanstalk sont configurées de sorte à générer deux journaux contenant des informations sur les demandes HTTP. La première est au format détaillé et fournit des informations complètes sur la demande, y compris les informations de l'agent utilisateur du demandeur et un horodatage contrôlable de visu.

/var/log/nginx/access.journal

L'exemple suivant provient d'un proxy nginx exécuté dans un environnement de serveur web Ruby, mais le format est similaire pour Apache.

```
172.31.24.3 - - [23/Jul/2015:00:21:20 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0
 (x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23
 librtmp/2.3" "177.72.242.17"
172.31.24.3 - - [23/Jul/2015:00:21:21 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0
 (x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23
 librtmp/2.3" "177.72.242.17"
```

```
172.31.24.3 - - [23/Jul/2015:00:21:22 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0
 (x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23
 librtmp/2.3" "177.72.242.17"
172.31.24.3 - - [23/Jul/2015:00:21:22 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0
 (x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23
 librtmp/2.3" "177.72.242.17"
172.31.24.3 - - [23/Jul/2015:00:21:22 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0
 (x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23
 librtmp/2.3" "177.72.242.17"
```

Le deuxième journal est au format court. Il comporte des informations pertinentes uniquement pour la création de rapports d'intégrité améliorée. Ce journal est sorti vers un sous-dossier nommé `healthd` et tourne chaque heure. Les anciens journaux sont supprimés immédiatement après la rotation.

/var/log/nginx/healthd/application.log.2015-07-23-00

L'exemple suivant montre un journal au format lisible par la machine.

```
1437609879.311"/"200"0.083"0.083"177.72.242.17
1437609879.874"/"200"0.347"0.347"177.72.242.17
1437609880.006"/bad/path"404"0.001"0.001"177.72.242.17
1437609880.058"/"200"0.530"0.530"177.72.242.17
1437609880.928"/bad/path"404"0.001"0.001"177.72.242.17
```

Le format des journaux d'intégrité améliorée inclut les informations suivantes :

- Le moment de la demande, en heure Unix
- Le chemin d'accès de la demande
- Le code de statut HTTP pour le résultat
- La durée des demandes
- Le temps en amont
- L'en-tête HTTP `X-Forwarded-For`

Pour les proxys nginx, les heures sont indiquées en secondes à virgule flottante, avec trois décimales. Pour Apache, les millisecondes entières sont utilisées.

> **Note**
>
> Si un avertissement similaire au suivant s'affiche dans un fichier journal, où `DATE-TIME` correspond à une date et une heure, et que vous utilisez un proxy personnalisé, comme dans un environnement Docker multi-conteneurs, vous devez utiliser un fichier .ebextension pour configurer votre environnement afin que `healthd` puisse lire vos fichiers journaux :
>
> ```
> W, [DATE-TIME #1922] WARN -- : log file "/var/log/nginx/healthd/
> application.log.DATE-TIME" does not exist
> ```
>
> Vous pouvez commencer par le fichier .ebextension dans l'exemple Docker multi-conteneurs.

/etc/nginx/conf.d/webapp_healthd.conf

L'exemple suivant montre la configuration de journal pour nginx avec le format de journal `healthd` mis en évidence.

```
upstream my_app {
  server unix:///var/run/puma/my_app.sock;
}

log_format healthd '$msec"$uri"'
                '$status"$request_time"$upstream_response_time"'
                '$http_x_forwarded_for';

server {
  listen 80;
  server_name _ localhost; # need to listen to localhost for worker tier

  if ($time_iso8601 ~ "^(\d{4})-(\d{2})-(\d{2})T(\d{2})") {
    set $year $1;
    set $month $2;
    set $day $3;
    set $hour $4;
  }

  access_log  /var/log/nginx/access.log  main;
  access_log /var/log/nginx/healthd/application.log.$year-$month-$day-$hour healthd;

  location / {
```

```
    proxy_pass http://my_app; # match the name of upstream directive which is defined
 above
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
  }

  location /assets {
    alias /var/app/current/public/assets;
    gzip_static on;
    gzip on;
    expires max;
    add_header Cache-Control public;
  }

  location /public {
    alias /var/app/current/public;
    gzip_static on;
    gzip on;
    expires max;
    add_header Cache-Control public;
  }
}
```

/etc/httpd/conf.d/healthd.conf

L'exemple suivant montre la configuration des journaux pour Apache.

```
LogFormat "%{%s}t\"%U\"%s\"%D\"%D\"%{X-Forwarded-For}i" healthd
CustomLog "|/usr/sbin/rotatelogs /var/log/httpd/healthd/application.log.%Y-%m-%d-%H
 3600" healthd
```

Génération de journaux pour la création de rapports d'intégrité améliorée

Pour fournir des journaux à l'agent d'intégrité, vous devez procéder comme suit :

- Sortir des journaux dans le bon format, comme illustré dans la section précédente
- Sortir des journaux dans `/var/log/nginx/healthd/`
- Nommer des journaux à l'aide du format suivant : `application.log.$year-$month-$day-$hour`
- Effectuer une rotation des journaux une fois par heure
- Ne pas tronquer de journaux

Notifications et dépannage

> **Essayez Amazon Q Developer CLI pour un dépannage assisté par l'IA**
>
> La CLI Amazon Q Developer peut vous aider à résoudre rapidement les problèmes d'environnement. La Q CLI fournit des solutions en vérifiant l'état de l'environnement, en examinant les événements, en analysant les journaux et en posant des questions de clarification. Pour plus d'informations et des instructions détaillées, consultez la section Résolution des problèmes liés aux environnements Elastic Beanstalk avec Amazon Q Developer CLI dans les blogs. AWS

Cette page répertorie les messages relatifs aux problèmes courants ainsi que des liens vers des informations supplémentaires. Les messages apparaissent dans Volet de présentation de l'environnement la console Elastic Beanstalk et sont enregistrés lors d'événements où des problèmes de santé persistent après plusieurs vérifications.

Déploiements

Elastic Beanstalk surveille la cohérence de votre environnement à la suite de déploiements. En cas de défaillance d'un déploiement propagé, la version de votre application s'exécutant sur les instances de votre environnement peut varier. Cela peut se produire si un déploiement réussit sur un ou plusieurs lots mais échoue avant que tous les lots aient abouti.

Une version d'application incorrecte a été détectée sur 2 instances sur 5. Version attendue « v1 » (déploiement 1).

Version d'application incorrecte sur les instances de l'environnement. Version attendue « v1 » (déploiement 1).

La version d'application attendue ne s'exécute pas sur tout ou partie des instances dans un environnement.

Version d'application « v2 » incorrecte (déploiement 2). Version attendue « v1 » (déploiement 1).

L'application déployée sur une instance diffère de la version attendue. En cas de défaillance d'un déploiement, la version attendue revient à la version du dernier déploiement ayant abouti. Dans l'exemple ci-dessus, le premier déploiement (version « v1 ») a abouti, mais le deuxième déploiement (version « v2 ») a échoué. Toutes les instances exécutant « v2 » sont considérées comme défectueuses.

Pour résoudre ce problème, démarrez un autre déploiement. Vous pouvez redéployer une version précédente dont vous savez qu'elle fonctionne, ou configurer votre environnement pour ignorer les vérifications de l'état au cours du déploiement et redéployer la nouvelle version pour forcer le déploiement à aboutir.

Vous pouvez également identifier et résilier les instances qui exécutent la mauvaise version d'application. Elastic Beanstalk lance des instances avec la version appropriée pour remplacer toutes les instances que vous résiliez. Utilisez la commande d'état de l'interface de ligne de commande EB pour identifier les instances qui exécutent la mauvaise version d'application.

Serveur d'application

15 % des demandes signalent une erreur avec HTTP 4xx

20 % des demandes à ELB signalent une erreur avec HTTP 4xx.

Un pourcentage élevé de demandes HTTP à une instance ou un environnement échouent avec des erreurs 4xx.

Un code de statut de série 400 indique que l'utilisateur a soumis une demande erronée, telle que la demande d'une page qui n'existe pas (404 : Fichier introuvable) ou à laquelle l'utilisateur n'a pas accès à (403 Interdit). Un petit nombre d'erreurs 404 n'est pas rare, mais un grand nombre pourrait signifier qu'il y a des liens internes ou externes vers des pages indisponibles. Ces problèmes peuvent être résolus en réparant des liens internes erronés et en ajoutant des redirections pour des liens externes erronés.

5 % des demandes échouent avec HTTP 5xx

3 % des demandes à ELB échouent avec HTTP 5xx.

Un pourcentage élevé de demandes HTTP à une instance ou un environnement échoue avec les codes de statut de série 500.

Un code de statut de série 500 indique que le serveur d'applications a rencontré une erreur interne. Ces problèmes indiquent qu'il y a une erreur dans le code de votre application et ils doivent être identifiés et corrigés rapidement.

95 % de l'UC est en cours d'utilisation

Sur une instance, l'agent d'état rapporte un pourcentage extrêmement élevé d'utilisation de l'UC et définit l'état de l'instance sur Avertissement ou Dégradé.

Mettez à l'échelle votre environnement pour réduire la charge des instances.

Instance de travail

20 messages en attente dans la file d'attente (il y a 25 secondes)

Des demandes sont ajoutées à la file d'attente de votre environnement de travail plus vite qu'elles ne peuvent être traitées. Mettez à l'échelle votre environnement pour accroître la capacité.

5 messages dans la file d'attente de lettres mortes (il y a 15 secondes)

Des demandes de travail échouent régulièrement et sont ajoutées à la the section called “Files d’attente de lettres mortes”. Vérifiez les demandes dans la file d'attente de lettres mortes pour voir pourquoi elles échouent.

Autres ressources

4 instances actives est inférieur à la taille minimale 5 du groupe Auto Scaling

Le nombre d'instances s'exécutant dans votre environnement est inférieur au nombre minimal configuré pour le groupe Auto Scaling.

Des notifications du groupe Auto Scaling (nom du groupe) ont été supprimées ou modifiées

Les notifications configurées pour votre groupe Auto Scaling ont été modifiées en dehors d'Elastic Beanstalk.

Analyse de l'environnement basée sur l'IA

AWS Elastic Beanstalk L'analyse basée sur l'IA identifie les causes profondes et recommande des solutions aux problèmes de santé liés à l'environnement. Lorsque votre environnement rencontre des problèmes, vous pouvez demander une analyse de l'IA à l'aide des opérations d'`RetrieveEnvironmentInfo`API `RequestEnvironmentInfo` et de type `analyze` info pour obtenir des informations générées par l'IA et des solutions recommandées.

> **Note**
>
> L'analyse par IA n'est disponible que sur les versions compatibles d'Amazon Linux 2 et de la AL2023 plateforme publiées le 16 février 2026 ou après cette date.

Comment ça marche

Lorsque vous demandez une analyse basée sur l'IA, Elastic Beanstalk exécute un script sur une instance de votre environnement qui collecte les événements récents, l'état de santé de l'instance et les journaux (jusqu'à 170 000 jetons de données). Il envoie ensuite ces données à Amazon Bedrock dans votre compte et renvoie des informations et des recommandations sur les prochaines étapes.

Conditions préalables

Avant d'utiliser l'analyse par IA, vérifiez que votre environnement répond aux exigences suivantes :

- Environnement exécutant une version de plate-forme prise en charge
- Profil d'instance avec les autorisations requises (voir Autorisations requises ci-dessous)
- Détails des cas d'utilisation anthropiques — L'analyse de l'IA utilise les modèles Anthropic Claude via Amazon Bedrock. Anthropic vous demande de soumettre un formulaire détaillé de cas d'utilisation unique avant de pouvoir invoquer ses modèles. Pour soumettre ce formulaire, sélectionnez n'importe quel modèle Anthropic dans le catalogue de modèles de la console Amazon Bedrock ou appelez l'`PutUseCaseForModelAccess`API. Vous ne devez effectuer cette opération qu'une seule fois par AWS compte. Si vous soumettez le formulaire depuis le compte de gestion des AWS Organizations, il couvre automatiquement tous les comptes membres de l'organisation. Pour plus d'informations, consultez Access Amazon Bedrock foundation models.
- GovCloud régions — Si vous utilisez des régions AWS GovCloud (États-Unis), vous devez autoriser l'accès au dernier modèle Anthropic Claude Sonnet and/or Opus dans Amazon Bedrock avant d'utiliser l'analyse par IA. Pour obtenir des instructions sur l'activation de l'accès aux modèles dans GovCloud les régions, consultez Gérer l'accès aux modèles Amazon Bedrock Foundation. Pour plus d'informations sur le dernier modèle Anthropic Claude Sonnet and/or Opus disponible, voir Régions et modèles pris en charge pour les profils d'inférence.

Autorisations requises

Pour utiliser l'analyse par IA, le profil d'instance Amazon EC2 de votre environnement doit être autorisé à invoquer Amazon Bedrock. Ajoutez les autorisations suivantes à votre profil d'instance :

- `bedrock:InvokeModel`
- `bedrock:ListFoundationModels`
- `elasticbeanstalk:DescribeEvents`
- `elasticbeanstalk:DescribeEnvironmentHealth`

Pour plus d'informations sur la configuration des profils d'instance, consultezGestion des profils d'instance Elastic Beanstalk.

Utilisation de l'analyse de l'IA dans la console

À partir de l'aperçu de l'environnement

Lorsque l'état de santé de votre environnement est Avertissement, Dégradé ou Sévère, un bouton d'analyse de l'IA apparaît dans la section de présentation de l'environnement. Cliquez sur ce bouton pour lancer une analyse IA de votre environnement.

Depuis la page Logs

Vous pouvez également accéder à l'analyse de l'IA depuis la page Logs du volet de navigation. Cliquez sur le bouton Analyse de l'IA pour demander une analyse basée sur l'IA de l'état actuel de votre environnement.

Utilisation de l'analyse basée sur l'IA avec AWS CLI

Vous pouvez utiliser l'API Elastic Beanstalk via le pour demander et récupérer des analyses d'IA AWS CLI par programmation.

Demander une analyse de l'IA

Utilisez l'`RequestEnvironmentInfo`opération avec le `InfoType` paramètre défini sur`analyze`.

Example AWS CLI - Demander une analyse de l'IA

```
aws elasticbeanstalk request-environment-info \
    --environment-name my-env \
    --info-type analyze \
    --region us-east-1
```

Récupérez une analyse d'IA

Utilisez l'`RetrieveEnvironmentInfo`opération avec le `InfoType` paramètre défini sur `analyze` pour récupérer les résultats de l'analyse.

Example AWS CLI - Récupérez l'analyse de l'IA

```
aws elasticbeanstalk retrieve-environment-info \
```

```
    --environment-name my-env \
    --info-type analyze \
    --region us-east-1
```

La réponse inclut une analyse de l'état actuel de l'environnement générée par l'IA, ainsi que des solutions recommandées pour tout problème identifié.

Utilisation de l'analyse de l'IA avec l'EB CLI

Si vous utilisez l'EB CLI, vous pouvez demander une analyse de l'IA avec l'option `--analyze` (`-ai`) de la eb logs commande. La commande demande l'analyse, attend qu'elle soit terminée et affiche les résultats.

Example EB CLI - Demander une analyse de l'IA

```
$ eb logs --analyze
```

L'`--analyze`option n'est pas compatible avec `--instance`--`all`,`--zip`, ou`--log-group`. Pour la référence complète des commandes, voireb logs.

> **Note**
>
> L'`--analyze`option nécessite la version 3.27 ou ultérieure de l'EB CLI.

Considérations importantes

- Tarification — L'analyse basée sur l'IA utilise Amazon Bedrock pour traiter les données de votre environnement, et la tarification standard d'Amazon Bedrock s'applique aux appels de modèles. Pour plus de détails sur la tarification, consultez Tarification Amazon Bedrock.
- Configuration requise : l'analyse de l'IA n'est disponible que sur Amazon Linux 2 et sur les versions AL2023 basées sur la plateforme publiées le 16 février 2026 ou après cette date. Pour utiliser cette fonctionnalité, mettez à jour votre environnement vers une version de plate-forme prise en charge. Pour de plus amples informations, veuillez consulter Mise à jour de la version de la plateforme de votre environnement Elastic Beanstalk.
- Autorisations — Avant d'utiliser l'analyse par IA, assurez-vous que votre profil d'instance dispose des autorisations Amazon Bedrock

(`bedrock:InvokeModeletbedrock:ListFoundationModels`) et des autorisations Elastic `elasticbeanstalk:DescribeEvents` Beanstalk (et) requises. `elasticbeanstalk:DescribeEnvironmentHealth`

- Confidentialité des données — L'analyse envoie les événements environnementaux et les journaux à Amazon Bedrock dans votre compte pour traitement. Pour plus d'informations sur la manière dont Amazon Bedrock gère vos données, consultez Amazon Bedrock Security and Compliance.
- Quotas de service — L'analyse par IA utilise un modèle Anthropic Claude dans Amazon Bedrock, qui a des quotas par défaut pour les demandes par minute et les jetons par minute. Si vous rencontrez des erreurs de régulation, vous pouvez demander une augmentation du quota. Pour plus d'informations, consultez Demande d'augmentation de quota.

Versions de plateforme prises en charge

L'analyse par IA est prise en charge sur Amazon Linux 2 et AL2023 sur les versions basées sur la plateforme publiées le 16 février 2026 ou après cette date. Pour vérifier la version de votre plateforme, consultez les notes de mise à jour d'Elastic Beanstalk.

Gestion des alarmes

Cette rubrique explique les étapes à suivre pour créer des alarmes pour les indicateurs que vous surveillez. Il fournit également des instructions pour consulter vos alarmes existantes et vérifier leur état.

Vous pouvez créer des alarmes pour les métriques que vous surveillez en utilisant la console Elastic Beanstalk. Les alarmes vous aident à surveiller les modifications apportées à votre AWS Elastic Beanstalk environnement afin que vous puissiez facilement identifier et atténuer les problèmes avant qu'ils ne surviennent. Par exemple, vous pouvez définir une alarme qui vous informe lorsque l'utilisation de l'UC dans un environnement dépasse un certain seuil, veiller à ce que être notifié avant qu'un problème potentiel se produise. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon CloudWatch.

> **Note**
>
> Elastic CloudWatch Beanstalk l'utilise pour la surveillance et les alarmes CloudWatch , ce qui signifie que des coûts sont appliqués AWS à votre compte pour toutes les alarmes que vous utilisez.

Pour plus d'informations sur la surveillance des métriques spécifiques, consultez Création de rapports d'intégrité de base.

Pour vérifier l'état de vos alarmes

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, cliquez sur Alarms (alertes).

   La page affiche une liste d'alarmes existantes. Si des alarmes sont en état d'alarme, elles sont signalées par l'icône d'avertissement ( ).
4. Pour filtrer les alarmes, choisissez le menu déroulant, puis sélectionnez un filtre.
5. Pour modifier ou supprimer une alarme, choisissez respectivement l'icône de modification ( ) ou l'icône de suppression ( ).

Pour créer une alarme

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Surveillance.
4. Localisez la métrique pour laquelle vous souhaitez créer une alarme, puis choisissez l'icône d'alarme ( ).

   La page Add alarm (Ajouter une alarme) s'affiche.
5. Entrez les détails relatifs à l'alarme :

   - Nom : un nom pour cette alarme.
   - Description (facultatif) : une brève description de ce qu'est cette alarme.

- Période : l'intervalle de temps entre les lectures.
- Seuil : décrit le comportement et la valeur que la métrique doit dépasser afin de déclencher une alarme.
- Changer l'état après : le délai après qu'un seuil a été dépassé qui déclenche une modification de l'état de l'alarme.
- Notifier : rubrique Amazon SNS qui est notifiée lorsqu'une alarme change d'état.
- Notifier quand l'état passe à :
  - OK : La métrique se trouve dans le seuil défini.
  - Alarme : la métrique a dépassé le seuil défini.
  - Données insuffisantes : l'alarme vient de démarrer, la métrique n'est pas disponible, ou la quantité de données n'est pas suffisante pour permettre à la métrique de déterminer le statut de l'alarme.

6. Choisissez Ajouter. L'état de l'environnement passe au gris pendant la mise à jour de l'environnement. Vous pouvez afficher l'alarme que vous avez créée en choisissant Alarms (Alarmes) dans le volet de navigation.

Affichage de l'historique des modifications d'un environnement Elastic Beanstalk

Cette rubrique explique comment utiliser la console Elastic Beanstalk pour consulter l'historique des modifications de configuration apportées à vos environnements Elastic Beanstalk.

Elastic Beanstalk récupère votre historique des modifications à partir d'événements enregistrés dans AWS CloudTrail et les affiche dans une liste que vous pouvez facilement parcourir et filtrer.

Le panneau Change History (Historique des modifications) affiche les informations suivantes concernant les modifications apportées à vos environnements :

- la date et l'heure auxquelles une modification a été apportée ;
- l'utilisateur IAM responsable d'une modification ;
- l'outil source (l'interface de ligne de commande Elastic Beanstalk (CLI EB) ou la console) utilisé pour effectuer la modification ;
- le paramètre de configuration et les nouvelles valeurs qui ont été définies.

Les données sensibles faisant partie de la modification, telles que les noms des utilisateurs de base de données concernés par la modification, n'apparaissent pas dans le panneau.

Pour afficher l'historique des modifications

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, sélectionnez Change history (Historique des modifications).

   La page Change History (Historique des modifications) affiche une liste des modifications de configuration apportées à vos environnements Elastic Beanstalk.

Notez les points suivants concernant la navigation dans les informations de cette page :

- Vous pouvez parcourir la liste en sélectionnant **<** (précédent) ou **>** (suivant) ou en sélectionnant un numéro de page spécifique.
- Sous la colonne Configuration changes (Modifications de configuration), sélectionnez l'icône flèche pour basculer entre le développement et la réduction de la liste des modifications sous l'en-tête Changes made (Modifications apportées).
- Utilisez la barre de recherche pour filtrer vos résultats à partir de la liste de l'historique des modifications. Vous pouvez saisir n'importe quelle chaîne pour affiner la liste des modifications qui s'affichent.

Remarques sur le filtrage des résultats affichés :

- Le filtre de recherche n'est pas sensible à la casse.
- Vous pouvez filtrer les modifications affichées en fonction des informations figurant dans la colonne Configuration changes (Modifications de configuration), même si elles ne sont pas visibles parce qu'elles sont réduites dans Changes made (Modifications apportées).
- Vous ne pouvez filtrer que les résultats affichés. Toutefois, le filtre ne change pas même si vous sélectionnez le bouton pour accéder à une autre page et afficher plus de résultats. Vos résultats filtrés s'ajoutent également au jeu de résultats de la page suivante.

Les exemples suivants montrent comment filtrer les données affichées sur l'écran précédent :

- Entrez **`GettingStartedApp-env`** dans le champ de recherche pour affiner les résultats afin d'inclure uniquement les modifications apportées à l'environnement nommé GettingStartedApp-env.
- Saisissez **`example3`** dans la zone de recherche pour affiner les résultats et inclure uniquement les modifications apportées par les utilisateurs IAM dont le nom d'utilisateur contient la chaîne exemple3.
- Saisissez **`2020-10`** dans la zone de recherche pour affiner les résultats et inclure uniquement les modifications apportées au cours du mois d'octobre 2020. Définissez la valeur de recherche sur **`2020-10-16`** pour filtrer davantage les résultats affichés et inclure uniquement les modifications effectuées le jour du 16 octobre 2020.
- Saisissez **`proxy:staticfiles`** dans la zone de recherche pour affiner les résultats et inclure uniquement les modifications apportées à l'espace de nom nommé aws:elasticbeanstalk:environment:proxy:staticfiles. Les lignes affichées sont le résultat du filtre. Cela est vrai même pour les résultats qui sont réduits sous Changes made (Modifications apportées).

Affichage du flux d'événements d'un environnement Elastic Beanstalk

Cette rubrique explique comment accéder aux événements et aux notifications associés à votre application.

Afficher des événements avec la console Elastic Beanstalk

Pour consulter les événements avec la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, sélectionnez Events.

   La page Événements affiche une liste de tous les événements enregistrés pour l'environnement. Vous pouvez parcourir la liste en choisissant **<** (précédent), **>** (suivant) ou les numéros de page. Vous pouvez filtrer selon le type d'événement affiché à l'aide de la liste déroulante Severity (Gravité).

Affichage des événements à l'aide des outils de ligne de commande

L'interface de ligne de commande EB et l'AWS CLI fournissent toutes les deux des commandes permettant de récupérer les événements. Si vous gérez votre environnement via l'interface de ligne de commande EB, utilisez eb events pour imprimer la liste des événements. Cette commande inclut également une option `--follow`, qui continue d'afficher les nouveaux événements jusqu'à ce que vous appuyiez sur Ctrl+C pour arrêter la sortie.

Pour extraire des événements à l'aide de AWS CLI, utilisez la `describe-events` commande et spécifiez l'environnement par son nom ou son identifiant :

```
$ aws elasticbeanstalk describe-events --environment-id e-gbjzqccra3
{
    "Events": [
        {
            "ApplicationName": "elastic-beanstalk-example",
            "EnvironmentName": "elasticBeanstalkExa-env",
            "Severity": "INFO",
            "RequestId": "a4c7bfd6-2043-11e5-91e2-9114455c358a",
            "Message": "Environment update completed successfully.",
            "EventDate": "2015-07-01T22:52:12.639Z"
        },
...
```

Pour plus d'informations sur les outils de ligne de commande, consultez???.

Affichage de la liste des instances de serveur et connexion à ces instances

Cette rubrique explique comment consulter la liste des EC2 instances Amazon exécutant votre environnement applicatif Elastic Beanstalk et comment s'y connecter.

Vous pouvez consulter la liste des EC2 instances Amazon exécutant votre environnement AWS Elastic Beanstalk applicatif via la console Elastic Beanstalk. Vous pouvez vous connecter aux instances en utilisant tout client SSH. Vous pouvez vous connecter aux instances exécutant Windows à l'aide de Bureau à distance.

> **Important**
>
> Avant de pouvoir accéder à vos instances Amazon EC2 fournies par Elastic Beanstalk, vous devez créer une EC2 paire de clés Amazon et configurer votre Amazon Elastic Beanstalk provisionnée pour utiliser la paire de clés Amazon. EC2instances EC2 Vous pouvez configurer vos paires de EC2 clés Amazon à l'aide de la console AWS de gestion. Pour obtenir des instructions sur la création d'une paire de clés pour Amazon EC2, consultez le guide de EC2 démarrage Amazon. Pour plus d'informations sur la façon de configurer vos EC2 instances Amazon pour utiliser une paire de EC2 clés Amazon, consultezEC2 paire de clés.
>
> Par défaut, Elastic Beanstalk n'active pas les EC2 connexions à distance aux instances d'un conteneur Windows, à l'exception de celles des anciens conteneurs Windows. (Elastic EC2 Beanstalk configure les instances des anciens conteneurs Windows pour utiliser le port 3389 pour les connexions RDP.) Vous pouvez activer les connexions à distance à vos EC2 instances exécutant Windows en ajoutant une règle à un groupe de sécurité qui autorise le trafic entrant vers les instances. Nous vous recommandons vivement de supprimer la règle lorsque vous mettez fin à votre connexion à distance. Vous pouvez ajouter la règle à nouveau la prochaine fois que vous avez besoin de vous connecter à distance. Pour de plus amples informations, veuillez consulter Ajout d'une règle pour le trafic RDP entrant vers une instance Windows et Connexion à votre instance Windows dans le Guide de l'utilisateur Amazon Elastic Compute Cloud pour Microsoft Windows.

Pour consulter les EC2 instances Amazon d'un environnement et s'y connecter

1. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans le panneau de navigation de la console, choisissez Équilibreurs de charge.
3. Les équilibreurs de charge créés par Elastic Beanstalk ont awseb dans le nom. Recherchez l'équilibreur de charge pour votre environnement et cliquez dessus.
4. Dans le volet inférieur de la console, choisissez l'onglet Instances.

   Une liste des instances qu'utilise l'équilibreur de charge pour votre environnement Elastic Beanstalk s'affiche. Notez un ID d'instance auquel vous souhaitez vous connecter.
5. Dans le volet de navigation de la EC2 console Amazon, choisissez Instances et recherchez l'ID de votre instance dans la liste.

6. Cliquez avec le bouton droit sur l'ID d' EC2 instance de l'instance Amazon exécutée dans l'équilibreur de charge de votre environnement, puis sélectionnez Connect dans le menu contextuel.
7. Notez adresse DNS publique de l'instance sur l'onglet Description.
8. Connectez-vous à une instance exécutant Linux à l'aide du client SSH de votre choix, puis tapez ssh -i .ec2/mykeypair.pem ec2-User@<public-DNS- >. of-the-instance

Pour plus d'informations sur la connexion à une instance Amazon EC2 Linux, consultez Getting Started with Amazon EC2 Linux Instances dans le guide de EC2 l'utilisateur Amazon.

Si votre environnement Elastic Beanstalk utilise la plateforme .NET on Windows Server, consultez Getting Started with EC2 Amazon Windows Instances dans le guide de l'utilisateur Amazon. EC2

Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk

Cette rubrique décrit les types de journaux d'instance fournis par Elastic Beanstalk. Il fournit également des instructions détaillées pour les récupérer et les gérer.

Les instances Amazon EC2 de votre environnement Elastic Beanstalk génèrent des journaux que vous pouvez afficher pour résoudre les problèmes avec vos fichiers de configuration ou d'application. Les journaux sont créés par le serveur Web, le serveur d'applications, les scripts de la plateforme Elastic Beanstalk et sont stockés localement sur CloudFormation des instances individuelles. Vous pouvez les récupérer facilement avec la console de gestion d'environnement ou l'interface de ligne de commande EB. Vous pouvez également configurer votre environnement pour diffuser les journaux sur Amazon CloudWatch Logs en temps réel.

Les journaux des processus sont les 100 dernières lignes des fichiers journaux les plus couramment utilisés : les journaux opérationnels Elastic Beanstalk et les journaux provenant du serveur web ou du serveur d'applications. Lorsque vous demandez des journaux de queue dans la console de gestion d'environnement ou avec eb logs, une instance dans votre environnement concatène les entrées du journal les plus récentes dans un fichier texte unique et les télécharge sur Amazon S3.

Les journaux de groupe sont des journaux complets pour un plus large éventail de fichiers journaux, y compris des journaux yum et cron et plusieurs journaux CloudFormation. Lorsque vous demandez des journaux de groupe, une instance de votre environnement rassemble les fichiers journaux complets dans une archive ZIP et les télécharge sur Amazon S3.

Pour télécharger les journaux soumis à rotation sur Amazon S3, les instances de votre environnement doivent avoir un profil d'instance avec l'autorisation d'écrire sur votre compartiment Elastic Beanstalk Amazon S3. Ces autorisations sont incluses dans le profil d'instance par défaut qu'Elastic Beanstalk vous invite à créer lorsque vous lancez un environnement dans la console Elastic Beanstalk pour la première fois.

Pour récupérer des journaux d'instance

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, sélectionnez Logs (Journaux).
4. Choisissez Request Logs (Journaux de demande), puis choisissez le type de journaux à récupérer. Pour obtenir des journaux de processus, choisissez Last 100 Lines (100 dernières lignes). Pour obtenir des journaux de bundle, choisissez Full Logs (Journaux complets).
5. Quand Elastic Beanstalk a fini de récupérer vos journaux, choisissez Download (Télécharger).

Elastic Beanstalk stocke les journaux de queue et de bundle dans un compartiment Amazon S3 et génère une URL Amazon S3 présignée que vous pouvez utiliser pour accéder à vos journaux. Elastic Beanstalk supprime les fichiers d'Amazon S3 après une durée de 15 minutes.

> **Warning**
>
> Quiconque possédant l'URL Amazon S3 pré-signée peut consulter les fichiers avant qu'ils ne soient supprimés. Faites en sorte que seules les parties approuvées aient accès à l'URL.

> **Note**
>
> Votre stratégie utilisateur doit disposer de l'autorisation `s3:DeleteObject`. Elastic Beanstalk utilise vos autorisations utilisateur pour supprimer les journaux d'Amazon S3.

Pour conserver des journaux, vous pouvez configurer votre environnement afin de publier les journaux dans Amazon S3 automatiquement après leur rotation. Pour activer la rotation des journaux dans Amazon S3, suivez la procédure présentée dans Configuration de l'affichage des journaux

d'instance. Les instances dans votre environnement essaient de télécharger des journaux qui ont fait l'objet d'une rotation une fois par heure.

Si votre application génère des journaux dans un emplacement qui ne fait pas partie de la configuration par défaut de la plateforme de votre environnement, vous pouvez étendre la configuration par défaut à l'aide des fichiers de configuration (`.ebextensions`). Vous pouvez ajouter les fichiers journaux de votre application aux journaux de processus, aux journaux de groupe ou à la rotation des journaux.

Pour le streaming des journaux en temps réel et le stockage à long terme, configurez votre environnement pour qu'il diffuse les journaux vers Amazon CloudWatch Logs.

Pour obtenir une analyse basée sur l'IA des journaux, des événements et de l'état de votre instance de votre environnement afin d'identifier les causes profondes des problèmes de santé et d'y apporter des solutions, consultez. Analyse de l'environnement basée sur l'IA

Sections

- Emplacement des journaux sur les instances Amazon EC2
- Emplacement des journaux dans Amazon S3
- Paramètres de rotation des journaux sous Linux
- Extension de la configuration de tâche de journal par défaut
- Streaming de fichiers journaux vers Amazon CloudWatch Logs

Emplacement des journaux sur les instances Amazon EC2

Les journaux sont stockés dans des emplacements standard des instances Amazon EC2 de votre environnement. Elastic Beanstalk génère les journaux suivants.

Amazon Linux 2

- `/var/log/eb-engine.log`

AMI Amazon Linux (AL1)

> **Note**
>
> Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations

sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2 ou Amazon Linux 2.

- `/var/log/eb-activity.log`
- `/var/log/eb-commandprocessor.log`

Windows Server

- `C:\Program Files\Amazon\ElasticBeanstalk\logs\`
- `C:\cfn\log\cfn-init.log`

Ces journaux contiennent des messages sur les activités de déploiement, y compris des messages liés aux fichiers de configuration (`.ebextensions`).

Chaque serveur d'applications et web stocke des journaux dans son propre dossier :

- Apache : `/var/log/httpd/`
- IIS : `C:\inetpub\wwwroot\`
- Node.js : `/var/log/nodejs/`
- nginx : `/var/log/nginx/`
- Passenger : `/var/app/support/logs/`
- Puma : `/var/log/puma/`
- Python : `/opt/python/log/`
- Tomcat : `/var/log/tomcat/`

Emplacement des journaux dans Amazon S3

Lorsque vous demandez les journaux de queue ou de bundle à partir de votre environnement, ou lorsque des instances téléchargent les journaux ayant fait l'objet d'une rotation, ils sont stockés dans votre compartiment Elastic Beanstalk d'Amazon S3. Elastic Beanstalk crée un `elasticbeanstalk-`*`region`*`-`*`account-id`* bucket nommé AWS pour chaque région dans laquelle vous créez des environnements. Dans ce compartiment, les journaux sont stockés dans le chemin d'accès `resources/environments/logs/`*`logtype`*`/`*`environment-id`*`/`*`instance-id`*.

Par exemple, les journaux de l'instance`i-0a1fd158`, dans l' AWS environnement Elastic `e-mpcwnwheky` Beanstalk `us-west-2` dans Region `123456789012` in account, sont stockés aux emplacements suivants :

- Journaux de queue :

  ```
  s3://elasticbeanstalk-us-west-2-123456789012/resources/environments/logs/tail/e-mpcwnwheky/i-0a1fd158
  ```

- Journaux de groupe :

  ```
  s3://elasticbeanstalk-us-west-2-123456789012/resources/environments/logs/bundle/e-mpcwnwheky/i-0a1fd158
  ```

- Journaux ayant fait l'objet d'une rotation :

  ```
  s3://elasticbeanstalk-us-west-2-123456789012/resources/environments/logs/publish/e-mpcwnwheky/i-0a1fd158
  ```

> **Note**
>
> Vous trouverez l'ID de votre environnement dans la console de gestion d'environnement.

Elastic Beanstalk supprime automatiquement les journaux de queue et de bundle d'Amazon S3 15 minutes après leur création. Les journaux faisant l'objet d'une rotation sont conservés jusqu'à ce que vous les supprimiez ou que vous les déplaciez vers Amazon Glacier.

Paramètres de rotation des journaux sous Linux

Sur les plates-formes Linux, Elastic Beanstalk utilise `logrotate` pour soumettre les journaux à rotation régulièrement. Si la configuration a été effectuée, une fois qu'un journal a effectué sa rotation localement, la tâche de rotation le sélectionne et le télécharge sur Amazon S3. Les journaux qui ont fait l'objet d'une rotation localement ne s'affichent pas dans les journaux de processus ou de groupe par défaut.

Vous pouvez trouver les fichiers de configuration Elastic Beanstalk pour `logrotate` dans `/etc/logrotate.elasticbeanstalk.hourly/`. Ces paramètres de rotation sont propres à la plateforme et sont susceptibles de changer dans de futures versions de la plateforme. Pour plus

d'informations concernant les paramètres disponibles et les exemples de configurations, exécutez `man logrotate`.

Les fichiers de configuration sont invoqués par des tâches cron dans `/etc/cron.hourly/`. Pour obtenir plus d'informations concernant `cron`, exécutez `man cron`.

Extension de la configuration de tâche de journal par défaut

Elastic Beanstalk utilise des fichiers des sous-répertoires de `/opt/elasticbeanstalk/tasks` (Linux) ou `C:\Program Files\Amazon\ElasticBeanstalk\config` (Windows Server) sur l'instance Amazon EC2 pour configurer les tâches pour les journaux de queue, les journaux de bundle et la rotation des journaux.

Sur Amazon Linux :

- Journaux de queue :

  ```
  /opt/elasticbeanstalk/tasks/taillogs.d/
  ```
- Journaux de groupe :

  ```
  /opt/elasticbeanstalk/tasks/bundlelogs.d/
  ```
- Journaux ayant fait l'objet d'une rotation :

  ```
  /opt/elasticbeanstalk/tasks/publishlogs.d/
  ```

Sous Windows Server :

- Journaux de queue :

  ```
  c:\Program Files\Amazon\ElasticBeanstalk\config\taillogs.d\
  ```
- Journaux de groupe :

  ```
  c:\Program Files\Amazon\ElasticBeanstalk\config\bundlelogs.d\
  ```
- Journaux ayant fait l'objet d'une rotation :

  ```
  c:\Program Files\Amazon\ElasticBeanstalk\config\publogs.d\
  ```

Par exemple, le fichier `eb-activity.conf` sous Linux ajoute deux fichiers journaux à la tâche de journaux de processus.

/opt/elasticbeanstalk/tasks/taillogs.d/eb-activity.conf

```
/var/log/eb-commandprocessor.log
/var/log/eb-activity.log
```

Vous pouvez utiliser les fichiers de configuration d'environnement (`.ebextensions`) pour ajouter vos propres fichiers `.conf` à ces dossiers. Un fichier `.conf` répertorie les fichiers journaux spécifiques à votre application, qu'Elastic Beanstalk ajoute aux tâches des fichiers journaux.

Utilisez la section `files` pour ajouter des fichiers de configuration aux tâches que vous voulez modifier. Par exemple, le texte de configuration suivant ajoute un fichier de configuration du journal à chaque instance dans votre environnement. Ce fichier de configuration du journal, `cloud-init.conf`, ajoute `/var/log/cloud-init.log` aux journaux de processus.

```
files:
  "/opt/elasticbeanstalk/tasks/taillogs.d/cloud-init.conf" :
    mode: "000755"
    owner: root
    group: root
    content: |
      /var/log/cloud-init.log
```

Ajoutez ce texte à un fichier avec l'extension de nom de fichier `.config` à votre bundle de fichiers source sous un dossier nommé `.ebextensions`.

```
~/workspace/my-app
|-- .ebextensions
|   `-- tail-logs.config
|-- index.php
`-- styles.css
```

Sur les plateforme Linux, vous pouvez également utiliser des caractères génériques dans les configurations de tâche de journal. Ce fichier de configuration ajoute tous les fichiers avec l'extension de nom de fichier `.log` provenant du dossier `log` situé à la racine de l'application aux journaux de bundle.

```
files:
  "/opt/elasticbeanstalk/tasks/bundlelogs.d/applogs.conf" :
    mode: "000755"
```

```
    owner: root
    group: root
    content: |
      /var/app/current/log/*.log
```

Les configurations de tâche de journal ne prennent pas en charge les caractères génériques sur les plateformes Windows.

> **Note**
>
> Pour vous familiariser avec les procédures de personnalisation des journaux, vous pouvez déployer un exemple d'application à l'aide de l'interface de ligne de commande EB. Pour cela, l'interface de ligne de commande EB crée un répertoire d'application local qui contient un sous-répertoire `.ebextentions` avec un exemple de configuration. Vous pouvez également utiliser les fichiers journaux de l'exemple d'application pour explorer la fonction d'extraction du journal décrite dans cette rubrique.

Pour de plus amples informations sur l'utilisation des fichiers de configuration, veuillez consulter Personnalisation d'environnement avancée avec fichiers de configuration (`.ebextensions`).

De la même manière que pour l'extension des journaux de processus et des journaux de groupe, vous pouvez étendre la rotation des journaux à l'aide d'un fichier de configuration. Chaque fois que Elastic Beanstalk fait pivoter ses propres journaux et les télécharge sur Amazon S3, il soumet également à rotation et télécharge vos journaux supplémentaires. L'extension de la rotation des journaux se comporte différemment en fonction du système d'exploitation utilisé par la plateforme. Les sections suivantes décrivent les deux cas possibles.

Extension de la rotation des journaux sous Linux

Comme expliqué dans Paramètres de rotation des journaux sous Linux, Elastic Beanstalk utilise `logrotate` pour soumettre les journaux à rotation sur les plateformes Linux. Lorsque vous configurez les fichiers journaux de votre application pour la rotation des fichiers, l'application n'a pas besoin de créer de copies des fichiers journaux. Elastic Beanstalk configure `logrotate` pour créer une copie des fichiers journaux de votre application pour chaque rotation. Par conséquent, l'application doit conserver les fichiers journaux déverrouillés lorsqu'elle n'écrit pas activement dans ces journaux.

Extension de la rotation des journaux sous Windows Server

Sur Windows Server, lorsque vous configurez les fichiers journaux de votre application pour la rotation, l'application doit effectuer une rotation régulière des fichiers journaux. Elastic Beanstalk recherche les fichiers dont le nom commence par le modèle que vous avez configuré et les sélectionne pour les charger vers Amazon S3. En outre, les points dans le nom du fichier sont ignorés et Elastic Beanstalk considère que le nom du fichier journal de base s'arrête au point.

Elastic Beanstalk charge toutes les versions d'un fichier journal de base, à l'exception de la plus récente qu'il considère comme le fichier journal actif de l'application, qui peut parfois être verrouillé. Votre application peut, par conséquent, garder le fichier journal actif verrouillé entre les rotations.

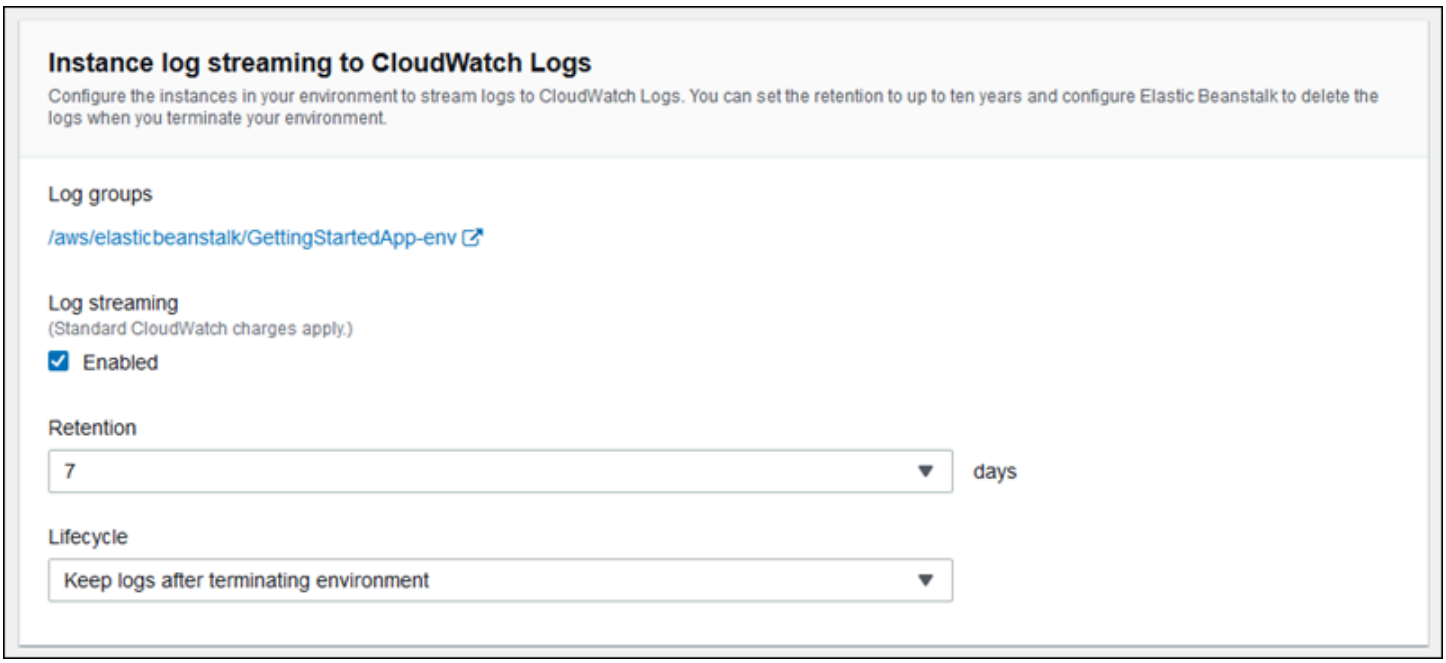
Par exemple, votre application écrit dans un fichier journal nommé `my_log.log`, et vous spécifiez ce nom dans votre fichier `.conf`. L'application effectue une rotation périodique du fichier. Pendant le cycle de rotation d'Elastic Beanstalk, les fichiers suivants se trouvent dans le dossier des fichiers journaux : `my_log.log`, `my_log.0800.log`, `my_log.0830.log`. Elastic Beanstalk considère tous ces fichiers comme des versions du nom de base `my_log`. Le fichier `my_log.log` comporte l'heure de modification la plus récente. Ainsi, Elastic Beanstalk charge uniquement les deux autres fichiers, `my_log.0800.log` et `my_log.0830.log`.

Streaming de fichiers journaux vers Amazon CloudWatch Logs

Vous pouvez configurer votre environnement pour diffuser les journaux vers Amazon CloudWatch Logs dans la console Elastic Beanstalk ou en utilisant les options de configuration. Avec CloudWatch Logs, chaque instance de votre environnement diffuse des journaux vers des groupes de journaux que vous pouvez configurer pour qu'ils soient conservés pendant des semaines, voire des années, même après la fermeture de votre environnement.

L'ensemble des journaux diffusés varie selon l'environnement, mais il inclut toujours `eb-engine.log` et les journaux d'accès provenant du serveur proxy nginx ou Apache qui s'exécute devant votre application.

Vous pouvez configurer la diffusion de journaux dans la console Elastic Beanstalk pendant la création de l'environnement ou pour un environnement existant. Vous pouvez définir les options suivantes depuis la console : activer/désactiver le streaming des CloudWatch journaux vers Logs, définir le nombre de jours de rétention et sélectionner l'une des options du cycle de vie. Dans l'exemple suivant, les journaux sont conservés jusqu'à sept jours, même lorsque l'environnement est résilié.



Le fichier de configuration suivant active la diffusion de journaux avec une conservation de 180 jours, même si l'environnement a été résilié.

Example.ebextensions/log-streaming.config

```
option_settings:
  aws:elasticbeanstalk:cloudwatch:logs:
    StreamLogs: true
    DeleteOnTerminate: false
    RetentionInDays: 180
```

Afficher les journaux de déploiement pour un environnement Elastic Beanstalk

Elastic Beanstalk génère un journal de déploiement pour chaque déploiement dans votre environnement. Le journal de déploiement fournit une vue chronologique consolidée de ce qui s'est passé au cours d'un déploiement, notamment de l'installation des dépendances, des résultats de compilation, du démarrage de l'application et des éventuelles erreurs rencontrées. Vous pouvez utiliser les journaux de déploiement pour diagnostiquer rapidement les échecs des déploiements sans avoir à vous connecter aux instances par SSH ou à corréler plusieurs fichiers journaux.

Les journaux de déploiement sont écrits localement sur chaque instance. Pour les déploiements déclenchés via la console, la CLI, l'API ou les mises à jour gérées, une instance télécharge en

permanence son journal sur Amazon S3 pendant le déploiement. La console Elastic Beanstalk lit le journal depuis Amazon S3, ce qui vous permet de suivre les progrès sans vous connecter à l'instance.

Les journaux de déploiement sont conçus pour être concis. En cas de succès, le journal affiche uniquement des messages récapitulatifs (par exemple, quelles commandes ont été exécutées et terminées). En cas d'échec, le journal inclut jusqu'à 50 lignes de sortie de l'étape ayant échoué, de sorte que vous pouvez voir l'erreur sans passer au crible les résultats détaillés.

> **Note**
>
> Les journaux de déploiement sont disponibles sur les versions des plateformes Amazon Linux 2 et Amazon Linux 2023 publiées le 11 mars 2026 ou après cette date. Les plateformes Windows ne sont actuellement pas prises en charge.

Opérations prises en charge

Les journaux de déploiement sont générés pour les opérations suivantes :

- Déploiements d'applications — Déploiement d'une nouvelle version d'application dans votre environnement.
- Mises à jour de configuration : modification des paramètres de configuration de l'environnement qui nécessitent des mises à jour d'instance.
- Création d'un environnement : déploiement initial lorsque vous créez un nouvel environnement.
- Redémarrer le serveur d'applications : redémarrez le serveur d'applications sur vos instances.

Les opérations qui ne modifient pas l'état de l'application ou de la configuration sur les instances, telles que la demande de journaux, l'échange CNAMEs ou la mise à jour de balises, ne génèrent pas de journaux de déploiement.

Contenu du journal de déploiement

Un journal de déploiement enregistre les informations suivantes lors d'un déploiement :

- Cycle de vie du déploiement : messages de début et de fin pour chaque phase de déploiement, tels que `Starting Application deployment` et`Completed Application deployment`.

- Sortie .ebextensions — En cas de succès, les noms des commandes exécutées. En cas d'échec, les 50 dernières lignes de `cfn-init` sortie pour aider à diagnostiquer le problème.
- Sortie des hooks de la plateforme : en cas de succès, les noms des scripts d'accroche exécutés. En cas d'échec, les 50 dernières lignes du hook sont sorties.
- Installation des dépendances : sortie des gestionnaires de packages tels que npm installpip install,composer install, etbundle install. En cas de succès, seul un message de fin est enregistré. En cas de panne, les 50 dernières lignes de sortie sont incluses.
- Sortie de construction : sortie à partir de commandes de génération telles que docker buildgo build, et les versions Java. En cas de panne, les 50 dernières lignes de sortie sont incluses.
- Sortie de démarrage de l'application : sortie initiale de votre application après son démarrage. La source dépend de votre plateforme :
  - Docker — Journaux de conteneurs provenant de ou docker logs docker compose logs
  - Java SE, Go, Node.js, Python, Ruby, .NET — Procédez aux journaux stdout
  - Tomcat — Catalina enregistre les résultats
  - PHP — Journaux d'erreurs du master et du pool PHP-FPM
  - ECS — Journaux de conteneur de chaque conteneur de tâches

> **Note**
>
> La sortie de l'application est capturée 2 secondes après le démarrage de l'application. Seuls les messages de démarrage initiaux sont inclus. Si votre application met plus de temps à produire une sortie, celle-ci n'apparaîtra pas dans le journal de déploiement. Pour consulter les journaux complets des applications, demandez des journaux de bundle ou connectez-vous directement à l'instance. Pour de plus amples informations, veuillez consulter Affichage des journaux d'instance.

Lorsqu'une étape de déploiement échoue, le journal la marque `[ERROR]` et inclut jusqu'à 50 lignes de sortie de l'étape échouée. Si le journal de déploiement ne contient pas suffisamment de détails, vous pouvez récupérer les journaux d'instance complets (y compris `eb-engine.log`eb-hooks.log, et les journaux d'application) à partir de l'onglet Logs. Pour de plus amples informations, veuillez consulter Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk.

Afficher les journaux de déploiement dans la console

La console Elastic Beanstalk propose un onglet Déploiements sur le tableau de bord de l'environnement où vous pouvez consulter l'historique et les journaux de vos déploiements.

Afficher l'historique des déploiements

Pour consulter l'historique des déploiements

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le tableau de bord de l'environnement, choisissez l'onglet Déploiements.

   L'onglet Déploiements affiche un tableau des déploiements pour l'environnement. Chaque ligne inclut les informations suivantes :

   - ID de demande : identifiant unique pour le déploiement.
   - État — Succès, échec ou en cours.
   - Type : type de déploiement, tel que création d'environnement, déploiement d'applications, mise à jour de configuration, mise à jour de plate-forme gérée, redémarrage du serveur d'applications, environnement de reconstruction, environnement de restauration, domaine d'environnement d'échange ou fin d'environnement.
   - Heure de début : date à laquelle le déploiement a commencé.
   - Durée : durée du déploiement.

Lorsqu'un déploiement est en cours, l'onglet interroge automatiquement les mises à jour. Vous pouvez également cliquer sur le bouton d'actualisation pour recharger manuellement la liste.

Afficher les détails et les journaux du déploiement

Pour consulter les détails du déploiement

1. Dans l'onglet Déploiements, choisissez le lien Request ID pour le déploiement que vous souhaitez inspecter.

2. La page détaillée du déploiement présente une section récapitulative avec l'ID de demande, le statut, le type de déploiement, l'heure de début, la durée et la politique de déploiement. La politique de déploiement (par exemple, Tout en une fois, Rolling, Rolling with additional batch, Immuable ou Traffic Splitting) est affichée lorsqu'elle peut être déterminée à partir des événements de déploiement.
3. Sous le résumé, choisissez l'un des onglets suivants :

   - Événements : chronologie des événements liés à ce déploiement, filtrée pour n'afficher que les événements relatifs au déploiement sélectionné.
   - Journaux de déploiement : journal de déploiement consolidé de l'instance. Vous pouvez effectuer une recherche, filtrer par niveau de journal et télécharger le fichier journal.

Pour les déploiements en cours, l'onglet journaux est automatiquement actualisé pour afficher les nouvelles entrées de journal au fur et à mesure qu'elles sont écrites. Une fois le déploiement terminé, la console récupère l'état final du journal pour s'assurer que vous voyez le résultat complet.

> **Important**
>
> L'affichage des journaux de déploiement dans la console nécessite une `s3:GetObject` autorisation sur le compartiment de stockage Amazon S3 de l'environnement (`elasticbeanstalk-`*`region`*`-`*`account-id`*). Si votre politique IAM n'inclut pas cette autorisation, l'historique et les événements du déploiement seront toujours disponibles, mais l'onglet journaux affichera une erreur.

Fichiers journaux de déploiement sur les instances

Les journaux de déploiement sont écrits `/var/log/deployments/` dans le répertoire de chaque instance. Le nom du fichier journal dépend de la manière dont le déploiement a été déclenché :

- Déploiements contrôlés par le flux de travail (déclenchés via la console, la CLI ou l'API) —`eb-deployment-`*`request-id`*`.log`, où *`request-id`* est l'ID unique de demande de déploiement.
- Déploiements à démarrage automatique (lancement d'instance ou redémarrage du serveur d'applications) —. `eb-deployment-`*`unix-timestamp`*`.log`

Elastic Beanstalk fait automatiquement pivoter ces fichiers, en conservant les 50 journaux de déploiement les plus récents sur chaque instance.

Pour les déploiements contrôlés par le flux de travail, le journal est chargé sur Amazon S3 via le chemin suivant :

```
s3://elasticbeanstalk-region-account-id/resources/environments/logs/
deployments/environment-id/log-filename
```

Dans les environnements multi-instances, la première instance à commencer le téléchargement revendique le rôle pour l'ensemble du déploiement. Cette instance télécharge son journal sur Amazon S3 pendant toute la durée du déploiement. Toutes les instances continuent d'écrire des journaux de déploiement localement.

> **Important**
>
> Le téléchargement des journaux de déploiement vers Amazon S3 nécessite une `s3:PutObject` autorisation sur le compartiment de stockage Amazon S3 de l'environnement dans le profil d'instance, et la configuration du VPC doit autoriser la connectivité à Amazon S3.

Les téléchargements du journal de déploiement sont plafonnés à 1 Mo par fichier. Si un journal de déploiement dépasse cette taille, la version téléchargée est tronquée avec un message indiquant que le journal complet est disponible sur l'instance.

Désactiver les téléchargements de journaux S3

Pour empêcher le chargement des journaux de déploiement sur Amazon S3, définissez la propriété d'environnement suivante sur votre environnement :

```
option_settings:
  - namespace:  aws:elasticbeanstalk:application:environment
    option_name:  EB_DEPLOYMENT_LOG_S3_DISABLED
    value:  true
```

Lorsque cette propriété d'environnement est définie, les journaux de déploiement sont toujours écrits localement `/var/log/deployments/` sur chaque instance, mais ils ne sont pas téléchargés sur Amazon S3 et ne seront pas disponibles dans l'onglet Déploiements de la console. Vous pouvez également définir cette propriété dans la page Configuration sous Logiciel, ou à l'aide de l'EB CLI ou AWS CLI.

Utilisation d'Elastic Beanstalk avec d'autres services AWS

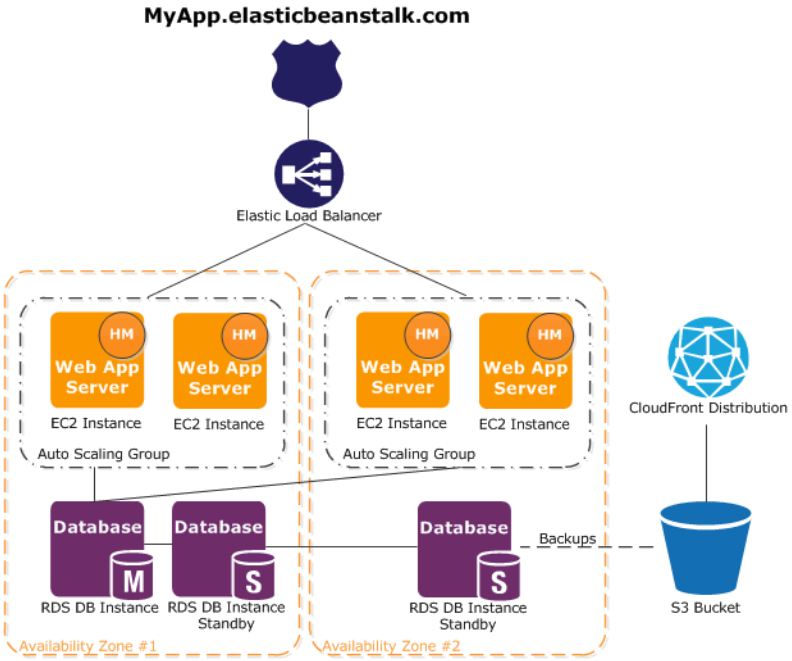
Les rubriques de cette section décrivent les nombreuses manières dont vous pouvez utiliser des AWS services supplémentaires avec votre application Elastic Beanstalk. Pour implémenter les environnements de votre application, Elastic Beanstalk gère les ressources AWS d'autres services ou utilise leurs fonctionnalités. En outre, Elastic Beanstalk AWS s'intègre à des services qu'il n'utilise pas directement dans le cadre de vos environnements.

Rubriques

- Présentation de l'architecture
- Utilisation d'Elastic Beanstalk avec Amazon CloudFront
- Journalisation des appels d'API Elastic Beanstalk avec AWS CloudTrail
- Utilisation d'Elastic Beanstalk avec Amazon CloudWatch
- Utilisation d'Elastic CloudWatch Beanstalk avec Amazon Logs
- Utilisation d'Elastic Beanstalk avec Amazon EventBridge
- Trouver et suivre les ressources Elastic Beanstalk avec AWS Config
- Utilisation d'Elastic Beanstalk avec Amazon DynamoDB
- Utilisation d'Elastic Beanstalk avec Amazon ElastiCache
- Utilisation d'Elastic Beanstalk avec Amazon Elastic File System
- Utilisation d'Elastic Beanstalk avec Gestion des identités et des accès AWS
- Utilisation d'Elastic Beanstalk avec Amazon RDS
- Utilisation d'Elastic Beanstalk avec Amazon S3
- Utilisation d'Elastic AWS Secrets Manager AWS Systems Manager Beanstalk avec et Parameter Store
- Utilisation d'Elastic Beanstalk avec Amazon VPC

Présentation de l'architecture

Le schéma suivant illustre un exemple d'architecture d'Elastic Beanstalk dans plusieurs zones de disponibilité fonctionnant avec AWS d'autres produits tels qu'Amazon, CloudFront Amazon Simple Storage Service (Amazon S3) et Amazon Relational Database Service (Amazon RDS).



Pour planifier la tolérance aux pannes, il est recommandé d'avoir N+1 instances Amazon EC2 et de répartir vos instances sur plusieurs zones de disponibilité. Dans le cas improbable où une seule zone de disponibilité s'arrête, vous aurez toujours vos autres instances Amazon EC2 en cours d'exécution dans une autre zone de disponibilité. Vous pouvez ajuster Amazon EC2 Auto Scaling afin d'autoriser un nombre minimal d'instances, ainsi que plusieurs zones de disponibilité. Pour obtenir des instructions sur la façon de procéder, veuillez consulter Auto Scaling de vos instances d'environnement Elastic Beanstalk. Pour plus d'informations sur la création d'applications tolérantes aux pannes, accédez à Création d'applications tolérantes aux pannes sur AWS.

Les sections suivantes traitent plus en détail de l'intégration avec Amazon CloudFront, Amazon CloudWatch, Amazon DynamoDB, Amazon, ElastiCache Amazon RDS, Amazon Route 53, Amazon Simple Storage Service, Amazon VPC et IAM.

Utilisation d'Elastic Beanstalk avec Amazon CloudFront

Amazon CloudFront est un service Web qui accélère la distribution de votre contenu Web statique et dynamique, par exemple des fichiers .html, .css, .php, des images et des fichiers multimédias, aux utilisateurs finaux. CloudFront diffuse votre contenu via un réseau mondial de sites périphériques. Lorsqu'un utilisateur final demande le contenu que vous diffusez CloudFront, il est dirigé vers l'emplacement périphérique offrant le moins de latence, afin que le contenu soit diffusé avec les meilleures performances possibles. Si ce contenu se trouve déjà dans l'emplacement périphérique, CloudFront le diffuse immédiatement. Si le contenu ne se trouve pas actuellement dans cet

emplacement périphérique, il est CloudFront extrait d'un compartiment Amazon S3 ou d'un serveur HTTP (par exemple, un serveur Web) que vous avez identifié comme source de la version définitive de votre contenu.

Après avoir créé et déployé votre application Elastic Beanstalk, vous pouvez vous CloudFront inscrire et CloudFront commencer à l'utiliser pour diffuser votre contenu. Pour en savoir plus, CloudFront consultez le guide du CloudFront développeur Amazon.

Journalisation des appels d'API Elastic Beanstalk avec AWS CloudTrail

Elastic Beanstalk est AWS CloudTrail intégré à un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans Elastic Beanstalk. CloudTrailcapture tous les appels d'API pour Elastic Beanstalk sous forme d'événements, y compris les appels provenant de la console Elastic Beanstalk, de l'EB CLI et de votre code vers Elastic Beanstalk. APIs Si vous créez un suivi, vous pouvez activer la diffusion continue d' CloudTrail événements vers un compartiment Amazon S3, y compris des événements pour Elastic Beanstalk. Si vous ne configurez pas de suivi, vous pouvez toujours consulter les événements les plus récents dans la CloudTrail console dans Historique des événements. À l'aide des informations collectées par CloudTrail, vous pouvez déterminer la demande qui a été envoyée à Elastic Beanstalk, l'adresse IP à partir de laquelle la demande a été faite, l'auteur de la demande, la date à laquelle elle a été faite, ainsi que des informations supplémentaires.

Pour en savoir plus CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.

Informations sur Elastic Beanstalk dans CloudTrail

CloudTrail est activé sur votre AWS compte lorsque vous le créez. Lorsqu'une activité se produit dans Elastic Beanstalk, elle est enregistrée CloudTrail dans un événement AWS avec d'autres événements de service dans l'historique des événements. Vous pouvez consulter, rechercher et télécharger les événements récents dans votre AWS compte. Pour plus d'informations, consultez la section Affichage des événements à l'aide de l'historique des CloudTrail événements.

Pour un enregistrement continu des événements de votre AWS compte, y compris des événements pour Elastic Beanstalk, créez un parcours. Un suivi permet CloudTrail de fournir des fichiers journaux à un compartiment Amazon S3. Par défaut, lorsque vous créez un journal de suivi dans la console, il s’applique à toutes les régions . Le journal d'activité consigne les événements de toutes les régions

dans la partition AWS et livre les fichiers journaux dans le compartiment Amazon S3 de votre choix. En outre, vous pouvez configurer d'autres AWS services pour analyser plus en détail les données d'événements collectées dans les CloudTrail journaux et agir en conséquence. Pour en savoir plus, consultez :

- Présentation de la création d'un journal d'activité
- CloudTrail Services et intégrations pris en charge
- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers CloudTrail journaux de plusieurs régions et réception de fichiers CloudTrail journaux de plusieurs comptes

Toutes les actions d'Elastic Beanstalk sont CloudTrail enregistrées et documentées dans le Guide de référence des API.AWS Elastic Beanstalk Par exemple, les appels aux `DescribeApplicationsUpdateEnvironment`, et `ListTagsForResource` les actions génèrent des entrées dans les fichiers CloudTrail journaux.

Chaque événement ou entrée de journal contient des informations sur la personne ayant initié la demande. Les informations relatives à l'identité permettent de déterminer les éléments suivants :

- Si la demande a été effectuée avec des informations d'identification d'utilisateur root ou IAM.
- Si la demande a été effectuée avec des informations d'identification de sécurité temporaires pour un rôle ou un utilisateur fédéré.
- Si la demande a été faite par un autre AWS service.

Pour plus d'informations, consultez la section Élément userIdentity CloudTrail .

Présentation des entrées du fichier journal Elastic Beanstalk

Un suivi est une configuration qui permet de transmettre des événements sous forme de fichiers journaux à un compartiment Amazon S3 que vous spécifiez. CloudTrail les fichiers journaux contiennent une ou plusieurs entrées de journal. Un événement représente une demande unique provenant de n'importe quelle source et inclut des informations sur l'action demandée, la date et l'heure de l'action, les paramètres de la demande, etc. CloudTrail les fichiers journaux ne constituent pas une trace ordonnée des appels d'API publics, ils n'apparaissent donc pas dans un ordre spécifique.

L'exemple suivant montre une entrée de CloudTrail journal qui illustre l'`UpdateEnvironment`action appelée par un utilisateur IAM nommé`intern`, pour l'`sample-env`environnement de l'`sample-app`application.


```
{
  "Records": [{
    "eventVersion": "1.05",
    "userIdentity": {
      "type": "IAMUser",
      "principalId": "AIXDAYQEXAMPLEUMLYNGL",
      "arn": "arn:aws:iam::123456789012:user/intern",
      "accountId": "123456789012",
      "accessKeyId": "ASXIAGXEXAMPLEQULKNXV",
      "userName": "intern",
      "sessionContext": {
        "attributes": {
          "mfaAuthenticated": "false",
          "creationDate": "2016-04-22T00:23:24Z"
        }
      },
      "invokedBy": "signin.amazonaws.com"
    },
    "eventTime": "2016-04-22T00:24:14Z",
    "eventSource": "elasticbeanstalk.amazonaws.com",
    "eventName": "UpdateEnvironment",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "255.255.255.54",
    "userAgent": "signin.amazonaws.com",
    "requestParameters": {
      "applicationName": "sample-app",
      "environmentName": "sample-env",
      "optionSettings": []
    },
    "responseElements": null,
    "requestID": "84ae9ecf-0280-17ce-8612-705c7b132321",
    "eventID": "e48b6a08-c6be-4a22-99e1-c53139cbfb18",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
  }]
}
```

Utilisation d'Elastic Beanstalk avec Amazon CloudWatch

Amazon vous CloudWatch permet de surveiller, de gérer et de publier diverses métriques, ainsi que de configurer des actions d'alarme en fonction des données issues des métriques. Amazon CloudWatch Monitoring vous permet de collecter, d'analyser et de consulter les indicateurs du système et des applications afin que vous puissiez prendre des décisions opérationnelles et commerciales plus rapidement et en toute confiance.

Vous pouvez utiliser Amazon CloudWatch pour collecter des statistiques concernant vos ressources Amazon Web Services (AWS), telles que les performances de vos instances Amazon. EC2
Vous pouvez également publier vos propres statistiques directement sur Amazon CloudWatch.
CloudWatch Les alarmes Amazon vous aident à mettre en œuvre des décisions plus facilement en vous permettant d'envoyer des notifications ou de modifier automatiquement les ressources que vous surveillez, en fonction de règles que vous définissez. Par exemple, vous pouvez créer des alarmes qui déclenchent des actions Amazon EC2 Auto Scaling et Amazon Simple Notification Service (Amazon SNS) en votre nom.

Elastic Beanstalk CloudWatch utilise automatiquement Amazon pour vous aider à surveiller l'état de votre application et de votre environnement. Vous pouvez accéder à la CloudWatch console Amazon pour consulter votre tableau de bord et avoir un aperçu de toutes vos ressources ainsi que de vos alarmes. Vous pouvez également afficher des métriques supplémentaires ou ajouter des métriques personnalisées.

Pour plus d'informations sur Amazon CloudWatch, consultez le manuel du CloudWatch développeur Amazon. Pour un exemple d'utilisation d'Amazon CloudWatch avec Elastic Beanstalk, consultez. the section called “Exemple : utilisation de CloudWatch métriques Amazon personnalisées”

Utilisation d'Elastic CloudWatch Beanstalk avec Amazon Logs

Cette rubrique explique les fonctionnalités de surveillance que le service Amazon CloudWatch Logs peut fournir à Elastic Beanstalk. Il vous guide également tout au long de la configuration et répertorie les emplacements des journaux pour chaque plateforme Elastic Beanstalk.

La mise en œuvre de CloudWatch journaux peut vous permettre d'effectuer les activités de surveillance suivantes :

- Surveillez et archivez votre application Elastic Beanstalk, votre système et vos fichiers journaux personnalisés à partir des instances Amazon EC2 de vos environnements.

- Configurez des alarmes qui vous permettent de réagir plus facilement à des événements spécifiques du flux de journal extraits par vos filtres métriques.

L'agent CloudWatch Logs installé sur chaque instance Amazon EC2 de votre environnement publie des points de données métriques sur le CloudWatch service pour chaque groupe de journaux que vous configurez. Chaque groupe de journaux applique ses propres modèles de filtre pour déterminer les événements du flux de journaux auxquels envoyer CloudWatch des points de données. Les flux de journaux qui appartiennent au même groupe de journaux partagent les mêmes paramètres de contrôle d'accès, de surveillance et de rétention. Vous pouvez configurer Elastic Beanstalk pour qu'il diffuse automatiquement les journaux CloudWatch vers le service, comme décrit dans. Streaming des logs d'instance vers CloudWatch Logs Pour plus d'informations sur CloudWatch les journaux, y compris la terminologie et les concepts, consultez le guide de l'utilisateur Amazon CloudWatch Logs.

Outre les journaux d'instance, si vous activez l'amélioration de l'état de santé de votre environnement, vous pouvez configurer l'environnement pour qu'il diffuse les informations de santé vers CloudWatch les journaux. Consultez Diffusion des informations relatives à l'état de santé de l'environnement Elastic Beanstalk vers Amazon Logs CloudWatch.

Rubriques

- Conditions préalables à la diffusion des journaux d'instance vers Logs CloudWatch
- Comment Elastic Beanstalk configure les journaux CloudWatch
- Streaming des logs d'instance vers CloudWatch Logs
- Résolution des problèmes et intégration CloudWatch des journaux
- Diffusion des informations relatives à l'état de santé de l'environnement Elastic Beanstalk vers Amazon Logs CloudWatch

Conditions préalables à la diffusion des journaux d'instance vers Logs CloudWatch

Pour activer le streaming des journaux depuis les instances Amazon EC2 de votre environnement vers CloudWatch Logs, vous devez remplir les conditions suivantes.

- Plateforme – Étant donné que cette fonctionnalité est disponible uniquement dans les versions de plateforme publiées en même temps que cette version ou après, si vous utilisez une version de plateforme antérieure, mettez à jour votre environnement vers la configuration actuelle.

- Si vous ne disposez pas de la politique gérée par AWSElasticBeanstalkWorkerTierElastic Beanstalk AWSElasticBeanstalkWebTierou d'Elastic Beanstalk dans votre profil d'instance Elastic Beanstalk, vous devez ajouter les éléments suivants à votre profil pour activer cette fonctionnalité.

  JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "logs:PutLogEvents",
      "logs:CreateLogStream"
    ],
    "Resource": [
    "*"
    ]
  }
  ]
}
```

Comment Elastic Beanstalk configure les journaux CloudWatch

Elastic Beanstalk CloudWatch installe un agent de journalisation avec les paramètres de configuration par défaut sur chaque instance créée. Pour en savoir plus, consultez le CloudWatch manuel Logs Agent Reference.

Lorsque vous activez le streaming des CloudWatch journaux d'instance vers Logs, Elastic Beanstalk envoie les fichiers journaux des instances de votre environnement vers Logs. CloudWatch Les journaux diffusés varient d'une plateforme à une autre. Le tableau suivant répertorie les différents journaux par plateforme.

Plateforme / Branche de plateforme	Journaux
Docker /	• /var/log/eb-engine.log • /var/log/eb-hooks.log • /var/log/docker

Plateforme / Branche de plateforme	Journaux
Branche de plateforme : Docker s'exécutant sur Amazon Linux 2 64 bits	• /var/log/docker-events.log • /var/log/eb-docker/containers/eb-current-app/stdouterr.journal • /var/log/nginx/access.journal • /var/log/nginx/error.journal
Docker / Branche de plateforme : ECS s'exécutant sur Amazon Linux 2 64 bits	• /var/log/docker-events.log • /var/log/eb-ecs-mgr.log • /var/log/eb-engine.log • /var/log/eb-hooks.log • /var/log/ecs/ecs-agent.log • /var/log/ecs/ecs-init.log
Go .NET Core sous Linux Java / Branche de plateform e : Corretto s'exécutant sur Amazon Linux 2 64 bits	• /var/log/eb-engine.log • /var/log/eb-hooks.log • /var/log/web.stdout.log • /var/log/nginx/access.journal • /var/log/nginx/error.journal
Node.js Python	• /var/log/eb-engine.log • /var/log/eb-hooks.log • /var/log/web.stdout.log • /var/log/httpd/access_journal • /var/log/httpd/error_journal • /var/log/nginx/access.journal • /var/log/nginx/error.journal

Plateforme / Branche de plateforme	Journaux
Tomcat PHP	• /var/log/eb-engine.log • /var/log/eb-hooks.log • /var/log/httpd/access_journal • /var/log/httpd/error_journal • /var/log/nginx/access.journal • /var/log/nginx/error.journal
.NET sous Windows Server	• C:\inetpub\logs \ \ W3 LogFiles SVC1 \ u_ex*.log • C:\Program Files \ Amazon \ ElasticBeanstalk \ logs \ AWSDeployment .log • C:\Program Files \ Amazon \ ElasticBeanstalk \ logs \ Hooks.log
Ruby	• /var/log/eb-engine.log • /var/log/eb-hooks.log • /var/log/puma/puma.journal • /var/log/web.stdout.log • /var/log/nginx/access.journal • /var/log/nginx/error.journal

Fichiers journaux sur les plateformes AMI Amazon Linux

> **Note**
>
> Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Pour plus d'informations sur la migration vers une branche de plateforme Amazon Linux 2023 actuelle et entièrement prise en charge, consultez Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2023 ou Amazon Linux 2.

Le tableau suivant répertorie les fichiers journaux diffusés à partir d'instances sur des branches de plateforme en fonction de l'AMI Amazon Linux (anciennement Amazon Linux 2), par plateforme.

Plateforme / Branche de plateforme	Journaux
Docker / Branche de plateforme : Docker s'exécutant sur Amazon Linux 64 bits	• /var/log/eb-activity.log • /var/log/nginx/error.journal • /var/log/docker-events.log • /var/log/docker • /var/log/nginx/access.journal • /var/log/eb-docker/containers/eb-current-app/stdouterr.journal
Docker / Branche de plateforme : Docker multiconteneur s'exécutant sur Amazon Linux 64 bits	• /var/log/eb-activity.log • /var/log/ecs/ecs-init.log • /var/log/eb-ecs-mgr.log • /var/log/ecs/ecs-agent.log • /var/log/docker-events.log
Glassfish (Docker préconfig uré)	• /var/log/eb-activity.log • /var/log/nginx/error.journal • /var/log/docker-events.log • /var/log/docker • /var/log/nginx/access.journal
Go	• /var/log/eb-activity.log • /var/log/nginx/error.journal • /var/log/nginx/access.journal
Java / Branche de plateforme : Java 8 s'exécutant sur Amazon Linux 64 bits	• /var/log/eb-activity.log • /var/log/nginx/access.journal • /var/log/nginx/error.journal • /var/log/web-1.error.log • /var/log/web-1.log

Plateforme / Branche de plateforme	Journaux
Branche de plateforme : Java 7 s'exécutant sur Amazon Linux 64 bits	
Tomcat	• /var/log/eb-activity.log • /var/log/httpd/error_journal • /var/log/httpd/access_journal • /var/log/nginx/error_journal • /var/log/nginx/access_journal
Node.js	• /var/log/eb-activity.log • /var/log/nodejs/nodejs.journal • /var/log/nginx/error.journal • /var/log/nginx/access.journal • /var/log/httpd/error.journal • /var/log/httpd/access.journal
PHP	• /var/log/eb-activity.log • /var/log/httpd/error_journal • /var/log/httpd/access_journal
Python	• /var/log/eb-activity.log • /var/log/httpd/error_journal • /var/log/httpd/access_journal • /opt/python/log/supervisord.journal
Ruby / Branche de plateforme : Puma s'exécutant sur Amazon Linux 64 bits	• /var/log/eb-activity.log • /var/log/nginx/error.journal • /var/log/puma/puma.journal • /var/log/nginx/access.journal

Plateforme / Branche de plateforme	Journaux
Ruby / Branche de plateforme : Passenger avec Ruby s'exécutant sur Amazon Linux 64 bits	• /var/log/eb-activity.log • /var/app/support/logs/passenger.journal • /var/app/support/logs/access.journal • /var/app/support/logs/error.journal

Elastic Beanstalk configure des CloudWatch groupes de journaux dans Logs pour les différents fichiers journaux qu'il diffuse. Pour récupérer des fichiers CloudWatch journaux spécifiques à partir de Logs, vous devez connaître le nom du groupe de journaux correspondant. Le schéma d'attribution de noms des groupes de journaux dépend du système d'exploitation utilisé par la plateforme.

Pour les plateformes Linux, préfixez l'emplacement du fichier journal de l'instance avec `/aws/elasticbeanstalk/`*`environment_name`* pour obtenir le nom du groupe de journaux. Par exemple, pour extraire le fichier `/var/log/nginx/error.log`, indiquez le nom du groupe de journaux `/aws/elasticbeanstalk/`*`environment_name`*`/var/log/nginx/error.log`.

Pour les plateformes Windows, consultez le tableau suivant pour connaître le groupe de journaux correspondant à chaque fichier journal.

Fichier journal d'instance	Groupe de journaux
`C:\Program Files\Amazon\ElasticBeanstalk\logs\AWSDeployment.log`	`/aws/elasticbeanstalk/<environment-name>/EBDeploy-Log`
`C:\Program Files\Amazon\ElasticBeanstalk\logs\Hooks.log`	`/aws/elasticbeanstalk/<environment-name>/EBHooks-Log`
`C:\inetpub\logs\LogFiles` (ensemble du répertoire)	`/aws/elasticbeanstalk/<environment-name>/IIS-Log`

Streaming des logs d'instance vers CloudWatch Logs

Vous pouvez activer le streaming des CloudWatch journaux d'instance vers Logs à l'aide de la console Elastic Beanstalk, de l'EB CLI ou des options de configuration.

Avant de l'activer, configurez les autorisations IAM à utiliser avec l'agent CloudWatch Logs. Vous pouvez attacher la politique personnalisée ci-après au profil d'instance que vous attribuez à votre environnement.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Streaming de journaux d'instance à l'aide de la console Elastic Beanstalk

Pour diffuser les journaux d'instance vers CloudWatch Logs

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.

4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Sous Transmission du journal d'instance vers CloudWatch Logs :
   - Activez Streaming des journaux.
   - Définissez Conservation sur le nombre de jours de conservation des journaux.
   - Sélectionnez le paramètre de cycle de vie qui détermine si les journaux sont enregistrés une fois l'environnement résilié.
6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Après avoir activé le streaming de journaux, vous pouvez revenir à la page ou à la catégorie de configuration Logiciels pour trouver le lien Groupes de journaux. Cliquez sur ce lien pour voir vos journaux dans la CloudWatch console.

Streaming de journaux d'instance à l'aide de l'interface de ligne de commande EB

Pour activer le streaming des CloudWatch journaux d'instance vers Logs à l'aide de l'interface de ligne de commande EB, utilisez la eb logs commande.

```
$ eb logs --cloudwatch-logs enable
```

Vous pouvez également l'utiliser eb logs pour récupérer des journaux à partir de CloudWatch Logs. Vous pouvez récupérer tous les journaux d'instance de l'environnement ou utiliser les nombreuses options de la commande pour spécifier des sous-ensembles de journaux à extraire. Par exemple, la commande suivante extrait l'ensemble complet des journaux d'instance de votre environnement et les enregistre dans un répertoire sous `.elasticbeanstalk/logs`.

```
$ eb logs --all
```

En particulier, l'option `--log-group` vous permet d'extraire les journaux d'instance d'un groupe de journaux spécifique, correspondant à un fichier journal d'instance spécifique. Pour ce faire, vous devez connaître le nom du groupe de journaux correspondant au fichier journal que vous souhaitez récupérer. Vous pouvez trouver ces informations dans Comment Elastic Beanstalk configure les journaux CloudWatch.

Streaming des journaux d'instance à l'aide de fichiers de configuration

Lorsque vous créez ou mettez à jour un environnement, vous pouvez utiliser un fichier de configuration pour configurer et configurer le streaming des CloudWatch journaux d'instance vers Logs. L'exemple de fichier de configuration suivant active le streaming des journaux d'instance par défaut. Elastic Beanstalk diffuse l'ensemble de fichiers journaux par défaut correspondant à la plateforme de votre environnement. Pour utiliser l'exemple, copiez le texte dans un fichier avec l'extension `.config` dans le répertoire `.ebextensions` au niveau supérieur du bundle de fichiers source de votre application.

```
option_settings:
  - namespace: aws:elasticbeanstalk:cloudwatch:logs
    option_name: StreamLogs
    value: true
```

Streaming de fichiers journaux personnalisés

L' CloudWatch intégration d'Elastic Beanstalk à Logs ne prend pas directement en charge le streaming de fichiers journaux personnalisés générés par votre application. Pour diffuser des journaux personnalisés, utilisez un fichier de configuration pour installer directement l' CloudWatch agent et pour configurer les fichiers à transférer. Pour obtenir un exemple de fichier de configuration, consultez `logs-streamtocloudwatch-linux.config`.

> **Note**
>
> L'exemple ne fonctionne pas sur la plateforme Windows.

Pour plus d'informations sur la configuration des CloudWatch journaux, consultez la référence du fichier de configuration de l'CloudWatch agent dans le guide de CloudWatch l'utilisateur Amazon.

Résolution des problèmes et intégration CloudWatch des journaux

> **Essayez Amazon Q Developer CLI pour un dépannage assisté par l'IA**
>
> L'interface de ligne de commande Amazon Q Developer peut vous aider à résoudre rapidement les problèmes d'environnement. La Q CLI fournit des solutions en vérifiant l'état de l'environnement, en examinant les événements, en analysant les journaux et en posant des questions de clarification. Pour plus d'informations et des instructions détaillées,

consultez la section Résolution des problèmes liés aux environnements Elastic Beanstalk avec Amazon Q Developer CLI dans les blogs. AWS

Impossible de localiser les journaux des instances d'environnement

Si vous ne trouvez pas certains des journaux d'instance de l'environnement auxquels vous vous attendiez dans CloudWatch Logs, étudiez les problèmes courants suivants :

- Votre rôle IAM ne dispose pas des autorisations IAM requises.
- Vous avez lancé votre environnement dans un environnement Région AWS qui ne prend pas en charge CloudWatch les journaux.
- L'un de vos fichiers journaux personnalisés n'existe pas dans le chemin que vous avez spécifié.

Journaux d'applications manquants ou intermittents

Si les journaux `/var/log/web.stdout.log` de votre application Elastic Beanstalk () semblent absents ou intermittents, cela peut être dû aux paramètres de limitation de débit par défaut dans rsyslog et journald. Bien que la désactivation complète de la limitation du débit puisse résoudre ce problème, elle n'est pas recommandée car elle pourrait entraîner une utilisation excessive du disque, un déni de service potentiel ou une dégradation des performances du système lors de rafales de log inattendues. Au lieu de cela, vous pouvez ajuster les limites de taux en utilisant ce qui suit `.ebextensions configuration`. Cette configuration augmente l'intervalle limite de débit à 600 secondes avec des limites de rafales plus élevées, offrant ainsi un équilibre entre une journalisation appropriée et la protection du système.

Problèmes de limitation

Si une opération Elastic Beanstalk qui lance simultanément un grand nombre d'instances renvoie un `Error: fail to create log stream: ThrottlingException: Rate exceeded` message tel que « cela est dû à un trop grand nombre d'appels à l'API ». CloudWatch

Pour résoudre le problème de limitation, effectuez l'une des actions suivantes :

- Utilisez une taille de lot plus petite avec des déploiements progressifs afin de réduire les mises à jour simultanées.
- Demandez une augmentation du quota de service limite de transactions par seconde (TPS) de votre AWS compte pour CreateLogStream. Pour plus d'informations, consultez les sections Quotas

de CloudWatch journaux et Gestion des quotas de votre service de CloudWatch journaux dans le guide de l'utilisateur Amazon CloudWatch Logs.

Diffusion des informations relatives à l'état de santé de l'environnement Elastic Beanstalk vers Amazon Logs CloudWatch

Si vous activez les rapports de santé améliorés pour votre environnement, vous pouvez configurer l'environnement pour qu'il diffuse les informations de santé dans les CloudWatch journaux. Ce streaming est indépendant du streaming du journal des EC2 instances Amazon. Cette rubrique décrit la diffusion des informations d'intégrité d'environnement. Pour plus d'informations sur la diffusion des journaux d'instance, consultez Utilisation d'Elastic CloudWatch Beanstalk avec Amazon Logs.

Lorsque vous configurez le streaming de l'état de l'environnement, Elastic Beanstalk CloudWatch crée un groupe de journaux pour l'état de l'environnement. Le nom de ce groupe de journaux est `/aws/elasticbeanstalk/`*`environment-name`*`/environment-health.log`. Dans ce groupe de journaux, Elastic Beanstalk crée des flux de journaux nommés *`YYYY-MM-DD`*`#`*`<hash-suffix>`* (il peut y avoir plusieurs flux de journaux par date).

Lorsque l'état de santé de l'environnement change, Elastic Beanstalk ajoute un enregistrement au flux de journaux d'état. L'enregistrement représente la transition d'état d'intégrité, c'est-à-dire le nouveau statut et une description de la cause du changement. Par exemple, un statut d'environnement peut passer à Grave parce que l'équilibreur de charge est en panne. Pour obtenir une description des statuts d'intégrité améliorée, consultez Couleurs et états utilisés dans les rapports d'intégrité.

Conditions préalables à la diffusion de l'état de l'environnement vers Logs CloudWatch

Pour activer le streaming de l'état de l'environnement vers CloudWatch Logs, vous devez remplir les conditions suivantes :

- Plateforme – Vous devez utiliser une version de plateforme qui prend en charge les rapports améliorés sur l'état.
- Autorisations – Vous devez accorder certaines autorisations de journalisation à Elastic Beanstalk afin qu'il puisse agir en votre nom pour diffuser les informations sur l'état de votre environnement. Si votre environnement n'utilise pas un rôle de service créé pour lui par Elastic Beanstalk, `aws-elasticbeanstalk-service-role` ou le rôle lié à un service de votre compte, `AWSServiceRoleForElasticBeanstalk`, veillez à ajouter les autorisations suivantes à votre rôle de service personnalisé.

```
{
      "Effect": "Allow",
      "Action": [
        "logs:DescribeLogStreams",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "arn:aws:logs:*:*:log-group:/aws/elasticbeanstalk/*:log-stream:*"
}
```

L'état de santé de l'environnement de streaming est enregistré dans CloudWatch des

Vous pouvez activer le streaming de l'état de l'environnement vers CloudWatch Logs à l'aide de la console Elastic Beanstalk, de l'EB CLI ou des options de configuration.

Diffusion des journaux d'état de l'environnement à l'aide de la console Elastic Beanstalk

Pour diffuser les journaux de santé de l'environnement vers CloudWatch Logs

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Monitoring (Surveillance), choisissez Edit (Modifier).
5. Sous Rapports sur l'état de santé, assurez-vous que l'option de rapports Système est définie sur Amélioré.
6. Diffusion des événements Under Health vers CloudWatch Logs
   - Activez Streaming des journaux.
   - Définissez Conservation sur le nombre de jours de conservation des journaux.
   - Sélectionnez le paramètre de cycle de vie qui détermine si les journaux sont enregistrés une fois l'environnement résilié.
7. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Après avoir activé le streaming de journaux, vous pouvez revenir à la page ou à la catégorie de configuration Surveillance pour trouver le lien Groupe de journaux. Cliquez sur ce lien pour consulter les journaux de santé de votre environnement dans la CloudWatch console.

Diffusion des journaux d'intégrité d'environnement à l'aide de l'interface de ligne de commande EB

Pour activer le streaming des journaux de santé de l'environnement vers CloudWatch les journaux à l'aide de l'interface de ligne de commande EB, utilisez la eb logs commande.

```
$ eb logs --cloudwatch-logs enable --cloudwatch-log-source environment-health
```

Vous pouvez également l'utiliser eb logs pour récupérer des journaux à partir de CloudWatch Logs. Par exemple, la commande suivante extrait l'ensemble des journaux d'intégrité de votre environnement et les enregistre dans un répertoire sous `.elasticbeanstalk/logs`.

```
$ eb logs --all --cloudwatch-log-source environment-health
```

Diffusion des journaux d'intégrité d'environnement à l'aide de fichiers de configuration

Lorsque vous créez ou mettez à jour un environnement, vous pouvez utiliser un fichier de configuration pour configurer et configurer le streaming des journaux de santé de l'environnement vers CloudWatch Logs. Pour utiliser l'exemple ci-dessous, copiez le texte dans un fichier avec l'extension `.config` dans le répertoire `.ebextensions` au niveau supérieur du bundle de fichiers source de votre application. Cet exemple configure Elastic Beanstalk pour activer la diffusion des journaux d'état de l'environnement, conserver les journaux après l'arrêt de l'environnement et les enregistrer pendant 30 jours.

Example Fichier de configuration de diffusion des journaux sur l'état de santé

```
#####################################################################
##  Sets up Elastic Beanstalk to stream environment health information
##  to Amazon CloudWatch Logs.
##  Works only for environments that have enhanced health reporting enabled.
#####################################################################

option_settings:
  aws:elasticbeanstalk:cloudwatch:logs:health:
    HealthStreamingEnabled: true
    ### Settings below this line are optional.
```

```
    # DeleteOnTerminate: Delete the log group when the environment is
    # terminated. Default is false. If false, the health data is kept
    # RetentionInDays days.
    DeleteOnTerminate: false
    # RetentionInDays: The number of days to keep the archived health data
    # before it expires, if DeleteOnTerminate isn't set. Default is 7 days.
    RetentionInDays: 30
```

Pour obtenir les valeurs par défaut et les valeurs valides des options, consultez `aws:elasticbeanstalk:cloudwatch:logs:health`.

Utilisation d'Elastic Beanstalk avec Amazon EventBridge

À l'aide d'Amazon EventBridge, vous pouvez configurer des règles basées sur les événements qui surveillent vos ressources Elastic Beanstalk et initient des actions ciblées utilisant d'autres services. AWS Par exemple, vous pouvez définir une règle pour l'envoi de notifications par e-mail en signalant une rubrique Amazon SNS chaque fois que l'état de santé d'un environnement de production passe à Warning (Avertissement). Vous pouvez également définir une fonction Lambda pour transmettre une notification à Slack chaque fois que l'état de santé de votre environnement passe à Degraded (Dégradé) ou à Severe (Grave).

Vous pouvez créer des règles dans Amazon EventBridge pour agir sur n'importe lequel des événements Elastic Beanstalk suivants :

- Changements d'état des opérations d'environnement (y compris les opérations de création, de mise à jour et de résiliation). L'événement indique si le changement d'état a démarré, réussi ou échoué.
- Changements d'état pour d'autres ressources.Outre les environnements, les autres ressources surveillées incluent les équilibreurs de charge, les groupes Auto Scaling et les instances.
- Transition d'état pour les environnements. L'événement indique l'endroit où l'état de l'environnement est passé d'un état de santé à un autre.
- Changement d'état pour les mises à jour gérées. L'événement indique si le changement d'état a démarré, réussi ou échoué.

Pour capturer des événements Elastic Beanstalk spécifiques qui vous intéressent, définissez des EventBridge modèles spécifiques à ces événements qui peuvent être utilisés pour détecter les événements. Les modèles d'événements ont la même structure que les événements auxquels ils

correspondent. Le modèle place entre guillemets les champs que vous voulez faire correspondre et fournit les valeurs que vous recherchez. Les événements sont générés dans la mesure du possible. Ils sont transmis depuis Elastic EventBridge Beanstalk en temps quasi réel dans des circonstances opérationnelles normales. Toutefois, certaines situations peuvent retarder ou empêcher la livraison d'un événement.

Pour obtenir la liste des champs contenus dans les événements Elastic Beanstalk et leurs valeurs de chaîne possibles, veuillez consulter Mappage des champs d'événement Elastic Beanstalk. Pour plus d'informations sur le fonctionnement EventBridge des règles avec les modèles d'événements, consultez la section Événements et modèles d'événements dans EventBridge.

Surveillez une ressource Elastic Beanstalk avec EventBridge

Avec EventBridge, vous pouvez créer des règles qui définissent les actions à entreprendre lorsqu'Elastic Beanstalk émet des événements pour ses ressources. Par exemple, vous pouvez créer une règle qui vous envoie un e-mail chaque fois que l'état d'un environnement change.

La EventBridge console dispose d'une option de modèle prédéfinie pour créer des modèles d'événements Elastic Beanstalk. Si vous sélectionnez cette option dans la EventBridge console lorsque vous créez une règle, vous pouvez créer rapidement un modèle d'événement Elastic Beanstalk. Il suffit de sélectionner les champs et les valeurs de l'événement. Au fur et à mesure que vous effectuez des sélections, la console crée et affiche le modèle d'événement. Vous pouvez également modifier manuellement le modèle d'événement que vous créez et pouvez l'enregistrer en tant que modèle personnalisé. La console vous fournit également l'option d'afficher un Sample Event (Exemple d'événement) détaillé que vous pouvez copier et coller dans le modèle d'événement que vous créez.

Si vous préférez taper ou copier-coller un modèle d'événement dans la EventBridge console, vous pouvez choisir d'utiliser l'option Modèle personnalisé dans la console. Ce faisant, vous n'avez pas besoin de suivre les étapes de sélection des champs et des valeurs décrites précédemment. Cette rubrique propose des exemples de modèles de correspondance d'événements et d'événements Elastic Beanstalk que vous pouvez utiliser.

Pour créer une règle pour un événement de ressource

1. Connectez-vous à AWS l'aide d'un compte autorisé à utiliser Elastic EventBridge Beanstalk.
2. Ouvrez la EventBridge console Amazon à l'adresse https://console.aws.amazon.com/events/.
3. Dans le panneau de navigation, choisissez Rules.

4. Choisissez Créer une règle.
5. Entrez un nom et éventuellement une description pour la règle.
6. Pour Event bus (Bus d'événement), choisissez default (défaut). Lorsqu'un AWS service de votre compte émet un événement, celui-ci est toujours redirigé vers le bus d'événements par défaut de votre compte.
7. Pour Type de règle, choisissez Règle avec un modèle d'événement.
8. Choisissez Suivant.
9. Dans Source de l'événement, choisissez AWS des événements ou des événements EventBridge partenaires.
10. (Facultatif) Dans Exemple d'événement, sélectionnez AWS événements. Saisissez Elastic Beanstalk dans le champ de recherche. Cela fournit une liste d'exemples d'événements Elastic Beanstalk que vous pouvez choisir d'afficher. Cette étape affiche simplement un exemple d'événement que vous pouvez référencer. Cela n'affecte pas le résultat de la création de la règle. La section Exemple d'événement Elastic Beanstalk plus loin dans cette rubrique vous permet d'avoir des exemples du même type d'événement.
11. Dans la section Event pattern (Modèle d'événement), choisissez Event pattern form (Modèle d'événement).

    > **Note**
    >
    > Si vous avez déjà du texte pour un modèle d'événement et que vous n'avez pas besoin de la EventBridge console pour le créer, sélectionnez Modèle personnalisé (éditeur JSON). Vous pouvez ensuite saisir ou copier et coller manuellement du texte dans la zone Event Pattern (Modèle d'événement). Choisissez Next (Suivant), puis passez à l'étape relative à la saisie d'une cible.

12. Pour Source d'événement, choisissez Services AWS .
13. Pour le AWS service, sélectionnez Elastic Beanstalk.
14. Pour Event type (Type d'événement), sélectionnez Statut Change (Changement de statut).
15. Cette étape explique comment utiliser les champs d'événement detail type (type de détail), statut (état) et severity (gravité) pour Elastic Beanstalk. Au fur et à mesure que vous sélectionnez ces champs et les valeurs de votre choix, la console crée et affiche le modèle d'événement.

    - Si vous sélectionnez une seule valeur pour le ou les types de détails spécifiques, vous pouvez choisir une ou plusieurs valeurs pour le champ suivant de la hiérarchie.

- Si vous choisissez plusieurs valeurs pour des types de détails spécifiques, ne choisissez pas de valeurs spécifiques pour les champs suivants de la hiérarchie. Cela permet d'avoir une logique de correspondance ambiguë entre les champs de votre modèle d'événement.

Le champ d'événement environment (environnement) n'est pas affecté par cette hiérarchie, il s'affiche comme décrit à l'étape suivante.

16. Pour l'environment, sélectionnez Any environment (N'importe quel environnement) ou Specific environment(s) (Environnement(s) spécifique(s)).

    - Si vous sélectionnez Environnement (s) spécifique (s), vous pouvez choisir un ou plusieurs environnements dans la liste déroulante. EventBridge ajoute tous les environnements que vous sélectionnez dans la liste EnvironmentName[] de la section détaillée du modèle d'événement. Ensuite, votre règle filtre tous les événements pour inclure uniquement les environnements spécifiques que vous sélectionnez.
    - Si vous sélectionnez Any environment (N'importe quel environnements), aucun environnement n'est ajouté à votre modèle d'événement. Pour cette raison, votre règle ne filtre aucun des événements Elastic Beanstalk en se basant sur l'environnement.

17. Choisissez Suivant.
18. Pour Types de cibles, choisissez service AWS .
19. Pour Select a target (Sélectionner une cible), choisissez l'action cible à effectuer lorsqu'un événement de changement d'état de ressource est reçu de la part d'Elastic Beanstalk.

    Par exemple, vous pouvez utiliser une rubrique Amazon Simple Notification Service (SNS) pour envoyer un e-mail ou un SMS lorsqu'un événement se produit. Pour ce faire, vous devez créer une rubrique Amazon SNS à l'aide de la console Amazon SNS. Pour en savoir plus, veuillez consulter Utilisation d'Amazon SNS pour les notifications utilisateur.

    > **Important**
    >
    > Certaines actions cibles peuvent nécessiter l'utilisation d'autres services entraînant des frais supplémentaires, tels que le service Amazon SNS ou Lambda. Pour plus d'informations sur la AWS tarification, consultezhttps://aws.amazon.com/pricing/. Certains services font partie du niveau d'utilisation AWS gratuit. Si vous êtes un nouveau client, vous pouvez essayer ces services gratuitement. Pour plus d’informations, consultez https://aws.amazon.com/free/.

20. (Facultatif) Sélectionnez Add another target (Ajouter une autre cible) pour spécifier une action cible supplémentaire pour la règle d'événement.
21. Choisissez Suivant.
22. (Facultatif) Saisissez une ou plusieurs balises pour la règle. Pour plus d'informations, consultez les EventBridge balises Amazon dans le guide de EventBridge l'utilisateur Amazon.
23. Choisissez Suivant.
24. Consultez les détails de la règle et choisissez Create rule (Créer une règle).

Exemple de modèles d'événements Elastic Beanstalk

Les modèles d'événements ont la même structure que les événements auxquels ils correspondent. Le modèle place entre guillemets les champs que vous voulez faire correspondre et fournit les valeurs que vous recherchez.

- Health status change (Changement de l'état de santé) pour tous les environnements

```
{
   "source": [
    "aws.elasticbeanstalk"
  ],
  "detail-type": [
    "Health status change"
    ]
}
```

- Health status change (Changement de l'état de santé) pour les environnements suivants : `myEnvironment1` et `myEnvironment2`. Ce modèle d'événement filtre pour ces deux environnements spécifiques, tandis que l'exemple précédent de Health status change (Changement de l'état de santé) qui ne filtre pas envoie des événements pour tous les environnements.

```
{"source": [
    "aws.elasticbeanstalk"
    ],
    "detail-type": [
        "Health status change"
    ],
    "detail": {
        "EnvironmentName": [
```

```
            "myEnvironment1",
            "myEnvironment2"
        ]
    }
}
```

- Elastic Beanstalk resource status change (Changement de l'état de santé des ressources Elastic Beanstalk) pour tous les environnements

```
{
  "source": [
    "aws.elasticbeanstalk"
  ],
  "detail-type": [
    "Elastic Beanstalk resource status change"
    ]
}
```

- Elastic Beanstalk resource status change (Changement de l'état de ressource Elastic Beanstalk) avec `Status` Environment update failed (Échec de la mise à jour d'environnement) et `Severity` ERROR (ERREUR) pour les environnements suivants : `myEnvironment1` et `myEnvironment2`

```
{"source": [
    "aws.elasticbeanstalk"
    ],
    "detail-type": [
        "Elastic Beanstalk resource status change"
    ],
    "detail": {
        "Status": [
            "Environment update failed"
            ],
        "Severity": [
            "ERROR"
            ],
        "EnvironmentName": [
            "myEnvironment1",
            "myEnvironment2"
        ]
    }
}
```

- Other resource status change (Autre changement d'état des ressources) pour les équilibreurs de charge, les groupes Auto Scaling et les instances

```
{
   "source": [
    "aws.elasticbeanstalk"
  ],
  "detail-type": [
    "Other resource status change"
    ]
}
```

- Managed update status change (Changement d'état des mises à jour gérées) pour tous les environnements

```
{
   "source": [
    "aws.elasticbeanstalk"
  ],
  "detail-type": [
    "Managed update status change"
    ]
}
```

- Pour capturer tous les événements à partir d'Elastic Beanstalk, excluez la section `detail-type` :

```
{
  "source": [
    "aws.elasticbeanstalk"
  ]
}
```

Exemple d'événement Elastic Beanstalk

Voici un exemple d'événement Elastic Beanstalk pour un changement d'état de ressource :

```
{
   "version":"0",
   "id":"1234a678-1b23-c123-12fd3f456e78",
   "detail-type":"Elastic Beanstalk resource status change",
   "source":"aws.elasticbeanstalk",
```

```
   "account":"111122223333",
   "time":"2020-11-03T00:31:54Z",
   "region":"us-east-1",
   "resources":[
      "arn:was:elasticbeanstalk:us-east-1:111122223333:environment/myApplication/
myEnvironment"
   ],
   "detail":{
      "Status":"Environment creation started",
      "EventDate":1604363513951,
      "ApplicationName":"myApplication",
      "Message":"createEnvironment is starting.",
      "EnvironmentName":"myEnvironment",
      "Severity":"INFO"
   }
}
```

Voici un exemple d'événement Elastic Beanstalk pour un changement d'état de santé :

```
{
   "version":"0",
   "id":"1234a678-1b23-c123-12fd3f456e78",
   "detail-type":"Health status change",
   "source":"aws.elasticbeanstalk",
   "account":"111122223333",
   "time":"2020-11-03T00:34:48Z",
   "region":"us-east-1",
   "resources":[
      "arn:was:elasticbeanstalk:us-east-1:111122223333:environment/myApplication/
myEnvironment"
   ],
   "detail":{
      "Status":"Environment health changed",
      "EventDate":1604363687870,
      "ApplicationName":"myApplication",
      "Message":"Environment health has transitioned from Pending to Ok. Initialization
 completed 1 second ago and took 2 minutes.",
      "EnvironmentName":"myEnvironment",
      "Severity":"INFO"
   }
}
```

Mappage des champs d'événement Elastic Beanstalk

Le tableau suivant associe les champs d'événements Elastic Beanstalk et leurs valeurs de chaîne possibles au champ. EventBridge `detail-type` Pour plus d'informations sur le EventBridge fonctionnement des modèles d'événements pour un service, consultez la section Événements et modèles d'événements dans EventBridge.

EventBridge type de détail du champ	Champ Elastic Beanstalk Status	Champ Elastic Beanstalk Severity	Champ Elastic Beanstalk Message
Elastic Beanstalk resource status change	Environment creation started	INFO	createEnvironment is starting.
	Environment creation successful	INFO	createEnvironment completed successfully.
	Environment creation successful	INFO	Launched environment: <Environment Name>. However, there were issues during launch. See event log for details.
	Environment creation failed	ERROR	Failed to launch environment.
	Environment update started	INFO	Environment update is starting.
	Environment update successful	INFO	Environment update completed successfully.

EventBridge type de détail du champ	Champ Elastic Beanstalk Status	Champ Elastic Beanstalk Severity	Champ Elastic Beanstalk Message
	Environment update failed	ERROR	Failed to deploy configuration.
	Environment termination started	INFO	terminateEnvironment is starting.
	Environment termination successful	INFO	terminateEnvironment completed successfully.
	Environment termination failed	INFO	The environment termination step failed because at least one of the environment termination workflows failed.
Other resource status change	Auto Scaling group created	INFO	createEnvironment is starting.
	Auto Scaling group deleted	INFO	createEnvironment is starting.
	Instance added	INFO	Added instance [i-123456789a12b1234] to your environment.
	Instance removed	INFO	Removed instance [i-123456789a12b1234] from your environment.

EventBridge type de détail du champ	Champ Elastic Beanstalk Status	Champ Elastic Beanstalk Severity	Champ Elastic Beanstalk Message
	Load balancer created	INFO	Created load balancer named: <LB Name>
	Load balancer deleted	INFO	Deleted load balancer named: <LB Name>
Health status change	Environment health changed	INFO/ WARN	Environment health has transitioned to <healthStatus>.
	Environment health changed	INFO/ WARN	Environment health has transitioned from <healthStatus> to <healthStatus>.
Managed update status change	Managed updated started	INFO	Managed platform update is in-progress.
	Managed update failed	INFO	Managed update failed, retrying in %s minutes.

Trouver et suivre les ressources Elastic Beanstalk avec AWS Config

AWS Configfournit une vue détaillée de la configuration des AWS ressources de votre AWS compte. Vous pouvez voir comment les ressources sont liées, obtenir un historique des changements de configuration, et voir comment les configurations et les relations changent au fil du temps. Vous pouvez l'utiliser AWS Config pour définir des règles qui évaluent les configurations des ressources en termes de conformité des données.

Plusieurs types de ressources Elastic Beanstalk sont intégrés à : AWS Config

- Applications
- Versions de l'application
- Environnements

La section suivante explique comment configurer AWS Config pour enregistrer des ressources de ce type.

Pour plus d'informations à ce sujet AWS Config, consultez le guide du AWS Config développeur. Pour obtenir des informations sur la tarification, consultez la page des informations de tarification d'AWS Config.

Con AWS Config figuration

Pour la configuration initiale AWS Config, consultez les rubriques suivantes du guide du AWS Config développeur.

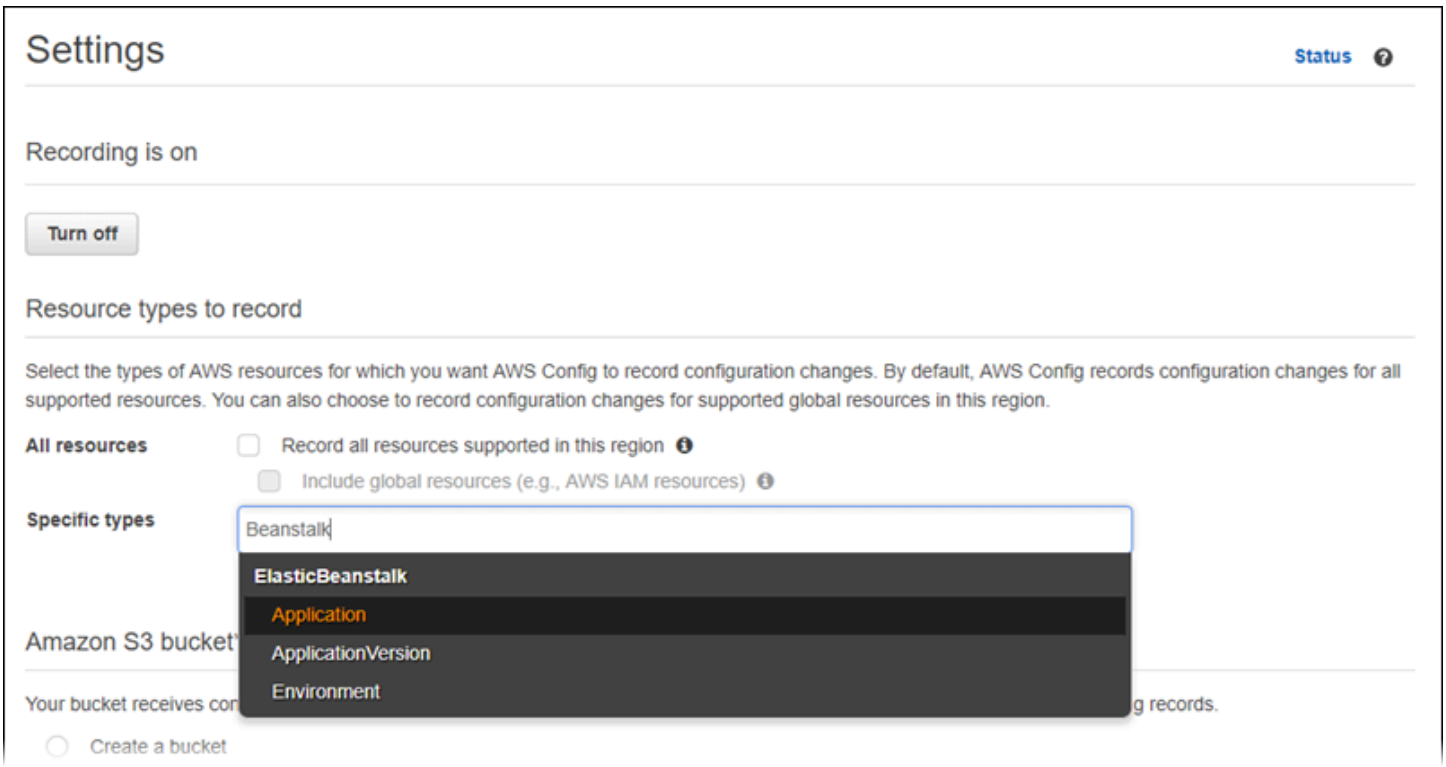
- Configuration à l' AWS Config aide de la console
- Configuration à l' AWS Config aide du AWS CLI

Configuration AWS Config pour enregistrer les ressources Elastic Beanstalk

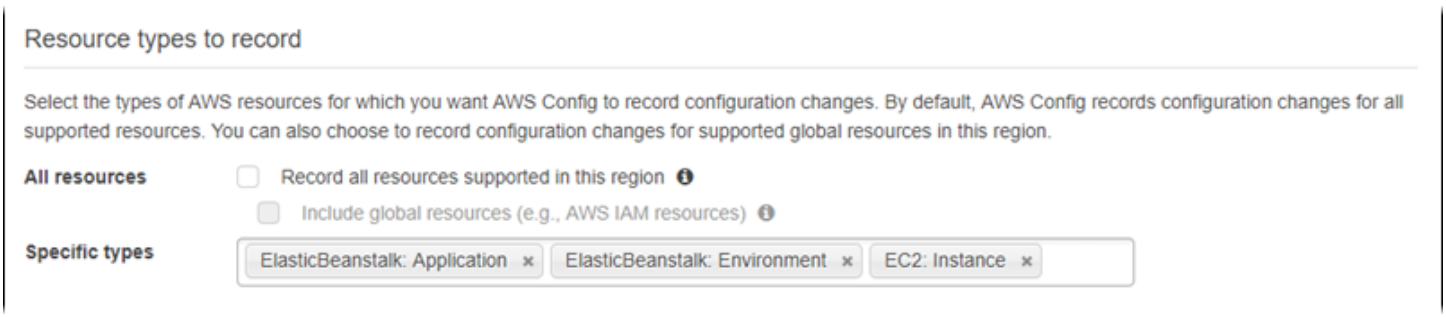
Par défaut, AWS Config enregistre les modifications de configuration pour tous les types de ressources régionales pris en charge qu'il découvre dans la région dans laquelle votre environnement s'exécute. Vous pouvez le personnaliser AWS Config pour enregistrer uniquement les modifications relatives à des types de ressources spécifiques ou les modifications apportées aux ressources globales.

Par exemple, vous pouvez configurer pour enregistrer les modifications apportées AWS Config aux ressources Elastic Beanstalk et à un sous-ensemble d'autres AWS ressources créées par Elastic Beanstalk pour vous. À l'aide de la AWS Config console, vous pouvez sélectionner Elastic Beanstalk comme ressource AWS Config dans la page Paramètres dans le champ Types spécifiques. À partir de là, vous pouvez choisir d'enregistrer n'importe quel type de ressource Elastic Beanstalk : ApplicationVersionapplication et environnement.

La figure suivante montre la page AWS Config Paramètres, avec les types de ressources Elastic Beanstalk que vous pouvez choisir d'enregistrer : ApplicationVersionApplication et Environnement.



Une fois que vous avez sélectionné quelques types de ressources, voici à quoi ressemble la liste Specific types (Types spécifiques).



Pour en savoir plus sur les ressources régionales par rapport aux ressources mondiales, et pour connaître la procédure de personnalisation complète, consultez la section Sélection des ressources AWS Config enregistrées.

Afficher les détails de configuration d'Elastic Beanstalk dans la console AWS Config

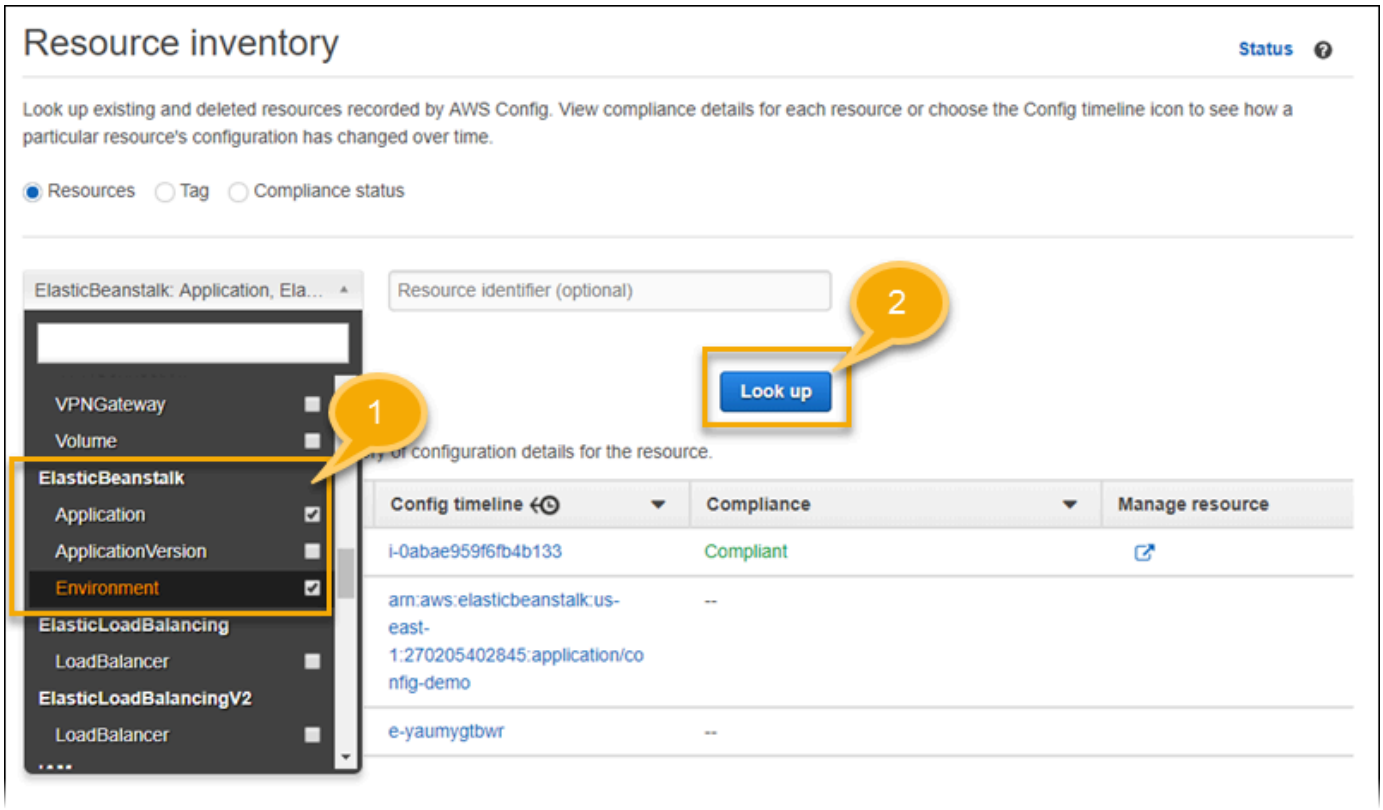
Vous pouvez utiliser la AWS Config console pour rechercher des ressources Elastic Beanstalk et obtenir des informations actuelles et historiques sur leurs configurations. L'exemple suivant montre comment rechercher des informations sur un environnement Elastic Beanstalk.

Pour rechercher un environnement Elastic Beanstalk dans la console AWS Config

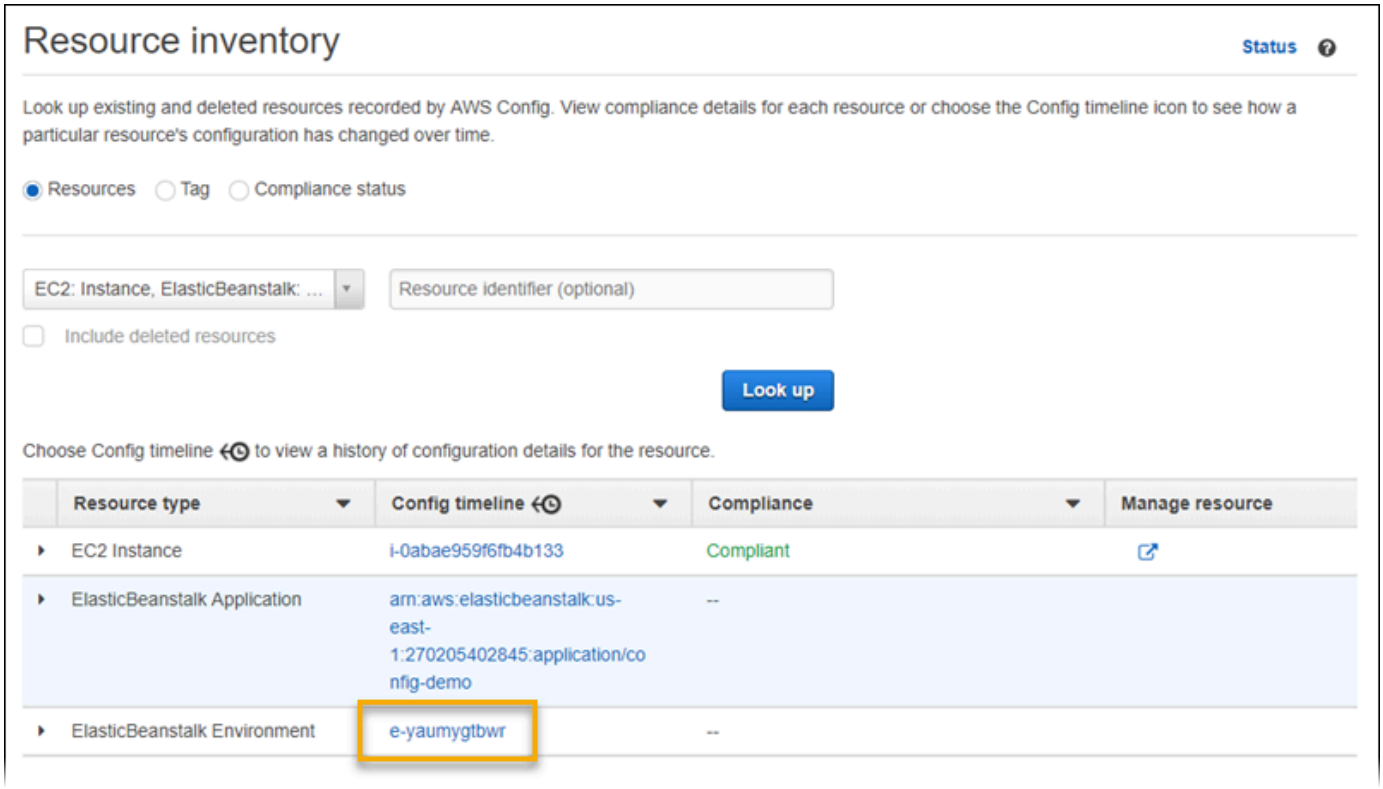
1. Ouvrez la console AWS Config.
2. Choisissez Ressources.
3. Sur la page Resource Inventory (Inventaire des ressources), choisissez Resources (Ressources).
4. Ouvrez le menu Type de ressource, faites défiler l'écran jusqu'à ElasticBeanstalk, puis choisissez un ou plusieurs types de ressources Elastic Beanstalk.

> **Note**
>
> Pour afficher les détails de configuration d'autres ressources créées par Elastic Beanstalk pour votre application, choisissez des types de ressources supplémentaires. Par exemple, vous pouvez sélectionner Instance sous EC2.

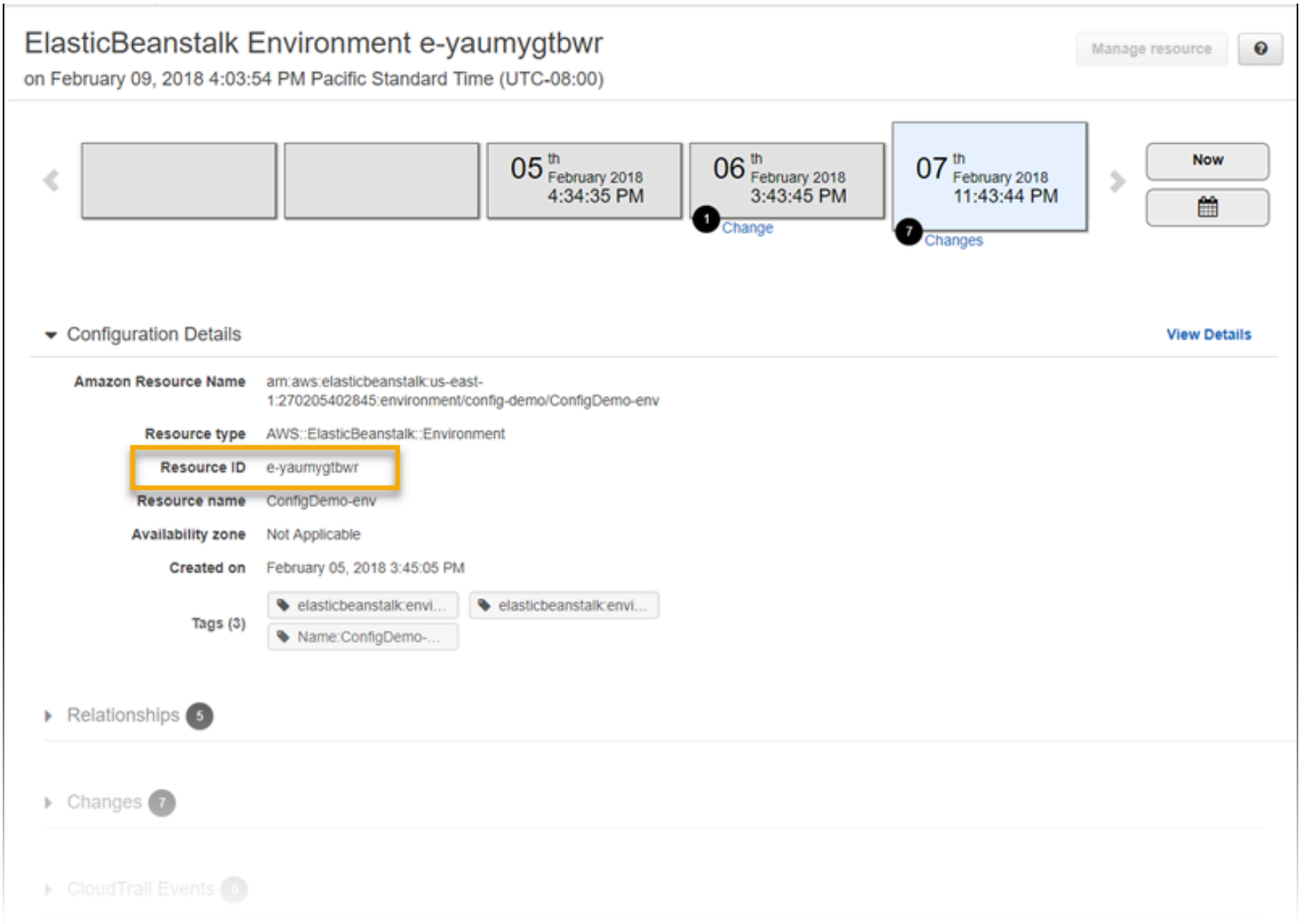
5. Choisissez Recherche. Voir 2 dans la figure suivante.



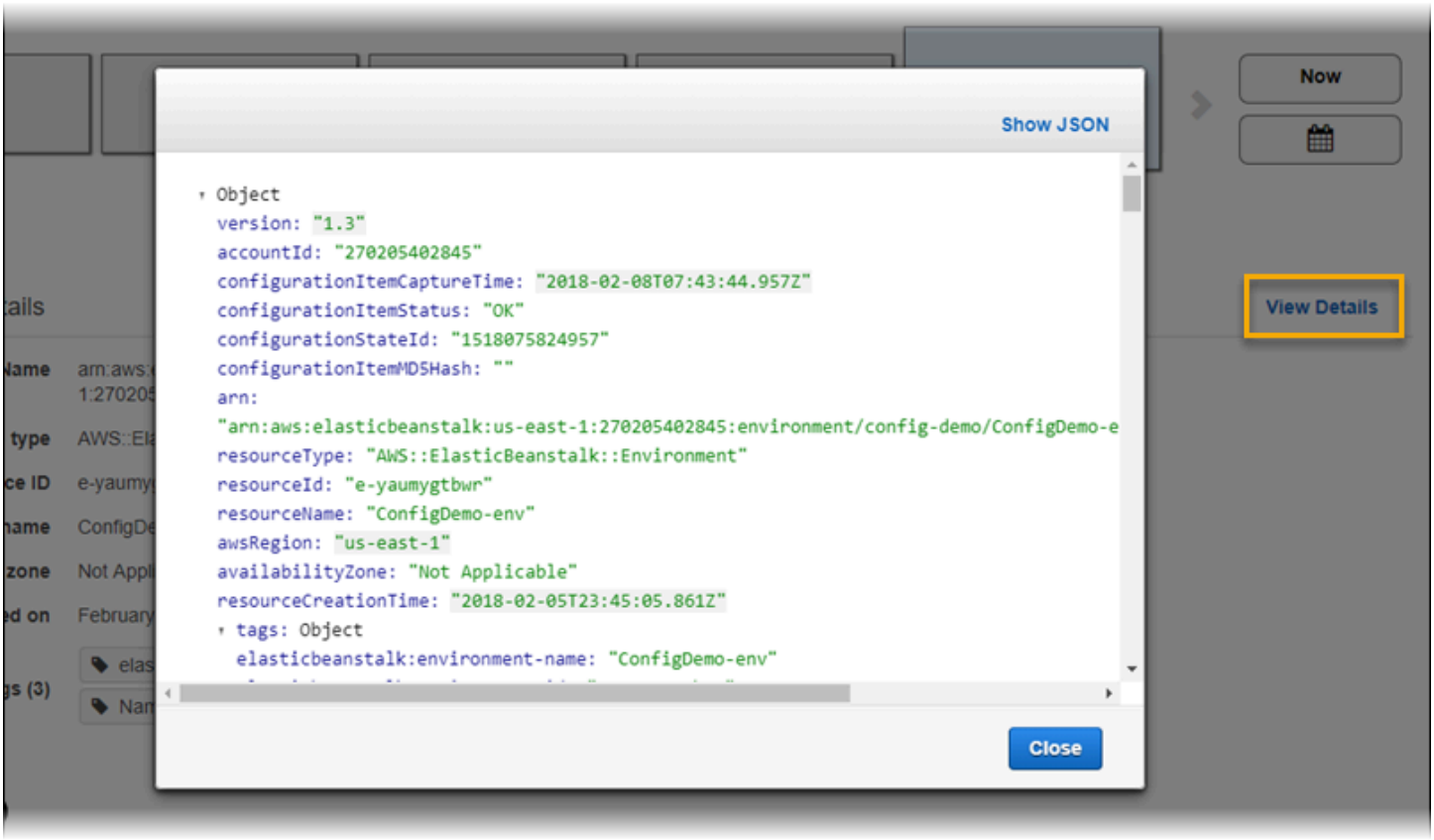
6. Choisissez un ID de ressource dans la liste des ressources qui AWS Config s'affiche.



AWS Config affiche les détails de configuration et d'autres informations sur la ressource que vous avez sélectionnée.



Pour voir tous les détails de la configuration enregistrée, choisissez View Details (Afficher les détails).



Pour en savoir plus sur les moyens de rechercher une ressource et d'afficher les informations sur cette page, consultez la section Affichage AWS des configurations et de l'historique des ressources dans le guide du AWS Config développeur.

Évaluation des ressources Elastic Beanstalk à l'aide de AWS Config règles

Vous pouvez créer des AWS Config règles qui représentent les paramètres de configuration idéaux pour vos ressources Elastic Beanstalk. Vous pouvez utiliser des règles AWS Managed Config prédéfinies ou définir des règles personnalisées. AWS Config suit en permanence les modifications apportées à la configuration de vos ressources afin de déterminer si ces modifications enfreignent l'une des conditions de vos règles. La AWS Config console indique l'état de conformité de vos règles et de vos ressources.

Si une ressource enfreint une règle et est signalée comme non conforme, vous AWS Config pouvez être alertée via une rubrique Amazon Simple Notification Service (Amazon SNS). Pour utiliser par programmation les données contenues dans ces AWS Config alertes, utilisez une file d'attente Amazon Simple Queue Service (Amazon SQS) comme point de terminaison de notification pour la rubrique Amazon SNS. Par exemple, vous pouvez écrire du code qui démarre un flux de travail lorsque quelqu'un modifie la configuration de groupe Auto Scaling de votre environnement.

Pour en savoir plus sur la configuration et l'utilisation des règles, consultez la section Évaluation des ressources à l'aide de AWS Config règles dans le guide du AWS Config développeur.

Utilisation d'Elastic Beanstalk avec Amazon DynamoDB

Amazon DynamoDB est un service de base de données NoSQL entièrement géré, offrant des performances exceptionnelles et prévisibles en termes de rapidité et d'évolutivité. Si vous êtes développeur, vous pouvez utiliser DynamoDB pour créer une table de base de données pouvant stocker et récupérer n'importe quel volume de données, et traiter tout niveau de trafic de demandes. DynamoDB répartit automatiquement les données et le trafic de la table sur un nombre suffisant de serveurs afin de gérer la capacité de demandes spécifiée par le client et le volume de données stockées, tout en assurant la cohérence et la rapidité des performances. Tous les éléments de données sont stockés sur des disques SSD (SSDs) et sont automatiquement répliqués dans plusieurs zones de disponibilité d'une AWS région afin de garantir une haute disponibilité et une durabilité des données intégrées.

Si vous utilisez des tâches périodiques dans un environnement de travail, Elastic Beanstalk crée une table DynamoDB et l'utilise pour effectuer l'élection de leader et stocker des informations sur la tâche. Chaque instance de l'environnement tente d'écrire vers la table à intervalles réguliers de quelques secondes pour devenir leader et effectuer la tâche lorsqu'elle est prévue.

Vous pouvez utiliser des fichiers de configuration pour créer une table DynamoDB pour votre application. Consultez eb-node-express-samplela suite GitHub pour un exemple d'application Node.js qui crée une table avec un fichier de configuration et s'y connecte à l'aide du AWS SDK pour JavaScript Node.js. Pour obtenir un exemple de procédure d'utilisation de DynamoDB avec PHP, consultez Exemple : DynamoDB, et SNS CloudWatch. Pour un exemple utilisant le AWS SDK pour Java, voir Gérer l'état de session Tomcat avec DynamoDB dans la documentation. AWS SDK pour Java

Lorsque vous créez une table DynamoDB à l'aide de fichiers de configuration, cette table n'est pas rattachée au cycle de vie de votre environnement et n'est pas supprimée lorsque vous arrêtez votre environnement. Pour être sûr que des informations personnelles ne sont pas inutilement conservées, supprimez ces enregistrements lorsque vous n'en avez plus besoin ou supprimez la table.

Pour de plus amples informations sur DynamoDB, consultez le Manuel du développeur DynamoDB.

Utilisation d'Elastic Beanstalk avec Amazon ElastiCache

Amazon ElastiCache est un service Web qui permet de configurer, de gérer et de dimensionner des environnements de cache en mémoire distribués dans le cloud. Il fournit un cache en mémoire performant, évolutif et économique, tout en éliminant la complexité associée au déploiement et à la gestion d'un environnement de cache distribué. ElastiCache est conforme au protocole avec Redis et Memcached, de sorte que le code, les applications et les outils les plus populaires que vous utilisez aujourd'hui avec vos environnements Redis et Memcached existants fonctionneront parfaitement avec le service. Pour plus d'informations à ce sujet ElastiCache, rendez-vous sur la page ElastiCache produit d'Amazon.

Pour utiliser Elastic Beanstalk avec Amazon ElastiCache

1. Créez un ElastiCache cluster.

   - Pour obtenir des instructions sur la création d'un ElastiCache cluster avec Redis, consultez Getting Started with Amazon ElastiCache for Redis dans le guide de l'utilisateur ElastiCachepour Redis.
   - Pour obtenir des instructions sur la création d'un ElastiCache cluster avec Memcached, consultez Getting Started with Amazon ElastiCache for Memcached dans le guide de l'utilisateur ElastiCache pour Memcached.
2. Configurez votre groupe ElastiCache de sécurité pour autoriser l'accès depuis le groupe de EC2 sécurité Amazon utilisé par votre application Elastic Beanstalk. Pour savoir comment trouver le nom de votre groupe de EC2 sécurité à l'aide de la console AWS de gestion, consultez la EC2 groupes de sécurité page du document EC2 Instances.

   - Pour de plus amples informations sur Redis, veuillez consulter Autoriser l'accès dans le Guide de l'utilisateur ElastiCache pour Redis.
   - Pour de plus amples informations sur Memcached,veuillez consulter Autoriser l'accès dans le Guide de l'utilisateur ElastiCache pour Memcached.

Vous pouvez utiliser les fichiers de configuration pour personnaliser votre environnement Elastic Beanstalk à utiliser. ElastiCache Pour des exemples de fichiers de configuration qui s'intègrent ElastiCache à Elastic Beanstalk, consultez. Exemple : ElastiCache

Utilisation d'Elastic Beanstalk avec Amazon Elastic File System

Avec Amazon Elastic File System (Amazon EFS), vous pouvez créer des systèmes de fichiers réseau pouvant être montés par des instances dans plusieurs zones de disponibilité. Un système de fichiers Amazon EFS est une AWS ressource qui utilise des groupes de sécurité pour contrôler l'accès au réseau inclus dans votre VPC par défaut ou personnalisé.

Dans un environnement Elastic Beanstalk, vous pouvez utiliser Amazon EFS pour créer un répertoire partagé qui stocke les fichiers chargés ou modifiés par les utilisateurs de votre application. Votre application peut traiter un volume Amazon EFS monté tel que le stockage local. De cette façon, vous n'avez pas à modifier le code de votre application pour augmenter à plusieurs instances.

Pour plus d'informations sur Amazon EFS, consultez le Guide de l'utilisateur Amazon Elastic File System.

> **Note**
>
> Elastic Beanstalk crée un utilisateur d'application Web que vous pouvez configurer en tant que propriétaire des annuaires d'applications sur les instances Amazon. EC2 Pour plus d'informations, consultez Stockage permanent dans la rubrique Considérations relatives à la conception de ce guide.

Sections

- Fichiers de configuration
- Systèmes de fichiers chiffrés
- Exemples d’applications
- Nettoyage de systèmes de fichiers

Fichiers de configuration

Elastic Beanstalk fournit des fichiers de configuration que vous pouvez utiliser pour créer et monter des systèmes de fichiers Amazon EFS. Vous pouvez créer un volume Amazon EFS dans le cadre de votre environnement, ou monter un volume Amazon EFS que vous avez créé indépendamment d'Elastic Beanstalk.

- storage-efs-createfilesystem.config — Utilise la `Resources` clé pour créer un nouveau système de fichiers et des points de montage dans Amazon EFS. Toutes les instances de votre environnement peuvent se connecter au même système de fichiers pour bénéficier d'un stockage partagé et évolutif. Utilisez `storage-efs-mountfilesystem.config` pour monter le système de fichiers sur chaque instance.

  > **Ressources internes**
  >
  > Toutes les ressources que vous créez avec des fichiers de configuration sont liées au cycle de vie de votre environnement. Si vous résiliez votre environnement ou supprimez le fichier de configuration, ces ressources sont perdues.

- storage-efs-mountfilesystem.config — Montez un système de fichiers Amazon EFS sur un chemin local sur les instances de votre environnement. Vous pouvez créer le volume dans le cadre de l'environnement avec `storage-efs-createfilesystem.config`. Vous pouvez également le monter dans votre environnement à l'aide de la console Amazon EFS ou du AWS SDK. AWS CLI

Pour utiliser les fichiers de configuration, commencez par créer votre système de fichiers Amazon EFS avec `storage-efs-createfilesystem.config`. Suivez les instructions du fichier de configuration et ajoutez celui-ci au répertoire .ebextensions dans votre code source pour créer le système de fichiers dans votre VPC.

Déployez votre code source mis à jour dans votre environnement Elastic Beanstalk. Ceci permet de confirmer que le système de fichiers a été créé avec succès. Ensuite, ajoutez `storage-efs-mountfilesystem.config` pour monter le système de fichiers sur les instances de votre environnement. En faisant ceci dans deux déploiements distincts, vous garantissez que le système de fichiers reste intact si l'opération de montage échoue. Si vous effectuez ces deux étapes dans le même déploiement, un problème avec l'une des étapes entraînerait une mise hors service du système de fichiers si le déploiement échoue.

Systèmes de fichiers chiffrés

Amazon EFS prend en charge les systèmes de fichiers chiffrés. Le fichier de configuration `storage-efs-createfilesystem.config` présenté dans cette rubrique définit deux options personnalisées. Vous pouvez utiliser ces options pour créer un système de fichiers chiffré Amazon EFS. Pour plus d'informations, consultez les instructions du fichier de configuration.

Exemples d’applications

Elastic Beanstalk fournit également des exemples d'applications qui utilisent Amazon EFS pour le stockage partagé. Les deux projets contiennent des fichiers de configuration que vous pouvez utiliser avec un programme d'installation standard WordPress ou Drupal pour exécuter un blog ou un autre système de gestion de contenu dans un environnement à charge équilibrée. Lorsqu'un utilisateur charge une photo ou un autre support, le fichier est stocké dans un système de fichiers Amazon EFS. Cela évite d'avoir à utiliser l'alternative, qui consiste à utiliser un plugin pour stocker les fichiers chargés dans Simple Storage Service (Amazon S3).

- Équilibre de charge WordPress : cela inclut les fichiers de configuration permettant de l'installer WordPress en toute sécurité et de l'exécuter dans un environnement Elastic Beanstalk à charge équilibrée.
- Load-Balanced Drupal (Drupal à charge équilibrée) : il s'agit des fichiers de configuration et des instructions pour installer Drupal en toute sécurité et l'exécuter dans un environnement Elastic Beanstalk à charge équilibrée.

Nettoyage de systèmes de fichiers

Si vous avez créé un système de fichiers Amazon EFS qui utilise un fichier de configuration dans le cadre de votre environnement Elastic Beanstalk, Elastic Beanstalk supprime le système de fichiers lorsque vous arrêtez l'environnement. Pour réduire les coûts de stockage d'une application en cours d'exécution, supprimez systématiquement les fichiers dont votre application n'a pas besoin. Sinon, assurez-vous que le code de l'application maintient correctement le cycle de vie des fichiers.

> **Important**
>
> Si vous avez créé un système de fichiers Amazon EFS à l'extérieur d'un environnement Elastic Beanstalk et que vous l'avez monté sur les instances de l'environnement, Elastic Beanstalk ne le supprime pas lorsque vous arrêtez l'environnement. Pour vous assurer que vos informations personnelles ne sont pas conservées et éviter les coûts de stockage, supprimez les fichiers stockés par votre application si vous n'en avez plus besoin. Vous pouvez également supprimer l'ensemble du système de fichiers.

Utilisation d'Elastic Beanstalk avec Gestion des identités et des accès AWS

Gestion des identités et des accès AWS (IAM) vous aide à contrôler en toute sécurité l'accès à vos AWS ressources. Cette section inclut des documents de référence concernant l'utilisation des stratégies IAM, des profils d'instance et des rôles de service.

Pour obtenir une présentation des autorisations, consultez Rôles d'Elastic Beanstalk Service, profils d'instance et politiques utilisateur. Pour la plupart des environnements, le rôle de service et le profil d'instance que la console Elastic Beanstalk vous invite à créer lorsque vous lancez votre premier environnement disposent de toutes les autorisations nécessaires. De même, les stratégies gérées fournies par Elastic Beanstalk pour l'accès complet et l'accès en lecture seule incluent toutes les autorisations utilisateur requises pour un usage quotidien.

Le guide de l'utilisateur IAM fournit des informations détaillées sur les AWS autorisations.

Rubriques

- Gestion des profils d'instance Elastic Beanstalk
- Gestion des rôles de service Elastic Beanstalk
- Utilisation des rôles liés à un service pour Elastic Beanstalk
- Gestion des stratégies utilisateur Elastic Beanstalk
- Format Amazon Resource Name (ARN) pour Elastic Beanstalk
- Ressources et conditions pour les actions Elastic Beanstalk
- Utilisation de balises pour contrôler l'accès aux ressources Elastic Beanstalk
- Exemples de stratégies basées sur des stratégies gérées
- Exemples de stratégies basées sur des autorisations de ressource
- Empêcher l'accès aux compartiments Amazon S3 entre environnements

Gestion des profils d'instance Elastic Beanstalk

Un profil d'instance est un conteneur pour un rôle Gestion des identités et des accès AWS (IAM) que vous pouvez utiliser pour transmettre des informations de rôle à une instance Amazon EC2 au démarrage de l'instance.

Si votre AWS compte ne possède pas de profil d'instance EC2, vous devez en créer un à l'aide du service IAM. Vous pouvez ensuite attribuer le profil d'instance EC2 aux nouveaux environnements que vous créez. Les étapes de création d'un environnement de la console Elastic Beanstalk vous permettent d'accéder à la console IAM, afin que vous puissiez créer un profil d'instance EC2 avec les autorisations requises.

> **Note**
>
> Elastic Beanstalk avait précédemment créé un `aws-elasticbeanstalk-ec2-role` profil d'instance EC2 par défaut nommé la première fois AWS qu'un compte créait un environnement. Ce profil d'instance incluait des stratégies gérées par défaut. Si votre compte possède déjà ce profil d'instance, vous pourrez toujours l'attribuer à vos environnements. Cependant, les directives AWS de sécurité récentes n'autorisent pas un AWS service à créer automatiquement des rôles avec des politiques de confiance vis-à-vis d'autres AWS services, EC2 dans ce cas. En raison de ces directives de sécurité, Elastic Beanstalk ne crée plus de profil d'instance `aws-elasticbeanstalk-ec2-role` par défaut.

Politiques gérées

Elastic Beanstalk fournit plusieurs stratégies gérées pour permettre à votre environnement de répondre à différents cas d'utilisation. Pour répondre aux cas d'utilisation par défaut d'un environnement, ces stratégies doivent être associées au rôle du profil d'instance EC2.

- AWSElasticBeanstalkWebTier— Autorise l'application à télécharger des journaux sur Amazon S3 et des informations de débogage vers AWS X-Ray. Pour consulter le contenu des politiques gérées, reportez-vous AWSElasticBeanstalkWebTierau Guide de référence des politiques AWS gérées.
- AWSElasticBeanstalkWorkerTier— Accorde des autorisations pour le téléchargement des journaux, le débogage, la publication des métriques et les tâches relatives aux instances de travail, notamment la gestion des files d'attente, l'élection du leader et les tâches périodiques. Pour consulter le contenu des politiques gérées, reportez-vous AWSElasticBeanstalkWorkerTierau Guide de référence des politiques AWS gérées.
- AWSElasticBeanstalkMulticontainerDocker— Autorise Amazon Elastic Container Service à coordonner les tâches de cluster pour les environnements Docker. Pour consulter le contenu des politiques gérées, reportez-vous AWSElasticBeanstalkMulticontainerDockerau Guide de référence des politiques AWS gérées.

> **Important**
>
> Les stratégies gérées par Elastic Beanstalk ne fournissent pas d'autorisations granulaires : elles accordent toutes les autorisations potentiellement nécessaires à l'utilisation des applications Elastic Beanstalk. Dans certains cas, vous souhaiterez peut-être restreindre davantage les autorisations de nos politiques gérées. Pour un exemple d'un cas d'utilisation, voirEmpêcher l'accès aux compartiments Amazon S3 entre environnements.
> Nos stratégies gérées ne couvrent pas non plus les autorisations relatives aux ressources personnalisées que vous pourriez ajouter à votre solution et qui ne sont pas gérées par Elastic Beanstalk. Pour implémenter des autorisations plus granulaires, des autorisations minimales requises ou des autorisations de ressources personnalisées, utilisez des stratégies personnalisées.

Stratégie de relation de confiance pour EC2

Pour autoriser les instances EC2 dans votre environnement à assumer le rôle requis, le profil d'instance doit spécifier Amazon EC2 comme une entité de confiance dans la stratégie de relation d'approbation de la manière suivante :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Pour personnaliser les autorisations, vous pouvez soit ajouter des stratégies au rôle attaché au profil d'instance par défaut, soit créer votre propre profil d'instance avec un ensemble limité d'autorisations.

Pour utiliser la fonctionnalité d'analyse de l'environnement basée sur l'IA, ajoutez les `elasticbeanstalk:DescribeEnvironmentHealth` autorisations `bedrock:InvokeModelbedrock:ListFoundationModels`,`elasticbeanstalk:DescribeEvents`, et à votre profil d'instance. Ces autorisations permettent à Elastic Beanstalk d'utiliser Amazon Bedrock et d'accéder aux données de l'environnement pour analyser les journaux, les événements et l'état de santé des instances.

Sections

- Création d'un profil d'instance
- Ajout d'autorisations au profil d'instance par défaut
- Vérification des autorisations attribuées à votre profil d'instance
- Mettre à jour un profil d'instance par out-of-date défaut

Création d'un profil d'instance

Un profil d'instance est une enveloppe autour d'un rôle IAM standard qui permet à une instance EC2 d'assumer le rôle. Vous pouvez créer un profil d'instance avec les politiques gérées par défaut d'Elastic Beanstalk. Vous pouvez également créer des profils d'instance supplémentaires afin de personnaliser les autorisations pour différentes applications. Vous pouvez également créer un profil d'instance qui n'inclut pas les deux politiques gérées qui accordent des autorisations pour les environnements Docker gérés au niveau des travailleurs ou par ECS, si vous n'utilisez pas ces fonctionnalités.

Pour créer un profil d'instance avec les politiques gérées par défaut

1. Ouvrez la page Roles (Rôles) dans la console IAM.
2. Choisissez Créer un rôle.
3. Pour Type d’entité de confiance, choisissez Service AWS .
4. Pour le service ou le cas d'utilisation, choisissez Elastic Beanstalk.
5. Dans le cas d'utilisation, choisissez Elastic Beanstalk — Compute.
6. Choisissez Suivant.
7. Entrez un nom de rôle.

   Vous pouvez saisir le nom du rôle par défaut suggéré par la console Elastic Beanstalk :. `aws-elasticbeanstalk-ec2-role`

8. Vérifiez que les politiques d'autorisations incluent les éléments suivants, puis choisissez Next :
   - `AWSElasticBeanstalkWebTier`
   - `AWSElasticBeanstalkWorkerTier`
   - `AWSElasticBeanstalkMulticontainerDocker`
9. Choisissez Créer un rôle.

Pour créer un profil d'instance avec les politiques gérées de votre choix

1. Ouvrez la page Roles (Rôles) dans la console IAM.
2. Choisissez Créer un rôle.
3. Sous Trusted entity type (Type d'entité approuvée), choisissez service AWS .
4. Sous Cas d'utilisation, choisissez EC2.
5. Choisissez Suivant.
6. Associez les stratégies gérées adéquates fournies par Elastic Beanstalk et toutes les stratégies supplémentaires fournissant les autorisations dont votre application a besoin.
7. Choisissez Suivant.
8. Entrez un nom pour le rôle.
9. (Facultatif) Ajoutez des balises au rôle.
10. Choisissez Créer un rôle.

Ajout d'autorisations au profil d'instance par défaut

Si votre application accède à des ressources AWS APIs ou auxquelles les autorisations ne sont pas accordées dans le profil d'instance par défaut, ajoutez des politiques qui accordent des autorisations dans la console IAM.

Pour ajouter des stratégies au rôle attaché au profil d'instance par défaut

1. Ouvrez la page Roles (Rôles) dans la console IAM.
2. Choisissez le rôle attribué en tant que profil d'instance EC2.
3. Sous l'onglet Permissions (Autorisations), choisissez Attach policies (Attacher des politiques).
4. Sélectionnez la politique gérée relative aux services supplémentaires utilisés par votre application. Par exemple, `AmazonS3FullAccess` ou `AmazonDynamoDBFullAccess`.

5. Choisissez Attach policy (Attacher la politique).

Vérification des autorisations attribuées à votre profil d'instance

Les autorisations allouées à votre profil d'instance par défaut peuvent varier en fonction de la date à laquelle il a été créé, de la date du dernier lancement d'un environnement et du client que vous avez utilisé. Vous pouvez vérifier les autorisations sur le profil d'instance par défaut dans la console IAM.

Pour vérifier les autorisations du profil d'instance par défaut

1. Ouvrez la page Roles (Rôles) dans la console IAM.
2. Choisissez le rôle attribué en tant que profil d'instance EC2.
3. Sur l'onglet Autorisations, vérifiez la liste de politiques attachée au rôle.
4. Pour voir les autorisations octroyées par une stratégie, choisissez la stratégie.

Mettre à jour un profil d'instance par out-of-date défaut

Si le profil d'instance par défaut ne dispose pas des autorisations requises, vous pouvez ajouter manuellement les stratégies gérées au rôle attribué en tant que profil d'instance EC2.

Pour ajouter des stratégies gérées au rôle attaché au profil d'instance par défaut

1. Ouvrez la page Roles (Rôles) dans la console IAM.
2. Choisissez le rôle attribué en tant que profil d'instance EC2.
3. Sous l'onglet Permissions (Autorisations), choisissez Attach policies (Attacher des politiques).
4. Saisissez **`AWSElasticBeanstalk`** pour filtrer les stratégies.
5. Sélectionnez les politiques suivantes, puis choisissez Attacher une politique :

   - `AWSElasticBeanstalkWebTier`
   - `AWSElasticBeanstalkWorkerTier`
   - `AWSElasticBeanstalkMulticontainerDocker`

Gestion des rôles de service Elastic Beanstalk

Pour gérer et surveiller votre environnement, AWS Elastic Beanstalk exécute des actions sur les ressources de l'environnement en votre nom. Elastic Beanstalk a besoin de certaines autorisations

pour effectuer ces actions, et Gestion des identités et des accès AWS elle assume des rôles de service (IAM) pour obtenir ces autorisations.

Elastic Beanstalk a besoin d'utiliser des informations d'identification de sécurité temporaires chaque fois qu'il assume un rôle de service. Pour obtenir ces informations d'identification, Elastic Beanstalk envoie une requête à AWS Security Token Service (AWS STS) sur un point de terminaison spécifique à une région. Pour plus d'informations, consultez Informations d'identification de sécurité temporaires dans le Guide de l'utilisateur IAM.

> **Note**
>
> Si le AWS STS point de terminaison de la région où se trouve votre environnement est désactivé, Elastic Beanstalk envoie la demande sur un autre point de terminaison qui ne peut pas être désactivé. Ce point de terminaison est associé à une région différente. Il s'agit par conséquent d'une demande inter-région. Pour plus d'informations, consultez la section Activation et désactivation AWS STS dans une AWS région dans le guide de l'utilisateur IAM.

Gestion des rôles de service à l'aide de la console Elastic Beanstalk et de l'interface de ligne de commande (CLI) EB

Vous pouvez utiliser la console Elastic Beanstalk et l'interface de ligne de commande (CLI) EB pour définir des rôles de service pour votre environnement avec un ensemble suffisant d'autorisations. Ils créent une fonction du service par défaut et utilisent des politiques gérées.

Politiques gérées des fonctions du service

Elastic Beanstalk fournit une politique gérée pour la surveillance améliorée de l'état, et une autre qui comporte les autorisations supplémentaires requises pour les mises à jour gérées de la plateforme. La console et l'interface de ligne de commande (CLI) EB attribuent ces deux politiques à la fonction du service par défaut qu'elles créent pour vous. Ces politiques doivent être utilisées uniquement pour cette fonction du service par défaut. Elles ne doivent pas être utilisées avec d'autres utilisateurs ou rôles dans vos comptes.

`AWSElasticBeanstalkEnhancedHealth`

Cette politique accorde des autorisations pour qu'Elastic Beanstalk surveille l'état de l'instance et de l'environnement. Cette stratégie inclut également des actions Amazon SQS qui permettent à Elastic Beanstalk de surveiller l'activité de file d'attente pour les environnements de travail. Pour consulter le

contenu de cette stratégie gérée, consultez la AWSElasticBeanstalkEnhancedHealthpage du Guide de référence des politiques AWS gérées.

`AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy`

Cette politique octroie des autorisations à Elastic Beanstalk pour mettre à jour les environnements en votre nom afin d'effectuer des mises à jour de plateforme gérées. Pour consulter le contenu de cette stratégie gérée, consultez la AWSElasticBeanstalkManagedUpdatesCustomerRolePolicypage du Guide de référence des politiques AWS gérées.

Groupements d'autorisations de niveau service

Cette politique est groupée en instructions basées sur le jeu d'autorisations fourni.

- *`ElasticBeanstalkPermissions`*— Ce groupe d'autorisations permet d'appeler les actions du service Elastic Beanstalk (Elastic Beanstalk). APIs
- *`AllowPassRoleToElasticBeanstalkAndDownstreamServices`* – Ce groupe d'autorisations permet de transmettre n'importe quel rôle à Elastic Beanstalk et à d'autres services aval comme CloudFormation.
- *`ReadOnlyPermissions`* — Ce groupe d'autorisations sert à collecter des informations sur l'environnement en cours d'exécution.
- *`*OperationPermissions`* — Les groupes avec ce schéma de nommage servent à appeler les opérations nécessaires pour effectuer les mises à jour de la plateforme.
- *`*BroadOperationPermissions`* — Les groupes avec ce schéma de nommage servent à appeler les opérations nécessaires pour effectuer les mises à jour de la plateforme. Ils incluent également des autorisations étendues pour la prise en charge d'environnements hérités.
- *`*TagResource`*— Les groupes dotés de ce modèle de dénomination sont destinés aux appels qui utilisent le tag-on-create APIs pour attacher des balises aux ressources créées dans un environnement Elastic Beanstalk.

Pour afficher le contenu d'une politique gérée, vous pouvez aussi utiliser la page Policies (Politiques) de la console IAM.

> **Important**
>
> Les stratégies gérées par Elastic Beanstalk ne fournissent pas d'autorisations granulaires : elles accordent toutes les autorisations potentiellement nécessaires à l'utilisation des

applications Elastic Beanstalk. Dans certains cas, vous souhaiterez peut-être restreindre davantage les autorisations de nos politiques gérées. Pour un exemple d'un cas d'utilisation, voirEmpêcher l'accès aux compartiments Amazon S3 entre environnements.
Nos stratégies gérées ne couvrent pas non plus les autorisations relatives aux ressources personnalisées que vous pourriez ajouter à votre solution et qui ne sont pas gérées par Elastic Beanstalk. Pour implémenter des autorisations plus granulaires, des autorisations minimales requises ou des autorisations de ressources personnalisées, utilisez des stratégies personnalisées.

Politiques gérées par obsolètes

Dans le passé, Elastic Beanstalk AWSElasticBeanstalkServicesoutenait la politique relative aux rôles de service géré. Cette stratégie a été remplacée par AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy. Vous pouvez toujours voir et utiliser la politique précédente dans la console IAM.

Pour consulter le contenu des politiques gérées, reportez-vous AWSElasticBeanstalkServiceau Guide de référence des politiques AWS gérées.

Toutefois, nous vous recommandons de passer à l'utilisation de la nouvelle politique gérée (AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy). Ajoutez des politiques personnalisées pour accorder des autorisations aux ressources personnalisées, le cas échéant.

Utilisation de la console Elastic Beanstalk

Lorsque vous lancez un environnement dans la console Elastic Beanstalk, la console crée une fonction du service par défaut, appelé `aws-elasticbeanstalk-service-role`, auquel elle associe des politiques gérées comportant des autorisations par défaut.

Pour autoriser Elastic Beanstalk à assumer le rôle `aws-elasticbeanstalk-service-role`, la fonction du service spécifie Elastic Beanstalk comme entité de confiance dans la politique de relation d'approbation.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
      {
        "Sid": "",
        "Effect": "Allow",
        "Principal": {
          "Service": "elasticbeanstalk.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
          "StringEquals": {
            "sts:ExternalId": "elasticbeanstalk"
          }
        }
      }
    ]
}
```

Lorsque vous activez les mises à jour de plateforme gérée pour votre environnement, Elastic Beanstalk assume un rôle de service de mises à jour gérées distinct pour effectuer les mises à jour gérées. Par défaut, la console Elastic Beanstalk utilise le même rôle de service généré, `aws-elasticbeanstalk-service-role`, pour le rôle de service de mises à jour gérées. Si vous modifiez votre rôle de service par défaut, la console définit le rôle de service de mises à jour gérées pour qu'il utilise le rôle lié au service de mises à jour gérées, `AWSServiceRoleForElasticBeanstalkManagedUpdates`. Pour plus d'informations sur les rôles liés à un service, consultez the section called “Utilisation des rôles liés à un service”.

> **Note**
>
> En raison de problèmes d'autorisation, le service Elastic Beanstalk ne réussit pas toujours à créer ce rôle lié au service pour vous. Par conséquent, la console tente de le créer explicitement. Pour vous assurer que votre compte dispose de ce rôle lié au service, créez un environnement au moins une fois à l'aide de la console et configurez les mises à jour gérées pour qu'elles soient activées avant de créer l'environnement.

Utilisation de l'interface de ligne de commande (CLI) EB

Si vous lancez un environnement à l'aide de la commande the section called “eb create” de l'interface de ligne de commande Elastic Beanstalk (EB CLI) et que vous ne spécifiez pas de rôle de service au moyen de l'option `--service-role`, Elastic Beanstalk crée le rôle de service par défaut `aws-`

`elasticbeanstalk-service-role`. Si le rôle de service par défaut existe déjà, Elastic Beanstalk l'utilise pour le nouvel environnement. La console Elastic Beanstalk effectue également des actions similaires dans ces circonstances.

À la différence de la console, vous ne pouvez pas spécifier de rôle de service de mises à jour gérées lorsque vous utilisez une option de commande EB CLI. Si vous activez les mises à jour gérées pour votre environnement, vous devez définir le rôle de service de mises à jour gérées au moyen des options de configuration. L'exemple suivant active les mises à jour gérées et utilise le rôle de service par défaut en tant que rôle de service de mises à jour gérées.

Example extensions .eb/ .config managed-platform-update

```
option_settings:
  aws:elasticbeanstalk:managedactions:
    ManagedActionsEnabled: true
    PreferredStartTime: "Tue:09:00"
    ServiceRoleForManagedUpdates: "aws-elasticbeanstalk-service-role"
  aws:elasticbeanstalk:managedactions:platformupdate:
    UpdateLevel: patch
    InstanceRefreshEnabled: true
```

Gestion des rôles de service à l'aide de l'API Elastic Beanstalk

Lorsque vous utilisez l'action `CreateEnvironment` de l'API Elastic Beanstalk pour créer un environnement, spécifiez un rôle de service avec l'option de configuration `ServiceRole` dans l'espace de noms `aws:elasticbeanstalk:environment`. Pour plus d'informations sur l'utilisation de la surveillance améliorée de l'état avec l'API Elastic Beanstalk, consultez Utilisation des rapports améliorés sur l'état à l'aide de l'API Elastic Beanstalk.

En outre, si vous activez les mises à jour de plateforme gérée pour votre environnement, vous pouvez spécifier un rôle de service de mises à jour gérées à l'aide de l'option `ServiceRoleForManagedUpdates` de l'espace de noms `aws:elasticbeanstalk:managedactions`.

Utilisation des rôles liés à un service

Un rôle lié à un service est un type unique de rôle de service prédéfini par Elastic Beanstalk pour inclure toutes les autorisations dont le service a besoin pour appeler d'autres services en votre nom. AWS Le rôle lié à un service est associé à votre compte. Elastic Beanstalk le crée une fois,

puis le réutilise lors de la création d'environnements supplémentaires. Pour plus d'informations sur l'utilisation des rôles liés à un service avec les environnements Elastic Beanstalk, consultez Utilisation des rôles liés à un service pour Elastic Beanstalk.

Si vous créez un environnement à l'aide de l'API Elastic Beanstalk et que vous ne spécifiez pas de rôle de service, Elastic Beanstalk crée un rôle lié au service de surveillance pour votre compte, s'il n'en existe pas déjà un. Elastic Beanstalk utilise ce rôle pour le nouvel environnement. Vous pouvez également utiliser IAM pour créer à l'avance le rôle lié au service de surveillance de votre compte. Une fois que votre compte dispose de ce rôle, vous pouvez l'utiliser pour créer un environnement à l'aide de l'API Elastic Beanstalk, de la console Elastic Beanstalk ou de l'interface de ligne de commande (CLI) EB.

Si vous activez les mises à jour de plateforme gérées pour l'environnement et que vous spécifiez `AWSServiceRoleForElasticBeanstalkManagedUpdates` comme valeur pour l'option `ServiceRoleForManagedUpdates` de l'espace de noms `aws:elasticbeanstalk:managedactions`, Elastic Beanstalk crée un rôle lié au service des mises à jour gérées pour votre compte, s'il n'en n'existe pas déjà un. Elastic Beanstalk utilise le rôle pour effectuer des mises à jour gérées pour le nouvel environnement.

> **Note**
>
> Quand Elastic Beanstalk tente de créer les rôles de surveillance et de mises à jour gérées liées au service pour votre compte lorsque vous créez un environnement, vous devez disposer de l'autorisation `iam:CreateServiceLinkedRole`. Si vous ne disposez pas de cette autorisation, la création de l'environnement échoue et un message expliquant le problème s'affiche.
> En lieu et place, un autre utilisateur ayant l'autorisation de créer des rôles liés à un service peut utiliser IAM pour créer le rôle de service lié à l'avance. En employant cette méthode, vous n'avez pas besoin de l'autorisation `iam:CreateServiceLinkedRole` pour créer votre environnement.

Vérification des autorisations de rôle de service par défaut

Les autorisations accordées par votre rôle de service par défaut peuvent varier selon la date à laquelle il a été créé, la date du dernier lancement d'un environnement et le client que vous avez utilisé. Dans la console IAM, vous pouvez vérifier les autorisations accordées par le rôle de service par défaut.

Pour vérifier les autorisations du rôle de service par défaut

1. Dans la console IAM, ouvrez la page Roles (Rôles).
2. Sélectionnez aws-elasticbeanstalk-service-role.
3. Sur l'onglet Autorisations, vérifiez la liste de politiques attachée au rôle.
4. Pour afficher les autorisations octroyées à une politique, choisissez la politique.

Mettre à jour un rôle de service out-of-date par défaut

Si le rôle de service par défaut ne dispose pas des autorisations requises, vous pouvez le mettre à jour en créant un environnement dans la console de gestion d'environnement Elastic Beanstalk.

En variante, vous pouvez ajouter manuellement les politiques gérées à la fonction du service par défaut.

Pour ajouter des politiques gérées à la fonction du service par défaut

1. Dans la console IAM, ouvrez la page Roles (Rôles).
2. Sélectionnez aws-elasticbeanstalk-service-role.
3. Sous l'onglet Permissions (Autorisations), choisissez Attach policies (Attacher des politiques).
4. Saisissez **`AWSElasticBeanstalk`** pour filtrer les politiques.
5. Sélectionnez les politiques suivantes, puis choisissez Attacher une politique :
   - `AWSElasticBeanstalkEnhancedHealth`
   - `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy`

Ajout d'autorisations au rôle de service par défaut

Si votre application inclut des fichiers de configuration qui font référence à AWS des ressources pour lesquelles les autorisations ne sont pas incluses dans le rôle de service par défaut, Elastic Beanstalk peut avoir besoin d'autorisations supplémentaires. Ces autorisations supplémentaires sont nécessaires pour résoudre ces références lorsque le service traite les fichiers de configuration lors d'une mise à jour gérée. S'il manque des autorisations, la mise à jour échoue et Elastic Beanstalk renvoie un message indiquant les autorisations dont il a besoin. Suivez ces étapes pour ajouter des autorisations pour les services supplémentaires au rôle de service par défaut dans la console IAM.

Pour ajouter des politiques supplémentaires à la fonction du service par défaut

1. Dans la console IAM, ouvrez la page Roles (Rôles).
2. Sélectionnez aws-elasticbeanstalk-service-role.
3. Sous l'onglet Permissions (Autorisations), choisissez Attach policies (Attacher des politiques).
4. Sélectionnez la politique gérée relative aux services supplémentaires utilisés par votre application. Par exemple, `AmazonAPIGatewayAdministrator` ou `AmazonElasticFileSystemFullAccess`.
5. Choisissez Attach policy (Attacher la politique).

Création d'un rôle de service

Si vous ne pouvez pas utiliser le rôle de service par défaut, créez un rôle de service.

Pour créer un rôle de service

1. Dans la console IAM, ouvrez la page Roles (Rôles).
2. Choisissez Créer un rôle.
3. Sous AWS service (Service AWS), choisissez AWS Elastic Beanstalk, puis sélectionnez votre cas d'utilisation.
4. Choisissez Suivant : Autorisations.
5. Associez les politiques gérées `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy` et `AWSElasticBeanstalkEnhancedHealth` et toutes les politiques supplémentaires fournissant les autorisations dont votre application a besoin.
6. Choisissez Suivant : Balises.
7. (Facultatif) Ajoutez des balises au rôle.
8. Choisissez Suivant : Vérification.
9. Entrez un nom pour le rôle.
10. Choisissez Créer un rôle.

Appliquez votre rôle de service personnalisé lorsque vous créez un environnement à l'aide de l'assistant de création d'environnement ou de l'option `--service-role` de la commande `eb create`.

Utilisation des rôles liés à un service pour Elastic Beanstalk

AWS Elastic Beanstalk utilise des Gestion des identités et des accès AWS rôles liés à un service (IAM). Un rôle lié à un service est un type unique de rôle IAM lié directement à Elastic Beanstalk. Les rôles liés à un service sont prédéfinis par Elastic Beanstalk et incluent toutes les autorisations dont le service a besoin pour appeler d'autres services en votre nom. AWS

Elastic Beanstalk définit quelques types de rôles liés au service :

- Rôle lié à un service de surveillance – Permet à Elastic Beanstalk de surveiller l'état des environnements en cours d'exécution et de publier des notifications d'événements d'état.
- Rôle lié à un service de maintenance – Permet à Elastic Beanstalk d'effectuer régulièrement des activités de maintenance pour vos environnements en cours d'exécution.
- Rôle lié à un service de mises à jour gérées – Permet à Elastic Beanstalk d'effectuer des mises à jour planifiées de vos environnements en cours d'exécution.

Rubriques

- Rôle lié à un service de surveillance
- Rôle lié à un service de maintenance
- Rôle lié à un service de mises à jour gérées

Rôle lié à un service de surveillance

AWS Elastic Beanstalk utilise des Gestion des identités et des accès AWS rôles liés à un service (IAM). Un rôle lié à un service est un type unique de rôle IAM lié directement à Elastic Beanstalk. Les rôles liés à un service sont prédéfinis par Elastic Beanstalk et incluent toutes les autorisations dont le service a besoin pour appeler d'autres services en votre nom. AWS

Un rôle lié à un service simplifie la configuration d'Elastic Beanstalk, car vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Elastic Beanstalk définit les autorisations de ses rôles liés à un service et, sauf indication contraire, seul Elastic Beanstalk peut assumer ses rôles. Les autorisations définies comprennent la politique de confiance et la politique d’autorisation. De plus, cette politique d’autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Vous ressources Elastic Beanstalk sont ainsi protégées car vous ne pouvez pas involontairement supprimer d'autorisation pour accéder aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés à un service, consultez Services AWS qui fonctionnent avec IAM et recherchez les services pour lesquels la colonne Service-Linked Role (Rôle lié à un service) indique Yes (Oui). Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Autorisations des rôles liés à un service pour Elastic Beanstalk

Elastic Beanstalk utilise le rôle lié à un service AWSServiceRoleForElasticBeanstalknommé — Permet à Elastic Beanstalk de surveiller l'état des environnements de course et de publier des notifications d'événements liés à l'état de santé.

Le rôle AWSService RoleForElasticBeanstalk lié à un service fait confiance aux services suivants pour assumer le rôle :

- `elasticbeanstalk.amazonaws.com`

La politique d'autorisations du rôle AWSService RoleForElasticBeanstalk lié à un service contient toutes les autorisations dont Elastic Beanstalk a besoin pour effectuer des actions en votre nom :

AllowCloudformationReadOperationsOnElasticBeanstalkStacks

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCloudformationReadOperationsOnElasticBeanstalkStacks",
            "Effect": "Allow",
            "Action": [
                "cloudformation:DescribeStackResource",
                "cloudformation:DescribeStackResources",
                "cloudformation:DescribeStacks"
            ],
            "Resource": [
                "arn:aws:cloudformation:*:*:stack/awseb-*",
                "arn:aws:cloudformation:*:*:stack/eb-*"
            ]
        },
        {
            "Sid": "AllowOperations",
```

```
            "Effect": "Allow",
            "Action": [
                "autoscaling:DescribeAutoScalingGroups",
                "autoscaling:DescribeAutoScalingInstances",
                "autoscaling:DescribeNotificationConfigurations",
                "autoscaling:DescribeScalingActivities",
                "autoscaling:PutNotificationConfiguration",
                "ec2:DescribeInstanceStatus",
                "ec2:AssociateAddress",
                "ec2:DescribeAddresses",
                "ec2:DescribeInstances",
                "ec2:DescribeSecurityGroups",
                "elasticloadbalancing:DescribeInstanceHealth",
                "elasticloadbalancing:DescribeLoadBalancers",
                "elasticloadbalancing:DescribeTargetHealth",
                "elasticloadbalancing:DescribeTargetGroups",
                "sqs:GetQueueAttributes",
                "sqs:GetQueueUrl",
                "sns:Publish"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour en savoir plus, consultez Service-Linked Role Permissions (autorisations du rôle lié à un service) dans le Guide de l'utilisateur IAM.

Vous pouvez également utiliser une politique AWS gérée pour fournir un accès complet à Elastic Beanstalk.

Création d'un rôle lié à un service pour Elastic Beanstalk

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un environnement Elastic Beanstalk à l'aide de l'API Elastic Beanstalk et que vous ne spécifiez pas de rôle de service, Elastic Beanstalk crée un rôle lié à un service en votre nom.

> **Important**
>
> Si vous utilisiez le service Elastic Beanstalk avant le 27 septembre 2017, date à laquelle il a commencé à prendre en AWSService RoleForElasticBeanstalk charge le rôle lié au service, et que votre compte en avait besoin, Elastic Beanstalk a créé le rôle dans votre compte. AWSService RoleForElasticBeanstalk Pour plus d'informations, consultez A New Role Appeared in My IAM Account (Un nouveau rôle est apparu dans mon compte IAM).

Lorsque Elastic Beanstalk essaie de créer le rôle lié à AWSService RoleForElasticBeanstalk un service pour votre compte lorsque vous créez un environnement, vous devez en avoir l'autorisation. `iam:CreateServiceLinkedRole` Si vous n'avez pas cette autorisation, la création d'environnement échoue, et un message expliquant le problème s'affiche.

En lieu et place, un autre utilisateur ayant l'autorisation de créer des rôles liés à un service peut utiliser IAM pour précréer le rôle lié à un service à l'avance. Vous pouvez ensuite créer votre environnement, même sans avoir l'autorisation `iam:CreateServiceLinkedRole`.

Vous pouvez utiliser la console IAM pour créer un rôle lié à un service avec le cas d'utilisation Elastic Beanstalk. Dans l'interface de ligne de commande (CLI) IAM ou l'API IAM, créez un rôle lié à un service avec le nom de service `elasticbeanstalk.amazonaws.com`. Pour plus d'informations, consultez Création d'un rôle lié à un service dans le Guide de l'utilisateur IAM. Si vous supprimez ce rôle lié à un service, vous pouvez utiliser ce même processus pour créer le rôle à nouveau.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un environnement Elastic Beanstalk à l'aide de l'API Elastic Beanstalk et que vous ne spécifiez pas de rôle de service, Elastic Beanstalk crée à nouveau un rôle lié à un service en votre nom.

Modification d'un rôle lié à un service pour Elastic Beanstalk

Elastic Beanstalk ne vous permet pas de modifier le rôle lié à un service. AWSService RoleForElasticBeanstalk Après avoir créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez Modification d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Suppression d'un rôle lié à un service pour Elastic Beanstalk

Si vous n’avez plus besoin d’utiliser une fonctionnalité ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n’avez aucune entité inutilisée qui n’est pas surveillée ou gérée activement. Cependant, vous devez nettoyer les ressources de votre rôle lié à un service avant de pouvoir les supprimer manuellement.

Nettoyage d’un rôle lié à un service

Avant de pouvoir utiliser IAM pour supprimer un rôle lié à un service, vous devez d'abord vous assurer que tous les environnements Elastic Beanstalk utilisent un rôle de service différent ou sont arrêtés.

> **Note**
>
> Si le service Elastic Beanstalk utilise le rôle lié à un service lorsque vous essayez d'arrêter les environnements, l'arrêt peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour mettre fin à un environnement Elastic Beanstalk qui utilise la (console) AWSService RoleForElasticBeanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Terminate Environment (Résilier l’environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Consultez eb terminate pour plus de détails sur l'arrêt d'un environnement Elastic Beanstalk à l'aide de l'interface de ligne de commande (CLI) EB.

Consultez TerminateEnvironmentpour plus de détails sur la fin d'un environnement Elastic Beanstalk à l'aide de l'API.

Suppression manuelle du rôle lié au service

Utilisez la console IAM, la CLI IAM ou l'API IAM pour supprimer le rôle lié au AWSService RoleForElasticBeanstalk service. Pour en savoir plus, consultez Suppression d’un rôle lié à un service dans le Guide de l’utilisateur IAM.

Régions prises en charge pour les rôles liés à un service Elastic Beanstalk

Elastic Beanstalk prend en charge l'utilisation des rôles liés à un service dans toutes les régions où le service est disponible. Pour plus d'informations, consultez Points de terminaison et quotas AWS Elastic Beanstalk.

Rôle lié à un service de maintenance

AWS Elastic Beanstalk utilise des Gestion des identités et des accès AWS rôles liés à un service (IAM). Un rôle lié à un service est un type unique de rôle IAM lié directement à Elastic Beanstalk. Les rôles liés à un service sont prédéfinis par Elastic Beanstalk et incluent toutes les autorisations dont le service a besoin pour appeler d'autres services en votre nom. AWS

Un rôle lié à un service simplifie la configuration d'Elastic Beanstalk, car vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Elastic Beanstalk définit les autorisations de ses rôles liés à un service et, sauf indication contraire, seul Elastic Beanstalk peut assumer ses rôles. Les autorisations définies comprennent la politique de confiance et la politique d’autorisation. De plus, cette politique d’autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Vous ressources Elastic Beanstalk sont ainsi protégées car vous ne pouvez pas involontairement supprimer d'autorisation pour accéder aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés à un service, consultez Services AWS qui fonctionnent avec IAM et recherchez les services pour lesquels la colonne Service-Linked Role (Rôle lié à un service) indique Yes (Oui). Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Autorisations des rôles liés à un service pour Elastic Beanstalk

Elastic Beanstalk utilise le rôle lié à un service AWSServiceRoleForElasticBeanstalkMaintenancenommé — Permet à Elastic Beanstalk d'effectuer des activités de maintenance régulières pour vos environnements de course.

Le rôle AWSService RoleForElasticBeanstalkMaintenance lié à un service fait confiance aux services suivants pour assumer le rôle :

- `maintenance.elasticbeanstalk.amazonaws.com`

La politique d'autorisations du rôle AWSService RoleForElasticBeanstalkMaintenance lié à un service contient toutes les autorisations dont Elastic Beanstalk a besoin pour effectuer des actions en votre nom :

JSON

```
{
    "Version":"2012-10-17",
    "Statement":
        {
            "Sid":
 "AllowCloudformationChangeSetOperationsOnElasticBeanstalkStacks",
            "Effect": "Allow",
            "Action": [
                "cloudformation:CreateChangeSet",
                "cloudformation:DescribeChangeSet",
                "cloudformation:ExecuteChangeSet",
                "cloudformation:DeleteChangeSet",
                "cloudformation:ListChangeSets",
                "cloudformation:DescribeStacks"
            ],
            "Resource": [
                "arn:aws:cloudformation:*:*:stack/awseb-*",
                "arn:aws:cloudformation:*:*:stack/eb-*"
            ]
        }
}
```

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour en savoir plus, consultez Service-Linked Role Permissions (autorisations du rôle lié à un service) dans le Guide de l'utilisateur IAM.

Vous pouvez également utiliser une politique AWS gérée pour fournir un accès complet à Elastic Beanstalk.

Création d'un rôle lié à un service pour Elastic Beanstalk

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un environnement Elastic Beanstalk à l'aide de l'API Elastic Beanstalk et que vous ne spécifiez pas de profil d'instance, Elastic Beanstalk crée un rôle lié à un service en votre nom.

> **Important**
>
> Ce rôle lié à un service peut apparaître dans votre compte si vous avez effectué une action dans un autre service qui utilise les fonctions prises en charge par ce rôle. Si vous utilisiez le service Elastic Beanstalk avant le 18 avril 2019, date à laquelle il a commencé à prendre en AWSService RoleForElasticBeanstalkMaintenance charge le rôle lié au service, et que votre compte en avait besoin, Elastic Beanstalk a créé le rôle dans votre compte. AWSService RoleForElasticBeanstalkMaintenance Pour plus d'informations, consultez A New Role Appeared in My IAM Account (Un nouveau rôle est apparu dans mon compte IAM).

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un environnement Elastic Beanstalk à l'aide de l'API Elastic Beanstalk et que vous ne spécifiez pas de profil d'instance, Elastic Beanstalk crée à nouveau un rôle lié à un service en votre nom.

Modification d'un rôle lié à un service pour Elastic Beanstalk

Elastic Beanstalk ne vous permet pas de modifier le rôle lié à un service. AWSService RoleForElasticBeanstalkMaintenance Après avoir créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez Modification d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Suppression d'un rôle lié à un service pour Elastic Beanstalk

Si vous n'avez plus besoin d'utiliser une fonctionnalité ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer les ressources de votre rôle lié à un service avant de pouvoir les supprimer manuellement.

Nettoyage d’un rôle lié à un service

Avant que vous puissiez utiliser IAM pour supprimer un rôle lié à un service, vous devez d'abord arrêter tous les environnements Elastic Beanstalk qui utilisent le rôle.

> **Note**
>
> Si le service Elastic Beanstalk utilise le rôle lié à un service lorsque vous essayez d'arrêter les environnements, l'arrêt peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour mettre fin à un environnement Elastic Beanstalk qui utilise la (console) AWSService RoleForElasticBeanstalkMaintenance

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Terminate Environment (Résilier l’environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Consultez eb terminate pour plus de détails sur l'arrêt d'un environnement Elastic Beanstalk à l'aide de l'interface de ligne de commande (CLI) EB.

Consultez TerminateEnvironmentpour plus de détails sur la fin d'un environnement Elastic Beanstalk à l'aide de l'API.

Suppression manuelle du rôle lié au service

Utilisez la console IAM, la CLI IAM ou l'API IAM pour supprimer le rôle lié au AWSService RoleForElasticBeanstalkMaintenance service. Pour en savoir plus, consultez Suppression d’un rôle lié à un service dans le Guide de l’utilisateur IAM.

Régions prises en charge pour les rôles liés à un service Elastic Beanstalk

Elastic Beanstalk prend en charge l'utilisation des rôles liés à un service dans toutes les régions où le service est disponible. Pour plus d'informations, consultez Points de terminaison et quotas AWS Elastic Beanstalk.

Rôle lié à un service de mises à jour gérées

AWS Elastic Beanstalk utilise des Gestion des identités et des accès AWS rôles liés à un service (IAM). Un rôle lié à un service est un type unique de rôle IAM lié directement à Elastic Beanstalk. Les rôles liés à un service sont prédéfinis par Elastic Beanstalk et incluent toutes les autorisations dont le service a besoin pour appeler d'autres services en votre nom. AWS

Un rôle lié à un service simplifie la configuration d'Elastic Beanstalk, car vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Elastic Beanstalk définit les autorisations de ses rôles liés à un service et, sauf indication contraire, seul Elastic Beanstalk peut assumer ses rôles. Les autorisations définies comprennent la politique de confiance et la politique d'autorisation. De plus, cette politique d'autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Vous ressources Elastic Beanstalk sont ainsi protégées car vous ne pouvez pas involontairement supprimer d'autorisation pour accéder aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés à un service, consultez Services AWS qui fonctionnent avec IAM et recherchez les services pour lesquels la colonne Service-Linked Role (Rôle lié à un service) indique Yes (Oui). Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Autorisations des rôles liés à un service pour Elastic Beanstalk

Elastic Beanstalk utilise le rôle lié à un service AWSServiceRoleForElasticBeanstalkManagedUpdatesnommé — Permet à Elastic Beanstalk d'effectuer des mises à jour programmées de la plateforme de vos environnements de course.

Le rôle AWSService RoleForElasticBeanstalkManagedUpdates lié à un service fait confiance aux services suivants pour assumer le rôle :

- `managedupdates.elasticbeanstalk.amazonaws.com`

La politique gérée AWSElasticBeanstalkManagedUpdatesServiceRolePolicyaccorde au rôle AWSService RoleForElasticBeanstalkManagedUpdates lié au service toutes les autorisations dont Elastic Beanstalk a besoin pour effectuer des actions de mise à jour gérées en votre nom. Pour consulter le contenu des politiques gérées, consultez la AWSElasticBeanstalkManagedUpdatesServiceRolePolicypage du Guide de référence des politiques AWS gérées.

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour en savoir plus, consultez Service-Linked Role Permissions (autorisations du rôle lié à un service) dans le Guide de l'utilisateur IAM.

Vous pouvez également utiliser une politique AWS gérée pour fournir un accès complet à Elastic Beanstalk.

Création d'un rôle lié à un service pour Elastic Beanstalk

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un environnement Elastic Beanstalk à l'aide de l'API Elastic Beanstalk, activez les mises à jour gérées et que vous spécifiez `AWSServiceRoleForElasticBeanstalkManagedUpdates` comme valeur pour l'option `ServiceRoleForManagedUpdates` de l'espace de noms `aws:elasticbeanstalk:managedactions`, Elastic Beanstalk crée le rôle lié à un service pour vous.

Lorsque Elastic Beanstalk essaie de créer le rôle lié à AWSService RoleForElasticBeanstalkManagedUpdates un service pour votre compte lorsque vous créez un environnement, vous devez en avoir l'autorisation. `iam:CreateServiceLinkedRole` Si vous n'avez pas cette autorisation, la création d'environnement échoue, et un message expliquant le problème s'affiche.

En lieu et place, un autre utilisateur ayant l'autorisation de créer des rôles liés à un service peut utiliser IAM pour précréer le rôle lié à un service à l'avance. Vous pouvez ensuite créer votre environnement, même sans avoir l'autorisation `iam:CreateServiceLinkedRole`.

Vous pouvez utiliser la console IAM pour créer un rôle lié à un service avec le cas d'utilisation Elastic Beanstalk Managed Updates (Mises à jour gérées Elastic Beanstalk). Dans l'interface de ligne de commande (CLI) IAM ou l'API IAM, créez un rôle lié à un service avec le nom de service `managedupdates.elasticbeanstalk.amazonaws.com`. Pour plus d'informations, consultez Création d'un rôle lié à un service dans le Guide de l'utilisateur IAM. Si vous supprimez ce rôle lié à un service, vous pouvez utiliser ce même processus pour créer le rôle à nouveau.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un environnement Elastic Beanstalk à l'aide de l'API Elastic Beanstalk, activez les mises à jour gérées et que vous spécifiez `AWSServiceRoleForElasticBeanstalkManagedUpdates` comme valeur pour l'option `ServiceRoleForManagedUpdates` de l'espace de noms

`aws:elasticbeanstalk:managedactions`, Elastic Beanstalk crée à nouveau le rôle lié à un service pour vous.

Modification d'un rôle lié à un service pour Elastic Beanstalk

Elastic Beanstalk ne vous permet pas de modifier le rôle lié à un service. AWSService RoleForElasticBeanstalkManagedUpdates Après avoir créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez Modification d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Suppression d'un rôle lié à un service pour Elastic Beanstalk

Si vous n'avez plus besoin d'utiliser une fonctionnalité ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer les ressources de votre rôle lié à un service avant de pouvoir les supprimer manuellement.

Nettoyage d'un rôle lié à un service

Avant de pouvoir utiliser IAM pour supprimer un rôle lié à un service, vous devez d'abord vous assurer que les environnements Elastic Beanstalk pour lesquels les mises à jour gérées sont activées utilisent un rôle de service différent ou sont arrêtés.

> **Note**
>
> Si le service Elastic Beanstalk utilise le rôle lié à un service lorsque vous essayez d'arrêter les environnements, l'arrêt peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour mettre fin à un environnement Elastic Beanstalk qui utilise la (console) AWSService RoleForElasticBeanstalkManagedUpdates

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions, puis Terminate Environment (Résilier l'environnement).

4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Consultez eb terminate pour plus de détails sur l'arrêt d'un environnement Elastic Beanstalk à l'aide de l'interface de ligne de commande (CLI) EB.

Consultez TerminateEnvironmentpour plus de détails sur la fin d'un environnement Elastic Beanstalk à l'aide de l'API.

Suppression manuelle du rôle lié au service

Utilisez la console IAM, la CLI IAM ou l'API IAM pour supprimer le rôle lié au AWSService RoleForElasticBeanstalkManagedUpdates service. Pour en savoir plus, consultez Suppression d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Régions prises en charge pour les rôles liés à un service Elastic Beanstalk

Elastic Beanstalk prend en charge l'utilisation des rôles liés à un service dans toutes les régions où le service est disponible. Pour plus d'informations, consultez Points de terminaison et quotas AWS Elastic Beanstalk.

Gestion des stratégies utilisateur Elastic Beanstalk

AWS Elastic Beanstalk fournit deux politiques gérées qui vous permettent d'attribuer un accès complet ou un accès en lecture seule à toutes les ressources gérées par Elastic Beanstalk. Vous pouvez associer les politiques à des utilisateurs ou à des groupes Gestion des identités et des accès AWS (IAM), ou à des rôles assumés par vos utilisateurs.

Stratégies utilisateur gérées

- AdministratorAccess- AWSElastic Beanstalk : donne à l'utilisateur les autorisations administratives complètes pour créer, modifier et supprimer des applications Elastic Beanstalk, des versions d'applications, des paramètres de configuration, des environnements et leurs ressources sous-jacentes. Pour consulter le contenu des politiques gérées, consultez la page AdministratorAccess-AWSElastic Beanstalk du Guide de référence des politiques AWS gérées.
- AWSElasticBeanstalkReadOnly— Permet à l'utilisateur de visualiser les applications et les environnements, mais pas d'effectuer des opérations qui les modifient. Il fournit un accès en lecture seule à toutes les ressources Elastic Beanstalk, ainsi qu'aux autres ressources AWS récupérées par la console Elastic Beanstalk. Notez que l'accès en lecture seule ne permet pas d'effectuer des actions telles que le téléchargement des journaux Elastic Beanstalk en vue de

les lire. Cela est dû au fait que les journaux sont indexés dans le compartiment Amazon S3, sur lequel Elastic Beanstalk demanderait une autorisation en écriture. Pour de plus amples informations sur l'activation de l'accès aux journaux Elastic Beanstalk, veuillez consulter l'exemple situé à la fin de cette rubrique. Pour consulter le contenu des politiques gérées, consultez la AWSElasticBeanstalkReadOnlypage du Guide de référence des politiques AWS gérées.

> **Important**
>
> Les stratégies gérées par Elastic Beanstalk ne fournissent pas d'autorisations granulaires : elles accordent toutes les autorisations potentiellement nécessaires à l'utilisation des applications Elastic Beanstalk. Dans certains cas, vous souhaiterez peut-être restreindre davantage les autorisations de nos politiques gérées. Pour un exemple d'un cas d'utilisation, voirEmpêcher l'accès aux compartiments Amazon S3 entre environnements.
> Nos stratégies gérées ne couvrent pas non plus les autorisations relatives aux ressources personnalisées que vous pourriez ajouter à votre solution et qui ne sont pas gérées par Elastic Beanstalk. Pour implémenter des autorisations plus granulaires, des autorisations minimales requises ou des autorisations de ressources personnalisées, utilisez des stratégies personnalisées.

Politiques gérées par obsolètes

Auparavant, Elastic Beanstalk prenait en charge deux autres politiques utilisateur gérées, et. AWSElasticBeanstalkFullAccessAWSElasticBeanstalkReadOnlyAccess Nous prévoyons de retirer ces anciennes stratégies. Vous pouvez toujours les voir et les utiliser dans la console IAM. Néanmoins, nous vous recommandons de passer aux nouvelles stratégies utilisateur gérées et d'ajouter des stratégies personnalisées pour accorder des autorisations aux ressources personnalisées, le cas échéant.

Politiques d'intégration à d'autres services

Nous proposons également des politiques plus détaillées qui vous permettent d'intégrer votre environnement à d'autres services, si vous préférez les utiliser.

- AWSElasticBeanstalkRoleCWL — Permet à un environnement de gérer les groupes de CloudWatch journaux Amazon Logs.
- AWSElasticBeanstalkRoleRDS — Permet à un environnement d'intégrer une instance Amazon RDS.

- AWSElasticBeanstalkRoleWorkerTier— Permet à un niveau d'environnement de travail de créer une table Amazon DynamoDB et une file d'attente Amazon SQS.
- AWSElasticBeanstalkRoleECS — Permet à un environnement Docker à conteneurs multiples de gérer les clusters Amazon ECS.
- AWSElasticBeanstalkRoleCore— Autorise les opérations de base d'un environnement de service Web.
- AWSElasticBeanstalkRoleSNS — Permet à un environnement d'activer l'intégration des rubriques Amazon SNS.

Pour consulter la source JSON d'une stratégie gérée spécifique, consultez le Guide de référence des politiques AWS gérées.

Contrôle de l'accès à l'aide de stratégies gérées

Vous pouvez utiliser des stratégies gérées pour accorder un accès complet ou en lecture seule à Elastic Beanstalk. Elastic Beanstalk met à jour ces stratégies automatiquement lorsque des autorisations supplémentaires sont nécessaires pour accéder aux nouvelles fonctionnalités.

Pour appliquer une stratégie gérée à des utilisateurs ou groupes IAM

1. Ouvrez la page Stratégies dans la console IAM.
2. Dans la zone de recherche, tapez **`AWSElasticBeanstalk`** pour filtrer les stratégies.
3. Dans la liste des politiques, cochez la case située à côté de AWSElasticBeanstalkReadOnlyou AdministratorAccess- AWSElastic Beanstalk.
4. Sélectionnez Policy Actions (Actions de politique), puis sélectionnez Attach (Attacher).
5. Sélectionnez un ou plusieurs utilisateurs et groupes auxquelles attacher la stratégie. Vous pouvez utiliser le menu Filtre et la zone de recherche pour filtrer la liste des entités du principal.
6. Choisissez Attach policy (Attacher la politique).

Création d'une stratégie utilisateur personnalisée

Vous pouvez créer votre propre stratégie IAM pour autoriser ou refuser des actions d'API propres à Elastic Beanstalk sur des ressources Elastic Beanstalk spécifique et pour contrôler l'accès aux ressources personnalisées qui ne sont pas gérées par Elastic Beanstalk. Pour en savoir plus sur l'association d'une stratégie à un utilisateur ou à un groupe, veuillez consulter la section Utilisation

de stratégies du guide de l'utilisateur IAM. Pour en savoir plus sur la création d'une stratégie personnalisée, veuillez consulter la section Création de stratégies IAM du guide de l'utilisateur IAM.

> **Note**
>
> Vous pouvez limiter la façon dont un utilisateur interagit avec l'API Elastic Beanstalk. Toutefois, aucune méthode efficace ne peut empêcher les utilisateurs autorisés à créer les ressources sous-jacentes nécessaires de créer d'autres ressources dans Amazon EC2 et d'autres services.
> Vous devez considérer ces stratégies comme un moyen efficace pour répartir les responsabilités Elastic Beanstalk, et non comme un moyen de sécuriser toutes les ressources sous-jacentes.

> **Important**
>
> Si des politiques personnalisées sont attribuées à un rôle de service Elastic Beanstalk, il est important de lui attribuer les autorisations appropriées pour les modèles de lancement. Dans le cas contraire, il se peut que vous ne disposiez pas des autorisations requises pour mettre à jour un environnement ou en lancer un nouveau. Pour de plus amples informations, veuillez consulter Autorisations requises pour les modèles de lancement.

Une stratégie IAM contient des déclarations de stratégie décrivant les autorisations que vous souhaitez accorder. Lorsque vous créez une déclaration de stratégie pour Elastic Beanstalk, vous devez comprendre comment utiliser les quatre parties suivantes d'une déclaration de stratégie :

- Effect indique s'il faut autoriser ou refuser les actions spécifiées dans la déclaration.
- Action indique les opérations d'API que vous souhaitez contrôler. Par exemple, utilisez `elasticbeanstalk:CreateEnvironment` pour spécifier l'opération `CreateEnvironment`. Certaines opérations, telles que la création d'un environnement, nécessitent des autorisations supplémentaires pour effectuer ces actions. Pour de plus amples informations, veuillez consulter Ressources et conditions pour les actions Elastic Beanstalk.

  > **Note**
  >
  > Pour utiliser l'opération d'API `UpdateTagsForResource`, spécifiez l'une des deux actions virtuelles suivantes (ou les deux) au lieu du nom de l'opération d'API :

`elasticbeanstalk:AddTags`

Contrôle l'autorisation d'appeler `UpdateTagsForResource` et de transmettre la liste des balises à ajouter dans le paramètre `TagsToAdd`.

`elasticbeanstalk:RemoveTags`

Contrôle l'autorisation d'appeler `UpdateTagsForResource` et de transmettre une liste de clés de balises à supprimer dans le paramètre `TagsToRemove`.

- Resource spécifie les ressources auxquelles vous souhaitez contrôler l'accès. Pour spécifier des ressources Elastic Beanstalk, répertoriez l'Amazon Resource Name (ARN) de chaque ressource.
- (Facultatif) Condition indique les restrictions concernant l'autorisation accordée dans la déclaration. Pour de plus amples informations, veuillez consulter Ressources et conditions pour les actions Elastic Beanstalk.

Les sections suivantes illustrent quelques cas dans lesquels vous pouvez envisager d'utiliser une stratégie d'utilisateur personnalisée.

Activation de la création d'un environnement Elastic Beanstalk limité

L'exemple de stratégie suivant permet à un utilisateur appliquant cette stratégie d'appeler l'action `CreateEnvironment` afin de créer un environnement dont le nom commence par **`Test`**, avec l'application et la version d'application spécifiées.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"CreateEnvironmentPerm",
      "Action": [
        "elasticbeanstalk:CreateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My First
 Elastic Beanstalk Application/Test*"
      ],
```

```
      "Condition": {
        "StringEquals": {
          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:application/My First Elastic Beanstalk Application"],
          "elasticbeanstalk:FromApplicationVersion":
 ["arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My First
 Elastic Beanstalk Application/First Release"]
        }
      }
    },
    {
      "Sid":"AllNonResourceCalls",
      "Action":[
        "elasticbeanstalk:CheckDNSAvailability",
        "elasticbeanstalk:CreateStorageLocation"
      ],
      "Effect":"Allow",
      "Resource":[
        "*"
      ]
    }
  ]
}
```

La stratégie ci-dessus montre comment accorder un accès limité aux opérations Elastic Beanstalk. Pour lancer réellement un environnement, l'utilisateur doit être autorisé à créer les AWS ressources qui alimentent également l'environnement. Par exemple, la stratégie suivante accorde l'accès à l'ensemble de ressources par défaut pour un environnement de serveur web :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:*",
        "ecs:*",
        "elasticloadbalancing:*",
        "autoscaling:*",
```

```
        "cloudwatch:*",
        "s3:*",
        "sns:*",
        "cloudformation:*",
        "sqs:*"
        ],
      "Resource": "*"
    }
  ]
}
```

Activation de l'accès aux journaux Elastic Beanstalk stockés dans Amazon S3

La stratégie de l'exemple suivant permet à un utilisateur d'extraire des journaux Elastic Beanstalk, de les afficher dans Amazon S3 et de les récupérer.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3:DeleteObject",
        "s3:GetObjectAcl",
        "s3:PutObjectAcl"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::elasticbeanstalk-*"
    }
  ]
}
```

> **Note**
>
> Pour limiter ces autorisations aux chemins d'accès aux journaux, utilisez le format de ressource suivant.

```
"arn:aws:s3:::elasticbeanstalk-us-east-2-123456789012/resources/environments/
logs/*"
```

Activation de la gestion d'une application Elastic Beanstalk spécifique

L'exemple de stratégie suivant permet à un utilisateur de gérer des environnements et d'autres ressources dans une application Elastic Beanstalk spécifique. La stratégie refuse les actions Elastic Beanstalk sur les ressources des autres applications, et refuse également la création et la suppression des applications Elastic Beanstalk.

> **Note**
>
> La stratégie ne refuse pas l'accès aux ressources par le biais d'autres services. Elle représente un moyen efficace pour répartir les responsabilités de la gestion des applications Elastic Beanstalk entre les différents utilisateurs, et non un moyen de sécuriser les ressources sous-jacentes.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "elasticbeanstalk:CreateApplication",
        "elasticbeanstalk:DeleteApplication"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Deny",
      "Action": [
        "elasticbeanstalk:CreateApplicationVersion",
        "elasticbeanstalk:CreateConfigurationTemplate",
```

```
        "elasticbeanstalk:CreateEnvironment",
        "elasticbeanstalk:DeleteApplicationVersion",
        "elasticbeanstalk:DeleteConfigurationTemplate",
        "elasticbeanstalk:DeleteEnvironmentConfiguration",
        "elasticbeanstalk:DescribeApplicationVersions",
        "elasticbeanstalk:DescribeConfigurationOptions",
        "elasticbeanstalk:DescribeConfigurationSettings",
        "elasticbeanstalk:DescribeEnvironmentResources",
        "elasticbeanstalk:DescribeEnvironments",
        "elasticbeanstalk:DescribeEvents",
        "elasticbeanstalk:DeleteEnvironmentConfiguration",
        "elasticbeanstalk:RebuildEnvironment",
        "elasticbeanstalk:RequestEnvironmentInfo",
        "elasticbeanstalk:RestartAppServer",
        "elasticbeanstalk:RetrieveEnvironmentInfo",
        "elasticbeanstalk:SwapEnvironmentCNAMEs",
        "elasticbeanstalk:TerminateEnvironment",
        "elasticbeanstalk:UpdateApplicationVersion",
        "elasticbeanstalk:UpdateConfigurationTemplate",
        "elasticbeanstalk:UpdateEnvironment",
        "elasticbeanstalk:RetrieveEnvironmentInfo",
        "elasticbeanstalk:ValidateConfigurationSettings"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringNotEquals": {
          "elasticbeanstalk:InApplication": [
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/
myapplication"
          ]
        }
      }
    }
  ]
}
```

Format Amazon Resource Name (ARN) pour Elastic Beanstalk

Vous pouvez spécifier une ressource pour une stratégie IAM via l'Amazon Resource Name (ARN) de cette ressource. Pour Elastic Beanstalk, l'ARN a le format suivant.

```
arn:aws:elasticbeanstalk:region:account-id:resource-type/resource-path
```

Où :

- *region* est la région dans laquelle se situe la ressource (par exemple, **us-west-2**).
- *account-id*est l'identifiant du AWS compte, sans tiret (par exemple,) **123456789012**
- *resource-type* identifie le type de la ressource Elastic Beanstalk, par exemple, `environment`. Consultez la liste de tous les types de ressource Elastic Beanstalk dans le tableau ci-dessous.
- *resource-path* est la partie identifiant la ressource spécifique. Une ressource Elastic Beanstalk comporte un chemin qui identifie cette ressource de façon unique. Consultez le tableau ci-dessous afin de connaître le format du chemin de ressource pour chaque type de ressource. Par exemple, un environnement est toujours associé à une application. Le chemin de ressource pour l'environnement **myEnvironment** de l'application **myApp** se présente comme suit :

  ```
  myApp/myEnvironment
  ```

Elastic Beanstalk comporte plusieurs types de ressources, que vous pouvez spécifier dans une stratégie. Le tableau suivant affiche le format ARN de chacun d'eux, accompagné d'un exemple.

Type de ressource	Format de l'ARN
application	`arn:aws:elasticbeanstalk:` *region*`:`*account-id*` :application/` *application-name* Exemple : **arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App**
applicationversion	`arn:aws:elasticbeanstalk:` *region*`:`*account-id*` :applicationversion/` *application-name* `/`*version-label* Exemple : **arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version**
configurationtemplate	`arn:aws:elasticbeanstalk:` *region*`:`*account-id*` :configurationtemplate/` *application-name* `/`*template-name*

Type de ressource	Format de l'ARN
	Exemple : **arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/My Template**
environment	arn:aws:elasticbeanstalk: *region*:*account-id* :environment/ *application-name* /*environment-name* Exemple : **arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/MyEnvironment**
platform	arn:aws:elasticbeanstalk: *region*:*account-id* :platform/ *platform-name* /*platform-version* Exemple : **arn:aws:elasticbeanstalk:us-east-2:123456789012:platform/MyPlatform/1.0**
solutionstack	arn:aws:elasticbeanstalk: *region*::solutionstack/ *solutionstack-name* Exemple : **arn:aws:elasticbeanstalk:us-east-2::solutionstack/32bit Amazon Linux running Tomcat 7**

Une application spécifique contient toujours un environnement, une version d'application et un modèle de configuration. Vous remarquerez que ces ressources comportent toutes un nom d'application dans leur chemin de ressource, ce qui permet de les identifier de façon unique par leur nom de ressource et l'application qu'elles contiennent. Bien que les piles de solutions soient utilisées par les modèles de configuration et les environnements, les piles de solutions ne sont pas spécifiques à une application ou à un AWS compte et ne contiennent pas l'application ou le AWS compte. ARNs

Ressources et conditions pour les actions Elastic Beanstalk

Cette section décrit les ressources et les conditions que vous pouvez utiliser dans les déclarations de politique pour autoriser des autorisations qui permettent la réalisation d'actions Elastic Beanstalk sur des ressources Elastic Beanstalk spécifiques.

Ces conditions vous permettent de spécifier des autorisations pour les ressources que l'action doit réaliser. Par exemple, quand vous pouvez appeler l'action `CreateEnvironment`, vous devez

également spécifier la version d'application à déployer, ainsi que l'application qui contient le nom de cette application. Lorsque vous définissez des autorisations pour l'action `CreateEnvironment`, vous spécifiez l'application et la version de l'application sur lesquelles vous souhaitez que l'action agissent en utilisant les conditions `InApplication`et `FromApplicationVersion`.

En outre, vous pouvez spécifier la configuration de l'environnement avec une pile de solutions (`FromSolutionStack`) ou un modèle de configuration (`FromConfigurationTemplate`). La déclaration de politique suivante autorise l'action `CreateEnvironment` à créer un environnement avec le nom **`myenv`** (spécifié par `Resource`) dans l'application **`My App`** (spécifiée par la condition `InApplication`) à l'aide de la version de l'application **`My Version`** (`FromApplicationVersion`) avec une configuration **`32bit Amazon Linux running Tomcat 7`** (`FromSolutionStack`) :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "elasticbeanstalk:CreateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/
myenv"
      ],
      "Condition": {
        "StringEquals": {
          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:application/My App"],
          "elasticbeanstalk:FromApplicationVersion":
 ["arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My
 Version"],
          "elasticbeanstalk:FromSolutionStack": ["arn:aws:elasticbeanstalk:us-
east-2::solutionstack/32bit Amazon Linux running Tomcat 7"]
        }
      }
    }
  ]
}
```

> **Note**
>
> La plupart des clés de condition mentionnées dans cette rubrique sont spécifiques à Elastic Beanstalk, et leurs noms contiennent le préfixe `elasticbeanstalk:`. Par souci de concision, nous omettons ce préfixe dans les noms de clé de condition lorsque nous les mentionnons dans les sections suivantes. Par exemple, nous mentionnons `InApplication` au lieu du nom complet `elasticbeanstalk:InApplication`.
> En revanche, nous mentionnons quelques clés de condition utilisées dans tous les AWS services, et nous incluons leur `aws:` préfixe pour mettre en évidence l'exception.
> Les exemples de politiques présentent toujours les noms complets de clé de condition, y compris le préfixe.

Sections

- Informations sur la politique pour les actions Elastic Beanstalk
- Clés de condition pour les actions Elastic Beanstalk

Informations sur la politique pour les actions Elastic Beanstalk

Le tableau suivant répertorie toutes les actions Elastic Beanstalk, la ressource sur laquelle agit chaque action, ainsi que les informations contextuelles supplémentaires qui peuvent être fournies à l'aide de conditions.

Informations sur la politique pour les actions Elastic Beanstalk, y compris des ressources, des conditions, des exemples et des dépendances

Ressource	Conditions	Exemple de déclaration
Action : `AbortEnvironmentUpdate`		

`application`
 	
`environment`
	`aws:ResourceTag/` *key-name* (Facultatif)
	
	`aws:TagKeys` (Facultatif)
		La politique suivante autorise un utilisateur à interrompre des opérations de mise à jour d'environnement sur des environnements dans une application nommée My App.
		
		`{`
		`"Version":"2012-10-17",`

Ressource	Conditions	Exemple de déclaration
		"Statement": [{ "Action": ["elasticbeanstalk:AbortEnvironmentUpdate"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"] }] }
Action : CheckDNSAvailability		
"*"	N/A	{ "Version":"2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:CheckDNSAvailability"], "Effect": "Allow", "Resource": "*" }] }
Action : ComposeEnvironments		

Ressource	Conditions	Exemple de déclaration
`application`	`aws:ResourceTag/` *key-name* (Facultatif) `aws:TagKeys` (Facultatif)	La politique suivante permet à un utilisateur de composer des environnements qui appartiennent à une application nommée `My App`. `{ "Version":"2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:ComposeEnvironments" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/*" ] } ] }`
Action : `CreateApplication`		

Ressource	Conditions	Exemple de déclaration
`application`	`aws:RequestTag/` *`key-name`* (Facultatif) `aws:TagKeys` (Facultatif)	Cet exemple permet à l'action `CreateApplication` de créer des applications dont les noms commencent par **`DivA`** : `{ "Version":"2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:CreateApplication" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/DivA*" ] } ] }`
Action : `CreateApplicationVersion`		

Ressource	Conditions	Exemple de déclaration
`applicationversion`	`InApplication` `aws:RequestTag/` *`key-name`* (Facultatif) `aws:TagKeys` (Facultatif)	Cet exemple permet à l'action `CreateApplicationVersion` de créer des versions d'applications avec n'importe quel nom (*****) dans l'application **My App** : **{** **"Version":"2012-10-17",** **"Statement": [** **{** **"Action": [** **"elasticbeanstalk:CreateApplicationVersion"** **],** **"Effect": "Allow",** **"Resource": [** **"arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/*"** **],** **"Condition": {** **"StringEquals": {** **"elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"]** **}** **}** **}** **]** **}**
Action : [`CreateConfigurationTemplate`]		

Ressource	Conditions	Exemple de déclaration

`configurationtemplate`	`InApplication`
	
	`FromApplication`
	
	`FromApplicationVersion`
	
	`FromConfigurationTemplate`
	
	`FromEnvironment`
	
	`FromSolutionStack`
	
	`aws:RequestTag/` *key-name* (Facultatif)
	
	`aws:TagKeys` (facultatif)
		La politique suivante autorise l'action `CreateConfigurationTemplate` à créer des modèles de configuration dont le nom commence par **My Template** (`My Template*`) dans l'application **My App** :
		
		`{ "Version":"2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:CreateConfigurationTemplate"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/My Template*"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"], "elasticbeanstalk:FromSolutionStack": ["arn:aws:elasticbeanstalk:us-east-2::solutionstack/32bit Amazon Linux running Tomcat 7"] } } }] }`

Action : `CreateEnvironment`		

Ressource	Conditions	Exemple de déclaration

`environment`	`InApplication`
	
	`FromApplicationVersion`
	
	`FromConfigurationTemplate`
	
	`FromSolutionStack`
	
	`aws:RequestTag/` *key-name* (Facultatif)
	
	`aws:TagKeys` (facultatif)
		La politique suivante autorise l'action `CreateEnvironment` à créer un environnement dont le nom est **`myenv`** dans l'application **`My App`** et à l'aide de la pile de solutions **`32bit Amazon Linux running Tomcat 7`** :
		
		`{ "Version":"2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:CreateEnvironment"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"], "elasticbeanstalk:FromApplicationVersion": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"], "elasticbeanstalk:FromSolutionStack": ["arn:aws:elasticbeanstalk:us-east-2::solutionstack/32bit Amazon Linux running Tomcat 7"] } } }]`

Ressource	Conditions	Exemple de déclaration
		`}`
Action : `CreatePlatformVersion`		
`platform`	`aws:RequestTag/` *key-name* (Facultatif) `aws:TagKeys` (facultatif)	Cet exemple autorise l'action `CreatePlatformVersion` à créer des versions de plateforme ciblant la région `us-east-2` , dont le nom commence par **`us-east-2_`** : `{ "Version":"2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:CreatePlatformVersion" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:platform/us-east-2_*" ] } ] }`
Action : `CreateStorageLocation`		

Ressource	Conditions	Exemple de déclaration
`"*"`	N/A	`{ "Version":"2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:CreateStorageLocation" ], "Effect": "Allow", "Resource": "*" } ] }`
Action : `DeleteApplication`		
`application`	`aws:ResourceTag/` *key-name* (Facultatif) `aws:TagKeys` (facultatif)	La politique suivante autorise l'action `DeleteApplication` à supprimer l'application **`My App`** : `{ "Version":"2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:DeleteApplication" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App" ] } ] }`
Action : `DeleteApplicationVersion`		

Ressource	Conditions	Exemple de déclaration

`applicationversion`	`InApplication`
	
	`aws:ResourceTag/` *`key-name`* (Facultatif)
	
	`aws:TagKeys` (facultatif)
		La politique suivante autorise l'action `DeleteApplicationVersion` à supprimer une version de l'application dont le nom est **`My Version`** dans l'application **`My App`** :
		
		`{`
		`  "Version":"2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:DeleteApplicationVersion"`
		`  ],`
		`  "Effect": "Allow",`
		`  "Resource": [`
		`  "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"`
		`  ],`
		`  "Condition": {`
		`  "StringEquals": {`
		`  "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"]`
		`  }`
		`  }`
		`  }`
		`  ]`
		`}`

Action : [DeleteConfigurationTemplate](#)		

Ressource	Conditions	Exemple de déclaration
`configurationtemplate`	`InApplication` (Facultatif) `aws:ResourceTag/` *`key-name`* (facultatif) `aws:TagKeys` (facultatif)	La politique suivante autorise l'action `DeleteConfigurationTemplate` à supprimer un modèle de configuration dont le nom est **`My Template`** dans l'application **`My App`**. Spécifier le nom de l'application en tant que condition est facultatif. `{ "Version":"2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:DeleteConfigurationTemplate" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/My Template" ] } ] }`

Action : `DeleteEnvironmentConfiguration`

Ressource	Conditions	Exemple de déclaration
`environment`	`InApplication` (Facultatif)	La politique suivante autorise l'action `DeleteEnvironmentConfiguration` à supprimer une configuration de l'ébauche pour l'environnement **`myenv`** dans l'application **`My App`**. Spécifier le nom de l'application en tant que condition est facultatif. `{ "Version":"2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:DeleteEnvironmentConfiguration" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv" ] } ] }`
Action : `DeletePlatformVersion`		

Ressource	Conditions	Exemple de déclaration

`platform`	`aws:ResourceTag/` *key-name* (Facultatif)
	
	`aws:TagKeys` (facultatif)
		La politique suivante autorise l'action `DeletePlatformVersion` à supprimer les versions de plateforme ciblant la région `us-east-2`, dont le nom commence par **`us-east-2_`** :
		
		`{`
		`  "Version":"2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:DeletePlatformVersion"`
		`  ],`
		`  "Effect": "Allow",`
		`  "Resource": [`
		`  "arn:aws:elasticbeanstalk:us-east-2:123456789012:platform/us-east-2_*"`
		`  ]`
		`  }`
		`  ]`
		`}`

Action : `DescribeApplications`		

Ressource	Conditions	Exemple de déclaration
`application`	`aws:ResourceTag/` *key-name* (Facultatif) `aws:TagKeys` (facultatif)	La politique suivante autorise l'action `DescribeApplications` à décrire l'application My App. `{ "Version":"2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:DescribeApplications" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App" ] } ] }`
Action : `DescribeApplicationVersions`		

Ressource	Conditions	Exemple de déclaration
`applicationversion`	`InApplication` (Facultatif) `aws:ResourceTag/` *`key-name`* (facultatif) `aws:TagKeys` (facultatif)	La politique suivante autorise l'action `DescribeApplicationVersions` à décrire la version de l'application **`My Version`** dans l'application **`My App`**. Spécifier le nom de l'application en tant que condition est facultatif. `{ "Version":"2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:DescribeApplicationVersions" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version" ] } ] }`
Action : `DescribeConfigurationOptions`		

Ressource	Conditions	Exemple de déclaration

`environment`
 	
`configurationtemplate`
 	
`solutionstack`
	`InApplication` (Facultatif)
	
	`aws:ResourceTag/` *key-name* (facultatif)
	
	`aws:TagKeys` (facultatif)
		La politique suivante autorise l'action `DescribeConfigurationOptions` à décrire les options de configuration pour l'environnement **`myenv`** dans l'application **`My App`**. Spécifier le nom de l'application en tant que condition est facultatif.
		
		`{ "Version":"2012-10-17", "Statement": [ { "Action": "elasticbeanstalk:DescribeConfigurationOptions", "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv" ] } ] }`

Action : `DescribeConfigurationSettings`		

Ressource	Conditions	Exemple de déclaration

`environment`
 	
`configurationtemplate`
	`InApplication` (Facultatif)
	
	`aws:ResourceTag/` *`key-name`* (facultatif)
	
	`aws:TagKeys` (facultatif)
		La politique suivante autorise l'action `DescribeConfigurationSettings` à décrire les paramètres de configuration pour l'environnement **`myenv`** dans l'application **`My App`**. Spécifier le nom de l'application en tant que condition est facultatif.
		
		`{ "Version":"2012-10-17", "Statement": [ { "Action": "elasticbeanstalk:DescribeConfigurationSettings", "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv" ] } ] }`

Action : `DescribeEnvironmentHealth`		

Ressource	Conditions	Exemple de déclaration
`environment`	`aws:ResourceTag/` *key-name* (Facultatif) `aws:TagKeys` (facultatif)	La politique suivante permet l'utilisation de `DescribeEnvironmentHealth` pour récupérer des informations d'intégrité pour un environnement nommé **`myenv`**. `{ "Version":"2012-10-17", "Statement": [ { "Action": "elasticbeanstalk:DescribeEnvironmentHealth", "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv" ] } ] }`
Action : `DescribeEnvironmentResources`		

Ressource	Conditions	Exemple de déclaration
`environment`	`InApplication` (Facultatif) `aws:ResourceTag/` *`key-name`* (facultatif) `aws:TagKeys` (facultatif)	La politique suivante permet à l'`DescribeEnvironmentResources` action de renvoyer la liste des AWS ressources pour l'environnement **`myenv`** dans l'application**`My App`**. Spécifier le nom de l'application en tant que condition est facultatif. `{ "Version":"2012-10-17", "Statement": [ { "Action": "elasticbeanstalk:DescribeEnvironmentResources", "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv" ] } ] }`
Action : `DescribeEnvironments`		

Ressource	Conditions	Exemple de déclaration
`environment`	`InApplication` (Facultatif) `aws:ResourceTag/` *`key-name`* (facultatif) `aws:TagKeys` (facultatif)	La politique suivante autorise l'action `DescribeEnvironments` à décrire les environnements **`myenv`** et **`myotherenv`** dans l'application **`My App`**. Spécifier le nom de l'application en tant que condition est facultatif. `{ "Version":"2012-10-17", "Statement": [ { "Action": "elasticbeanstalk:DescribeEnvironments", "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv", "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App2/myotherenv" ] } ] }`
Action : `DescribeEvents`		

Ressource	Conditions	Exemple de déclaration

`application`
 	
`applicationversion`
 	
`configurationtemplate`
 	
`environment`
	`InApplication`
	
	`aws:ResourceTag/` *`key-name`* (Facultatif)
	
	`aws:TagKeys` (facultatif)
		La politique suivante autorise l'action `DescribeEvents` à répertorier les descriptions de l'événement pour l'environnement **`myenv`** et la version de l'application **`My Version`** dans l'application **`My App`**.
		
		`{`
		`  "Version":"2012-10-17",`
		`  "Statement": [`
		`    {`
		`      "Action": "elasticbeanstalk:DescribeEvents",`
		`      "Effect": "Allow",`
		`      "Resource": [`
		`        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv",`
		`        "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"`
		`      ],`
		`      "Condition": {`
		`        "StringEquals": {`
		`          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"]`
		`        }`
		`      }`
		`    }`
		`  ]`
		`}`

Action : `DescribeInstancesHealth`		

Ressource	Conditions	Exemple de déclaration

`environment`	N/A	La politique suivante permet l'utilisation de `DescribeInstancesHealth` pour récupérer des informations d'intégrité pour des instances dans un environnement nommé **`myenv`**.
		
		`{`
		`  "Version":"2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": "elasticbeanstalk:DescribeInstancesHealth",`
		`  "Effect": "Allow",`
		`  "Resource": [`
		`  "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"`
		`  ]`
		`  }`
		`  ]`
		`}`

Action : `DescribePlatformVersion`		

Ressource	Conditions	Exemple de déclaration

`platform`	`aws:ResourceTag/` *key-name* (Facultatif)
	
	`aws:TagKeys` (facultatif)
		La politique suivante autorise l'action `DescribePlatformVersion` pour décrire les versions de plateforme ciblant la région `us-east-2`, dont le nom commence par **`us-east-2_`** :
		
		`{`
		`  "Version":"2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:DescribePlatformVersion"`
		`  ],`
		`  "Effect": "Allow",`
		`  "Resource": [`
		`  "arn:aws:elasticbeanstalk:us-east-2:123456789012:platform/us-east-2_*"`
		`  ]`
		`  }`
		`  ]`
		`}`

Action : `ListAvailableSolutionStacks`		

Ressource	Conditions	Exemple de déclaration

`solutionstack`	N/A	La politique suivante autorise l'action `ListAvailableSolutionStacks` à renvoyer uniquement la pile de solutions **`32bit Amazon Linux running Tomcat 7`**.
		
		`{`
		`  "Version":"2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:ListAvailableSolutionStacks"`
		`  ],`
		`  "Effect": "Allow",`
		`  "Resource": "arn:aws:elasticbeanstalk:us-east-2::solutionstack/32bit Amazon Linux running Tomcat 7"`
		`  }`
		`  ]`
		`}`

Action : `ListPlatformVersions`		

Ressource	Conditions	Exemple de déclaration

`platform`	`aws:RequestTag/` *key-name* (Facultatif)
	
	`aws:TagKeys` (facultatif)
		Cet exemple autorise l'action `CreatePlatformVersion` à créer des versions de plateforme ciblant la région `us-east-2`, dont le nom commence par **`us-east-2_`** :
		
		`{`
		`  "Version":"2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:ListPlatformVersions"`
		`  ],`
		`  "Effect": "Allow",`
		`  "Resource": [`
		`  "arn:aws:elasticbeanstalk:us-east-2:123456789012:platform/us-east-2_*"`
		`  ]`
		`  }`
		`  ]`
		`}`

Action : `ListTagsForResource`		

Ressource	Conditions	Exemple de déclaration

`application`
 	
`applicationversion`
 	
`configurationtemplate`
 	
`environment`
 	
`platform`
	`aws:ResourceTag/` *`key-name`* (Facultatif)
	
	`aws:TagKeys` (facultatif)
		La politique suivante autorise l'action `ListTagsForResource` à afficher la liste des identifications des environnements existants uniquement s'ils possèdent une identification nommée `stage` et dont la valeur est `test` :
		
		`{ "Version":"2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:ListTagsForResource" ], "Effect": "Allow", "Resource": "*", "Condition": { "StringEquals": { "aws:ResourceTag/stage": ["test"] } } } ] }`

Action : `RebuildEnvironment`		

Ressource	Conditions	Exemple de déclaration
`environment`	`InApplication` `aws:ResourceTag/` *key-name* (Facultatif) `aws:TagKeys` (facultatif)	La politique suivante autorise l'action `RebuildEnvironment` à recréer l'environnement **myenv** dans l'application **My App**. `{ "Version":"2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:RebuildEnvironment"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"] } } }] }`
Action : `RequestEnvironmentInfo`		

Ressource	Conditions	Exemple de déclaration
`environment`	`InApplication` `aws:ResourceTag/` *key-name* (Facultatif) `aws:TagKeys` (facultatif)	La politique suivante autorise l'action `RequestEnvironmentInfo` à compiler des informations sur l'environnement **`myenv`** dans l'application **`My App`**. **{** **"Version":"2012-10-17",** **"Statement": [** **{** **"Action": [** **"elasticbeanstalk:RequestEnvironmentInfo"** **],** **"Effect": "Allow",** **"Resource": [** **"arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"** **],** **"Condition": {** **"StringEquals": {** **"elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"]** **}** **}** **}** **]** **}**
Action : `RestartAppServer`		

Ressource	Conditions	Exemple de déclaration

`environment`	`InApplication`	La politique suivante autorise l'action `RestartAppServer` à redémarrer le serveur conteneur d'application pour l'environnement **`myenv`** dans l'application **`My App`**.
		
		`{`
		`  "Version":"2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:RestartAppServer"`
		`  ],`
		`  "Effect": "Allow",`
		`  "Resource": [`
		`  "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"`
		`  ],`
		`  "Condition": {`
		`  "StringEquals": {`
		`  "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"]`
		`  }`
		`  }`
		`  }`
		`  ]`
		`}`

Action : `RetrieveEnvironmentInfo`		

Ressource	Conditions	Exemple de déclaration
`environment`	`InApplication` `aws:ResourceTag/` *`key-name`* (Facultatif) `aws:TagKeys` (facultatif)	La politique suivante autorise l'action `RetrieveEnvironmentInfo` à récupérer les informations compilées pour l'environnement **`myenv`** dans l'application **`My App`**. `{ "Version":"2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:RetrieveEnvironmentInfo"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"] } } }] }`
Action : `SwapEnvironmentCNAMEs`		

Ressource	Conditions	Exemple de déclaration
`environment`	`InApplication` (Facultatif) `FromEnvironment` (facultatif)	La politique suivante permet à l'`SwapEnvironmentCNAMEs` action de remplacer CNAMEs les environnements **`mysrcenv`** et**`mydestenv`** . `{ "Version":"2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:SwapEnvironmentCNAMEs" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/mysrcenv", "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/mydestenv" ] } ] }`
Action : `TerminateEnvironment`		

Ressource	Conditions	Exemple de déclaration
`environment`	`InApplication` `aws:ResourceTag/` *`key-name`* (Facultatif) `aws:TagKeys` (facultatif)	La politique suivante autorise l'action `TerminateEnvironment` à supprimer l'environnement **`myenv`** dans l'application **`My App`**. `{ "Version":"2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:TerminateEnvironment"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"] } } }] }`
Action : UpdateApplication		

Ressource	Conditions	Exemple de déclaration
`application`	`aws:ResourceTag/` *`key-name`* (Facultatif) `aws:TagKeys` (facultatif)	La politique suivante autorise l'action `UpdateApplication` à mettre à jour les propriétés de l'application **`My App`**. `{ "Version":"2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:UpdateApplication" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App" ] } ] }`
Action : UpdateApplicationResourceLifecycle		

Ressource	Conditions	Exemple de déclaration
`application`	`aws:ResourceTag/` *key-name* (Facultatif) `aws:TagKeys` (facultatif)	La politique suivante autorise l'action `UpdateApplicationResourceLifecycle` à mettre à jour les paramètres du cycle de vie de l'application **My App**. `{ "Version":"2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:UpdateApplicationResourceLifecycle" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App" ] } ] }`
Action : `UpdateApplicationVersion`		

Ressource	Conditions	Exemple de déclaration

`applicationversion`	`InApplication`
	
	`aws:ResourceTag/` *key-name* (Facultatif)
	
	`aws:TagKeys` (facultatif)
		La politique suivante autorise l'action `UpdateApplicationVersion` à mettre à jour les propriétés de la version de l'application **`My Version`** dans l'application **`My App`**.
		
		`{`
		`  "Version":"2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:UpdateApplicationVersion"`
		`  ],`
		`  "Effect": "Allow",`
		`  "Resource": [`
		`  "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"`
		`  ],`
		`  "Condition": {`
		`  "StringEquals": {`
		`  "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"]`
		`  }`
		`  }`
		`  }`
		`  ]`
		`}`

Action : [UpdateConfigurationTemplate](#)		

Ressource	Conditions	Exemple de déclaration
`configurationtemplate`	`InApplication` `aws:ResourceTag/` *key-name* (Facultatif) `aws:TagKeys` (facultatif)	La politique suivante autorise l'action `UpdateConfigurationTemplate` à mettre à jour les propriétés ou les options du modèle de configuration **`My Template`** dans l'application **`My App`**. **`{`** **`  "Version":"2012-10-17",`** **`  "Statement": [`** **`    {`** **`      "Action": [`** **`        "elasticbeanstalk:UpdateConfigurationTemplate"`** **`      ],`** **`      "Effect": "Allow",`** **`      "Resource": [`** **`        "arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/My Template"`** **`      ],`** **`      "Condition": {`** **`        "StringEquals": {`** **`          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"]`** **`        }`** **`      }`** **`    }`** **`  ]`** **`}`**
Action : `UpdateEnvironment`		

Ressource	Conditions	Exemple de déclaration

`environment`	`InApplication`
	
	`FromApplicationVersion`
	
	`FromConfigurationTemplate`
	
	`aws:ResourceTag/` *key-name* (Facultatif)
	
	`aws:TagKeys` (facultatif)
		La politique suivante autorise l'action `UpdateEnvironment` à mettre à jour l'environnement **`myenv`**dans l'application **`My App`** en déployant la version de l'application **`My Version`**.
		
		`{ "Version":"2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:UpdateEnvironment"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"], "elasticbeanstalk:FromApplicationVersion": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"] } } }] }`

Mesure : [`UpdateTagsForResource`] — `AddTags`		

Ressource	Conditions	Exemple de déclaration

`application`
 	
`applicationversion`
 	
`configurationtemplate`
 	
`environment`
 	
`platform`
	`aws:ResourceTag/` *key-name* (facultatif)
	
	`aws:RequestTag/` *key-name* (facultatif)
	
	`aws:TagKeys` (facultatif)
		L'action `AddTags` est l'une des deux actions virtuelles associées à l'API `UpdateTagsForResource`.
		
		La politique suivante autorise l'action `AddTags` à modifier des balises des environnements existants uniquement s'ils possèdent une balise nommée `stage` et dont la valeur est `test` :
		
		`{ "Version":"2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:AddTags" ], "Effect": "Allow", "Resource": "*", "Condition": { "StringEquals": { "aws:ResourceTag/stage": ["test"] } } } ] }`

Mesure : **`UpdateTagsForResource`** — `RemoveTags`		

Ressource	Conditions	Exemple de déclaration

`application`
 	
`applicationversion`
 	
`configurationtemplate`
 	
`environment`
 	
`platform`
	`aws:ResourceTag/` *`key-name`* (facultatif)
	
	`aws:TagKeys` (facultatif)
		L'action `RemoveTags` est l'une des deux actions virtuelles associées à l'API `UpdateTagsForResource`.
		
		La politique suivante empêche l'action `RemoveTags` de demander la suppression d'une balise nommée `stage` à partir des environnements existants :
		
		`{`
		`  "Version":"2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:RemoveTags"`
		`  ],`
		`  "Effect": "Deny",`
		`  "Resource": "*",`
		`  "Condition": {`
		`  "ForAnyValue:StringEquals": {`
		`  "aws:TagKeys": ["stage"]`
		`  }`
		`  }`
		`  }`
		`  ]`
		`}`

Action : `ValidateConfigurationSettings`		

Ressource	Conditions	Exemple de déclaration

`template`
 	
`environment`
	`InApplication`
	
	`aws:ResourceTag/` *key-name* (Facultatif)
	
	`aws:TagKeys` (facultatif)
		La politique suivante autorise l'action `ValidateConfigurationSettings` à valider les paramètres de configuration par rapport à l'environnement **myenv** dans l'application **My App**.
		
		`{ "Version":"2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:ValidateConfigurationSettings"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"] } } }] }`

Clés de condition pour les actions Elastic Beanstalk

Les clés vous permettent de spécifier des conditions qui expriment les dépendances, limitent les autorisations ou spécifient les contraintes sur les paramètres d'entrée pour une action. Elastic Beanstalk prend en charge les clés suivantes.

`InApplication`

Spécifie l'application qui contient la ressource sur laquelle l'action agit.

L'exemple suivant autorise l'action `UpdateApplicationVersion` à mettre à jour les propriétés de la version de l'application **`My Version`**. La condition `InApplication` spécifie **`My App`** comme le conteneur pour **`My Version`**.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "elasticbeanstalk:UpdateApplicationVersion"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My
 App/My Version"
      ],
      "Condition": {
        "StringEquals": {
          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:application/My App"]
        }
      }
    }
  ]
}
```

`FromApplicationVersion`

Spécifie une version de l'application comme une dépendance ou une contrainte sur un paramètre d'entrée.

L'exemple suivant autorise l'action `UpdateEnvironment` à mettre à jour l'environnement **`myenv`** dans l'application **`My App`**. La condition `FromApplicationVersion` contraint le paramètre `VersionLabel` à autoriser uniquement la version de l'application **`My Version`** pour mettre à jour l'environnement.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "elasticbeanstalk:UpdateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/
myenv"
      ],
      "Condition": {
        "StringEquals": {
          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:application/My App"],
          "elasticbeanstalk:FromApplicationVersion":
 ["arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/
My Version"]
        }
      }
    }
  ]
}
```

`FromConfigurationTemplate`

Spécifie un modèle de configuration comme une dépendance ou une contrainte sur un paramètre d'entrée.

L'exemple suivant autorise l'action `UpdateEnvironment` à mettre à jour l'environnement **`myenv`** dans l'application **`My App`**. La condition `FromConfigurationTemplate` contraint le paramètre `TemplateName` à autoriser uniquement le modèle de configuration **`My Template`** pour mettre à jour l'environnement.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Action": [
        "elasticbeanstalk:UpdateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/
myenv"
      ],
      "Condition": {
        "StringEquals": {
          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:application/My App"],
          "elasticbeanstalk:FromConfigurationTemplate":
 ["arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My
 App/My Template"]
        }
      }
    }
  ]
}
```

`FromEnvironment`

Spécifie un environnement comme une dépendance ou une contrainte sur un paramètre d'entrée.

L'exemple suivant permet à l'`SwapEnvironmentCNAMEs`action de remplacer le paramètre CNAMEs in **`My App`** pour tous les environnements dont le nom commence **`mysrcenv`** par**`mydestenv`**, mais pas pour les environnements dont le nom commence par **`mysrcenvPROD*`** et**`mydestenvPROD*`**.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "elasticbeanstalk:SwapEnvironmentCNAMEs"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/
mysrcenv*",
```

```
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/
mydestenv*"
      ],
      "Condition": {
        "ArnNotLike": {
          "elasticbeanstalk:FromEnvironment": [
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My
 App/mysrcenvPROD*",
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My
 App/mydestenvPROD*"
          ]
        }
      }
    }
  ]
}
```

`FromSolutionStack`

Spécifie une pile de solutions comme une dépendance ou une contrainte sur un paramètre d'entrée.

La politique suivante autorise l'action `CreateConfigurationTemplate` pour créer des modèles de configuration dont le nom commence par **`My Template`** (`My Template*`) dans l'application **`My App`**. La condition `FromSolutionStack` restreint le paramètre `solutionstack` pour autoriser uniquement la pile de solutions **`32bit Amazon Linux running Tomcat 7`** en tant que valeur d'entrée pour ce paramètre.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "elasticbeanstalk:CreateConfigurationTemplate"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-
east-2:123456789012:configurationtemplate/My App/My Template*"
      ],
      "Condition": {
```

```
        "StringEquals": {
          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:application/My App"],
          "elasticbeanstalk:FromSolutionStack": ["arn:aws:elasticbeanstalk:us-
east-2::solutionstack/32bit Amazon Linux running Tomcat 7"]
        }
      }
    }
  ]
}
```

`aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, `aws:TagKeys`

Spécifiez des conditions basées sur la balise. Pour plus d'informations, consultez Utilisation de balises pour contrôler l'accès aux ressources Elastic Beanstalk.

Utilisation de balises pour contrôler l'accès aux ressources Elastic Beanstalk

Cette rubrique explique comment le contrôle d'accès basé sur des balises peut vous aider à créer et à gérer des politiques IAM.

Nous pouvons utiliser les conditions figurant dans les déclarations de politique utilisateur d'IAM pour configurer les autorisations d'accès d'Elastic Beanstalk aux ressources. Pour en savoir plus sur les conditions des déclarations de politique, consultezRessources et conditions pour les actions Elastic Beanstalk. L'utilisation des balises dans les conditions est un moyen de contrôler l'accès aux ressources et demandes. Pour de plus amples informations sur le balisage des ressources Elastic Beanstalk, veuillez consulter Étiquette des ressources d'application Elastic Beanstalk.

Lorsque vous concevez des stratégies IAM, vous pouvez définir des autorisations granulaires en accordant l'accès à des ressources spécifiques. Au fur et à mesure que le nombre de ressources que vous gérez s'accroît, cette tâche devient plus difficile. Le balisage des ressources et l'utilisation de balises dans les déclarations de politique peuvent rendre cette tâche plus facile. Vous accordez l'accès en bloc à toute ressource avec une balise spécifique. Puis, vous appliquez cette balise à plusieurs reprises aux ressources correspondantes, lors de la création ou ultérieurement.

Les balises peuvent être attachées à la ressource ou transmises dans la demande aux services qui prennent en charge le balisage. Dans Elastic Beanstalk, les ressources et certaines actions peuvent comporter des balises. Lorsque vous créez une politique IAM, vous pouvez utiliser les clés de condition des balises pour contrôler les conditions suivantes :

- quels utilisateurs peuvent effectuer des actions sur un environnement, en fonction des balises qu'il possède déjà ;
- quelles balises peuvent être transmises dans une demande d'action ;
- si des clés de balise spécifiques peuvent être utilisées dans une demande.

Pour connaître la syntaxe complète et la sémantique des clés de condition de balise, consultez Contrôle de l’accès à l’aide de balises dans le Guide de l’utilisateur IAM.

Exemples de conditions relatives aux balises dans les politiques

Les exemples suivants montrent comment spécifier des conditions de balises dans les stratégies pour les utilisateurs Elastic Beanstalk.

Example 1 : Limiter les actions en fonction des balises dans la demande

La politique utilisateur gérée par AdministratorAccessElastic Beanstalk - AWSElastic Beanstalk donne aux utilisateurs des autorisations illimitées pour effectuer toute action Elastic Beanstalk sur n'importe quelle ressource gérée par Elastic Beanstalk.

La stratégie suivante refuse aux utilisateurs non autorisés l'autorisation de créer des environnements de production Elastic Beanstalk. Pour ce faire, elle refuse l'action `CreateEnvironment` si la demande spécifie une balise nommée `stage` avec une valeur `gamma` ou `prod`. En outre, la politique empêche ces utilisateurs non autorisés de modifier la phase d'environnements de production, en n'autorisant pas les actions de modification de balise visant à inclure ces mêmes valeurs de balise ou à supprimer entièrement la balise `stage`. L'administrateur d'un client peut attacher cette stratégie IAM aux utilisateurs IAM non autorisés et à la stratégie d'utilisateur gérée.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "elasticbeanstalk:CreateEnvironment",
        "elasticbeanstalk:AddTags"
      ],
```

```
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/stage": ["gamma", "prod"]
        }
      }
    },
    {
      "Effect": "Deny",
      "Action": [
        "elasticbeanstalk:RemoveTags"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:TagKeys": ["stage"]
        }
      }
    }
  ]
}
```

Example 2 : Limiter les actions en fonction des balises de ressource

La politique utilisateur gérée par AdministratorAccessElastic Beanstalk - AWSElastic Beanstalk donne aux utilisateurs des autorisations illimitées pour effectuer toute action Elastic Beanstalk sur n'importe quelle ressource gérée par Elastic Beanstalk.

La stratégie suivante refuse aux utilisateurs non autorisés l'autorisation d'effectuer des actions sur les environnements de production Elastic Beanstalk. Pour ce faire, elle refuse des actions spécifiques si l'environnement possède une balise nommée `stage` avec une valeur `gamma` ou `prod`. L'administrateur d'un client peut attacher cette stratégie IAM aux utilisateurs IAM non autorisés et à la stratégie d'utilisateur gérée.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
```

```
      "Action": [
        "elasticbeanstalk:AddTags",
        "elasticbeanstalk:RemoveTags",
        "elasticbeanstalk:DescribeEnvironments",
        "elasticbeanstalk:TerminateEnvironment",
        "elasticbeanstalk:UpdateEnvironment",
        "elasticbeanstalk:ListTagsForResource"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/stage": ["gamma", "prod"]
        }
      }
    }
  ]
}
```

Example 3 : Limiter les actions en fonction des balises dans la demande

La stratégie suivante accorde aux utilisateurs l'autorisation de créer des applications de développement Elastic Beanstalk.

Pour ce faire, il autorise les actions `CreateApplication` et `AddTags` si la demande spécifie une balise nommée `stage` avec la valeur `development`. La condition `aws:TagKeys` garantit que l'utilisateur ne peut pas ajouter d'autres clés de balise. En particulier, elle garantit la sensibilité à la casse de la clé de balise `stage`. Notez que cette politique est utile pour les utilisateurs IAM auxquels la politique utilisateur gérée d'Elastic Beanstalk - AWSElasticBeanstalk AdministratorAccess n'est pas jointe. La stratégie gérée fournit aux utilisateurs les autorisations complètes nécessaires pour effectuer une action Elastic Beanstalk sur une ressource gérée par Elastic Beanstalk.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "elasticbeanstalk:CreateApplication",
        "elasticbeanstalk:AddTags"
```

```
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/stage": "development"
        },
        "ForAllValues:StringEquals": {
          "aws:TagKeys": ["stage"]
        }
      }
    }
  ]
}
```

Example 4 : Limiter les actions en fonction des balises de ressource

La stratégie suivante accorde aux utilisateurs l'autorisation d'effectuer des actions et d'obtenir des informations sur les applications de développement Elastic Beanstalk.

Pour ce faire, il autorise les actions spécifiques si l'application a une balise nommée `stage` avec la valeur `development`. La condition `aws:TagKeys` garantit que l'utilisateur ne peut pas ajouter d'autres clés de balise. En particulier, elle garantit la sensibilité à la casse de la clé de balise `stage`. Notez que cette politique est utile pour les utilisateurs IAM auxquels la politique utilisateur gérée d'Elastic Beanstalk - AWSElasticBeanstalk AdministratorAccess n'est pas jointe. La stratégie gérée fournit aux utilisateurs les autorisations complètes nécessaires pour effectuer une action Elastic Beanstalk sur une ressource gérée par Elastic Beanstalk.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "elasticbeanstalk:UpdateApplication",
        "elasticbeanstalk:DeleteApplication",
        "elasticbeanstalk:DescribeApplications"
      ],
      "Resource": "*",
      "Condition": {
```

```
        "StringEquals": {
          "aws:ResourceTag/stage": "development"
        },
        "ForAllValues:StringEquals": {
          "aws:TagKeys": ["stage"]
        }
      }
    }
  ]
}
```

Exemples de stratégies basées sur des stratégies gérées

Cette section explique comment contrôler l'accès des utilisateurs à AWS Elastic Beanstalk et inclut des exemples de politiques fournissant l'accès requis pour les scénarios courants. Ces stratégies sont dérivées des stratégies gérées par Elastic Beanstalk. Pour plus d'informations sur l'attachement de stratégies gérées à des groupes et des utilisateurs, consultez Gestion des stratégies utilisateur Elastic Beanstalk.

Dans ce scénario, Example Corp. est une société de logiciels avec trois équipes chargées du site web d'entreprise : les administrateurs qui gèrent l'infrastructure, les développeurs qui construisent le logiciel du site web et une équipe d'assurance qualité qui teste le site web. Pour aider à gérer les autorisations pour ses ressources Elastic Beanstalk, Example Corp. crée trois groupes auxquels appartiennent les membres de chaque équipe respective : administrateurs, développeurs et testeurs. Example Corp. souhaite que le groupe des administrateurs dispose d'un accès complet à toutes les applications, tous les environnements et leurs ressources sous-jacentes afin qu'ils puissent créer, dépanner et supprimer toutes les ressources Elastic Beanstalk. Les développeurs ont besoin d'autorisations pour afficher toutes les ressources Elastic Beanstalk ainsi que créer et déployer des versions d'application. Les développeurs ne doivent pas pouvoir créer de nouvelles applications ou de nouveaux environnements ou encore interrompre des environnements en cours d'exécution. Les testeurs doivent consulter toutes les ressources Elastic Beanstalk pour surveiller et tester les applications. Les testeurs ne devraient pas être en mesure d'apporter des modifications aux ressources Elastic Beanstalk.

Les exemples de stratégies suivants fournissent les autorisations requises pour chaque groupe.

Exemple 1 : groupe d'administrateurs — All Elastic Beanstalk et services associés APIs

La stratégie suivante fournit aux utilisateurs des autorisations pour toutes les actions requises pour utiliser Elastic Beanstalk. Cette stratégie permet également à Elastic Beanstalk de mettre en service et de gérer des ressources en votre nom dans les services suivants. Elastic Beanstalk s'appuie sur ces services supplémentaires pour fournir des ressources sous-jacentes lors de la création d'un environnement.

- Amazon Elastic Compute Cloud
- Elastic Load Balancing
- Auto Scaling
- Amazon CloudWatch
- Amazon Simple Storage Service
- Amazon Simple Notification Service
- Amazon Relational Database Service
- CloudFormation

Notez que cette stratégie est un exemple. Elle offre un large ensemble d'autorisations pour les services AWS qu'Elastic Beanstalk utilise pour gérer les applications et les environnements. `ec2:*`Permet par exemple à un utilisateur Gestion des identités et des accès AWS (IAM) d'effectuer n'importe quelle action sur n'importe quelle ressource Amazon EC2 du compte. AWS Ces autorisations ne sont pas limitées aux ressources que vous utilisez avec Elastic Beanstalk. La bonne pratique consiste à accorder aux personnes autorisées uniquement les autorisations dont elles ont besoin pour réaliser leur travail.

JSON

```
{
  "Version":"2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "elasticbeanstalk:*",
        "ec2:*",
```

```
        "elasticloadbalancing:*",
        "autoscaling:*",
        "cloudwatch:*",
        "s3:*",
        "sns:*",
        "rds:*",
        "cloudformation:*"
      ],
      "Resource" : "*"
    }
  ]
}
```

Exemple 2 : Groupe de développeurs – Toutes les opérations à l'exception des opérations nécessitant des privilèges élevés

La stratégie suivante refuse l'autorisation de créer des applications et des environnements, mais autorise toutes les autres actions Elastic Beanstalk.

Notez que cette stratégie est un exemple. Elle offre un large ensemble d'autorisations aux produits AWS qu'Elastic Beanstalk utilise pour gérer des applications et des environnements. Par exemple, `ec2:*` permet à un utilisateur IAM d'effectuer n'importe quelle action sur n'importe quelle ressource Amazon EC2 du compte AWS . Ces autorisations ne sont pas limitées aux ressources que vous utilisez avec Elastic Beanstalk. La bonne pratique consiste à accorder aux personnes autorisées uniquement les autorisations dont elles ont besoin pour réaliser leur travail.

JSON

```
{
  "Version":"2012-10-17",
  "Statement" : [
    {
      "Action" : [
        "elasticbeanstalk:CreateApplication",
        "elasticbeanstalk:CreateEnvironment",
        "elasticbeanstalk:DeleteApplication",
        "elasticbeanstalk:RebuildEnvironment",
        "elasticbeanstalk:SwapEnvironmentCNAMEs",
        "elasticbeanstalk:TerminateEnvironment"],
      "Effect" : "Deny",
```

```
      "Resource" : "*"
    },
    {
      "Action" : [
        "elasticbeanstalk:*",
        "ec2:*",
        "elasticloadbalancing:*",
        "autoscaling:*",
        "cloudwatch:*",
        "s3:*",
        "sns:*",
        "rds:*",
        "cloudformation:*"],
      "Effect" : "Allow",
      "Resource" : "*"
    }
  ]
}
```

Exemple 3 : Testeurs – Affichage uniquement

La stratégie suivante autorise l'accès en lecture seule à toutes les applications, toutes les versions d'applications, tous les événements et tous les environnements. Elle ne permet d'effectuer aucune action.

JSON

```
{
  "Version":"2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "elasticbeanstalk:Check*",
        "elasticbeanstalk:Describe*",
        "elasticbeanstalk:List*",
        "elasticbeanstalk:RequestEnvironmentInfo",
        "elasticbeanstalk:RetrieveEnvironmentInfo",
        "ec2:Describe*",
        "elasticloadbalancing:Describe*",
        "autoscaling:Describe*",
```

```
        "cloudwatch:Describe*",
        "cloudwatch:List*",
        "cloudwatch:Get*",
        "s3:Get*",
        "s3:List*",
        "sns:Get*",
        "sns:List*",
        "rds:Describe*",
        "cloudformation:Describe*",
        "cloudformation:Get*",
        "cloudformation:List*",
        "cloudformation:Validate*",
        "cloudformation:Estimate*"
      ],
      "Resource" : "*"
    }
  ]
}
```

Exemples de stratégies basées sur des autorisations de ressource

Cette section décrit un cas d'utilisation pour contrôler les autorisations utilisateur pour les actions Elastic Beanstalk qui accèdent à des ressources Elastic Beanstalk spécifiques. Nous suivrons les exemples de stratégies qui prennent en charge le cas d'utilisation. Pour de plus amples informations sur les stratégies des ressources Elastic Beanstalk, veuillez consulter Création d'une stratégie utilisateur personnalisée. Pour plus d'informations sur l'attachement de stratégies à des utilisateurs et des groupes, consultez Gestion des stratégies IAM dans Utilisation d' Gestion des identités et des accès AWS.

Dans notre cas d'utilisation, Example Corp. est une petite agence de conseils développant des applications pour les deux clients différents. John est le gestionnaire de développement chargé du développement des deux applications Elastic Beanstalk : app1 et app2. John fait du développement et quelques tests sur les deux applications, et lui seul peut mettre à jour l'environnement de production pour les deux applications. Voici les autorisations dont il a besoin pour app1 et app2 :

- Afficher des applications, des versions d'applications, des environnements et des modèles de configuration
- Créer des versions d'applications et les déployer dans l'environnement intermédiaire

- Mettre à jour l'environnement de production
- Créer et résilier des environnements

Jill est testeur qui a besoin d'un accès pour afficher les ressources suivantes afin de surveiller et de tester les deux applications : applications, versions d'applications, environnements et modèles de configuration. Cependant, elle ne devrait pas être en mesure d'apporter des modifications aux ressources Elastic Beanstalk.

Jack est le développeur pour app1 qui a besoin d'un accès pour afficher toutes les ressources pour app1 ainsi que de créer des versions d'applications pour app1 et de les déployer dans l'environnement intermédiaire.

Judy est l'administratrice du AWS compte d'Example Corp. Elle a créé des utilisateurs IAM pour John, Jill et Jack et applique les politiques suivantes à ces utilisateurs afin d'accorder les autorisations appropriées aux applications app1 et app2.

Exemple 1 : John – gestionnaire de développement pour app1, app2

Nous avons divisé la stratégie de John en trois stratégies distinctes afin qu'elles soient plus faciles à lire et à gérer. Ensemble, elles donnent à John les autorisations dont il a besoin pour effectuer des actions de développement, de test et de déploiement sur les deux applications.

La première politique spécifie les actions pour Auto Scaling, Amazon S3, Amazon EC2, Amazon SNS CloudWatch, Elastic Load Balancing, Amazon RDS et. CloudFormation Elastic Beanstalk s'appuie sur ces services supplémentaires pour fournir des ressources sous-jacentes lors de la création d'un environnement.

Notez que cette stratégie est un exemple. Elle octroie un large éventail d'autorisations aux AWS produits utilisés par Elastic Beanstalk pour gérer les applications et les environnements. `ec2:*`Permet par exemple à un utilisateur IAM d'effectuer n'importe quelle action sur n'importe quelle ressource Amazon EC2 du compte. AWS Ces autorisations ne sont pas limitées aux ressources que vous utilisez avec Elastic Beanstalk. La bonne pratique consiste à accorder aux personnes autorisées uniquement les autorisations dont elles ont besoin pour réaliser leur travail.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
```

```
{
"Effect": "Allow",
"Action": [
"ec2:*",
"ecs:*",
"ecr:*",
"elasticloadbalancing:*",
"autoscaling:*",
"cloudwatch:*",
"s3:*",
"sns:*",
"cloudformation:*",
"dynamodb:*",
"rds:*",
"sqs:*",
"logs:*",
"iam:GetPolicyVersion",
"iam:GetRole",
"iam:ListRolePolicies",
"iam:ListAttachedRolePolicies",
"iam:ListInstanceProfiles",
"iam:ListRoles",
"iam:ListServerCertificates",
"acm:DescribeCertificate",
"acm:ListCertificates",
"codebuild:CreateProject",
"codebuild:DeleteProject",
"codebuild:BatchGetBuilds",
"codebuild:StartBuild"
],
"Resource": "*"
},
{
"Effect": "Allow",
"Action": [
"iam:PassRole"
],
"Resource": "arn:aws:iam::111122223333:role/MyRole"
}
]
}
```

La deuxième stratégie spécifie les actions Elastic Beanstalk que John est autorisé à effectuer sur les ressources app1 et app2. La déclaration `AllCallsInApplications` autorise toutes les actions Elastic Beanstalk (`"elasticbeanstalk:*"`) effectuées sur toutes les ressources au sein d'app1 et d'app2 (par exemple, `elasticbeanstalk:CreateEnvironment`). La déclaration `AllCallsOnApplications` autorise toutes les actions Elastic Beanstalk (`"elasticbeanstalk:*"`) sur les ressources d'application app1 et app2 (par exemple, `elasticbeanstalk:DescribeApplications`, `elasticbeanstalk:UpdateApplication`, etc..). La déclaration `AllCallsOnSolutionStacks` autorise toutes les actions Elastic Beanstalk (`"elasticbeanstalk:*"`) pour les ressources d'une pile de solutions (par exemple, `elasticbeanstalk:ListAvailableSolutionStacks`).

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"AllCallsInApplications",
         "Action":[
            "elasticbeanstalk:*"
         ],
         "Effect":"Allow",
         "Resource":[
            "*"
         ],
         "Condition":{
            "StringEquals":{
               "elasticbeanstalk:InApplication":[
                  "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/
app1",
                  "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/
app2"
               ]
            }
         }
      },
      {
         "Sid":"AllCallsOnApplications",
         "Action":[
            "elasticbeanstalk:*"
         ],
```

```
         "Effect":"Allow",
         "Resource":[
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1",
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app2"
         ]
      },
      {
         "Sid":"AllCallsOnSolutionStacks",
         "Action":[
            "elasticbeanstalk:*"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:elasticbeanstalk:us-east-2::solutionstack/*"
         ]
      }
   ]
}
```

La troisième stratégie spécifie les actions Elastic Beanstalk pour lesquelles la deuxième stratégie a besoin d'autorisations afin de réaliser ces actions Elastic Beanstalk. L'instruction `AllNonResourceCalls` autorise l'action `elasticbeanstalk:CheckDNSAvailability`, qui est obligatoire pour appeler `elasticbeanstalk:CreateEnvironment` et d'autres actions. Elle autorise également l'action `elasticbeanstalk:CreateStorageLocation`, qui est obligatoire pour `elasticbeanstalk:CreateApplication`, `elasticbeanstalk:CreateEnvironment` et d'autres actions.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"AllNonResourceCalls",
         "Action":[
            "elasticbeanstalk:CheckDNSAvailability",
            "elasticbeanstalk:CreateStorageLocation"
         ],
         "Effect":"Allow",
         "Resource":[
```

```
            "*"
         ]
      }
   ]
}
```

Exemple 2 : Jill – testeur pour app1, app2

Nous avons divisé la stratégie de Jill en trois stratégies distinctes afin qu'elles soient plus faciles à lire et à gérer. Ensemble, elles donnent à Jill les autorisations dont elle a besoin pour effectuer les actions de tests et de suivi sur les deux applications.

La première politique définit et définit `Describe*` les `Get*` actions relatives à Auto Scaling, Amazon S3, Amazon EC2, Amazon SNS CloudWatch, Elastic Load Balancing, Amazon RDS CloudFormation et (pour les types de conteneurs non traditionnels) afin que les actions Elastic Beanstalk soient en mesure de récupérer les informations pertinentes sur les ressources sous-jacentes des applications app1 et app2. `List*`

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "ec2:Describe*",
            "elasticloadbalancing:Describe*",
            "autoscaling:Describe*",
            "cloudwatch:Describe*",
            "cloudwatch:List*",
            "cloudwatch:Get*",
            "s3:Get*",
            "s3:List*",
            "sns:Get*",
            "sns:List*",
            "rds:Describe*",
            "cloudformation:Describe*",
         "cloudformation:Get*",
         "cloudformation:List*",
         "cloudformation:Validate*",
```

```
            "cloudformation:Estimate*"
            ],
            "Resource":"*"
         }
      ]
}
```

La deuxième stratégie spécifie les actions Elastic Beanstalk que Jill est autorisée à effectuer sur les ressources app1 et app2. L'instruction `AllReadCallsInApplications` lui permet d'appeler les actions `Describe*` et les actions d'informations d'environnement. L'instruction `AllReadCallsOnApplications` lui permet d'appeler les actions `DescribeApplications` et `DescribeEvents` sur les ressources d'application app1 et app2. L'instruction `AllReadCallsOnSolutionStacks` permet l'affichage d'actions qui impliquent des ressources d'une pile de solutions (`ListAvailableSolutionStacks`, `DescribeConfigurationOptions` et `ValidateConfigurationSettings`).

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"AllReadCallsInApplications",
         "Action":[
            "elasticbeanstalk:Describe*",
            "elasticbeanstalk:RequestEnvironmentInfo",
            "elasticbeanstalk:RetrieveEnvironmentInfo"
         ],
         "Effect":"Allow",
         "Resource":[
            "*"
         ],
         "Condition":{
            "StringEquals":{
               "elasticbeanstalk:InApplication":[
                  "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/
app1",
                  "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/
app2"
               ]
```

```
            }
         }
      },
      {
         "Sid":"AllReadCallsOnApplications",
         "Action":[
            "elasticbeanstalk:DescribeApplications",
            "elasticbeanstalk:DescribeEvents"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1",
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app2"
         ]
      },
      {
         "Sid":"AllReadCallsOnSolutionStacks",
         "Action":[
            "elasticbeanstalk:ListAvailableSolutionStacks",
            "elasticbeanstalk:DescribeConfigurationOptions",
            "elasticbeanstalk:ValidateConfigurationSettings"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:elasticbeanstalk:us-east-2::solutionstack/*"
         ]
      }
   ]
}
```

La troisième stratégie spécifie les actions Elastic Beanstalk pour lesquelles la deuxième stratégie a besoin d'autorisations afin de réaliser ces actions Elastic Beanstalk. L'instruction `AllNonResourceCalls` permet l'action `elasticbeanstalk:CheckDNSAvailability`, qui est obligatoire pour certaines actions d'affichage.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
```

```
         "Sid":"AllNonResourceCalls",
         "Action":[
            "elasticbeanstalk:CheckDNSAvailability"
         ],
         "Effect":"Allow",
         "Resource":[
            "*"
         ]
      }
   ]
}
```

Exemple 3 : Jack – développeur pour app1

Nous avons divisé la stratégie de Jack en trois stratégies distinctes afin qu'elles soient plus faciles à lire et à gérer. Ensemble, elles donnent à Jack les autorisations dont il a besoin pour effectuer des tests, des contrôles et des actions de déploiement sur la ressource app1.

La première politique spécifie les actions sur Auto Scaling, Amazon S3, Amazon EC2, Amazon SNS CloudWatch, Elastic Load Balancing, Amazon RDS CloudFormation et (pour les types de conteneurs non existants) afin que les actions Elastic Beanstalk puissent visualiser et utiliser les ressources sous-jacentes d'app1. Pour afficher la liste des types de conteneurs non hérités pris en charge, consultez the section called “Pourquoi certaines versions de plate-forme sont-elles marquées héritées ?”

Notez que cette stratégie est un exemple. Elle octroie un large éventail d'autorisations aux AWS produits utilisés par Elastic Beanstalk pour gérer les applications et les environnements. `ec2:*`Permet par exemple à un utilisateur IAM d'effectuer n'importe quelle action sur n'importe quelle ressource Amazon EC2 du compte. AWS Ces autorisations ne sont pas limitées aux ressources que vous utilisez avec Elastic Beanstalk. La bonne pratique consiste à accorder aux personnes autorisées uniquement les autorisations dont elles ont besoin pour réaliser leur travail.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
```

```
         "Action":[
            "ec2:*",
            "elasticloadbalancing:*",
            "autoscaling:*",
            "cloudwatch:*",
            "s3:*",
            "sns:*",
            "rds:*",
            "cloudformation:*"
         ],
         "Resource":"*"
      }
   ]
}
```

La deuxième stratégie spécifie les actions Elastic Beanstalk que Jack est autorisé à effectuer sur la ressource app1.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"AllReadCallsAndAllVersionCallsInApplications",
         "Action":[
            "elasticbeanstalk:Describe*",
            "elasticbeanstalk:RequestEnvironmentInfo",
            "elasticbeanstalk:RetrieveEnvironmentInfo",
            "elasticbeanstalk:CreateApplicationVersion",
            "elasticbeanstalk:DeleteApplicationVersion",
            "elasticbeanstalk:UpdateApplicationVersion"
         ],
         "Effect":"Allow",
         "Resource":[
            "*"
         ],
         "Condition":{
            "StringEquals":{
               "elasticbeanstalk:InApplication":[
```

```
                "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/
app1"
            ]
         }
      }
   },
   {
      "Sid":"AllReadCallsOnApplications",
      "Action":[
         "elasticbeanstalk:DescribeApplications",
         "elasticbeanstalk:DescribeEvents"
      ],
      "Effect":"Allow",
      "Resource":[
         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1"
      ]
   },
   {
      "Sid":"UpdateEnvironmentInApplications",
      "Action":[
         "elasticbeanstalk:UpdateEnvironment"
      ],
      "Effect":"Allow",
      "Resource":[
         "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/app1/
app1-staging*"
      ],
      "Condition":{
         "StringEquals":{
            "elasticbeanstalk:InApplication":[
               "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/
app1"
            ]
         },
         "ArnLike":{
            "elasticbeanstalk:FromApplicationVersion":[
               "arn:aws:elasticbeanstalk:us-
east-2:123456789012:applicationversion/app1/*"
            ]
         }
      }
   },
   {
      "Sid":"AllReadCallsOnSolutionStacks",
```

```
         "Action":[
            "elasticbeanstalk:ListAvailableSolutionStacks",
            "elasticbeanstalk:DescribeConfigurationOptions",
            "elasticbeanstalk:ValidateConfigurationSettings"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:elasticbeanstalk:us-east-2::solutionstack/*"
         ]
      }
   ]
}
```

La troisième stratégie spécifie les actions Elastic Beanstalk pour lesquelles la deuxième stratégie a besoin d'autorisations afin de réaliser ces actions Elastic Beanstalk. L'instruction `AllNonResourceCalls` autorise l'action `elasticbeanstalk:CheckDNSAvailability`, qui est obligatoire pour appeler `elasticbeanstalk:CreateEnvironment` et d'autres actions. Elle autorise également l'action `elasticbeanstalk:CreateStorageLocation`, qui est obligatoire pour `elasticbeanstalk:CreateEnvironment`, et d'autres actions.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"AllNonResourceCalls",
         "Action":[
            "elasticbeanstalk:CheckDNSAvailability",
            "elasticbeanstalk:CreateStorageLocation"
         ],
         "Effect":"Allow",
         "Resource":[
            "*"
         ]
      }
   ]
}
```

Empêcher l'accès aux compartiments Amazon S3 entre environnements

Cette rubrique explique comment les politiques gérées peuvent autoriser l'accès aux compartiments S3 entre environnements et comment créer des politiques personnalisées pour gérer ce type d'accès.

Elastic Beanstalk fournit des politiques gérées pour AWS gérer les ressources requises par les environnements Elastic Beanstalk de votre compte. AWS Les autorisations accordées par défaut à une application de votre AWS compte ont accès aux ressources S3 appartenant à d'autres applications du même AWS compte.

Si votre AWS compte exécute plusieurs applications Beanstalk, vous pouvez limiter la sécurité de vos politiques en créant votre propre politique personnalisée à associer à votre propre rôle de service ou profil d'instance pour chaque environnement. Vous pouvez ensuite limiter les autorisations S3 de votre politique personnalisée à un environnement spécifique.

> **Note**
>
> Sachez que vous êtes responsable du maintien de votre politique personnalisée. Si une politique gérée par Elastic Beanstalk sur laquelle repose votre politique personnalisée change, vous devrez modifier votre politique personnalisée en fonction des modifications respectives apportées à la politique de base. Pour un historique des modifications apportées aux politiques gérées par Elastic Beanstalk, consultez. Elastic AWS Beanstalk met à jour les politiques gérées

Exemple d'autorisations délimitées

L'exemple suivant est basé sur la politique AWSElasticBeanstalkWebTiergérée.

La politique par défaut inclut les lignes suivantes pour les autorisations d'accès aux compartiments S3. Cette politique par défaut ne limite pas les actions du compartiment S3 à des environnements ou à des applications spécifiques.

```
{
   "Sid" : "BucketAccess",
   "Action" : [
      "s3:Get*",
      "s3:List*",
      "s3:PutObject"
```

```
      ],
   "Effect" : "Allow",
   "Resource" : [
      "arn:aws:s3:::elasticbeanstalk-*",
      "arn:aws:s3:::elasticbeanstalk-*/*"
      ]
}
```

Vous pouvez limiter l'accès en qualifiant des ressources spécifiques à un rôle de service spécifié en tant que`Principal`. L'exemple suivant fournit les `aws-elasticbeanstalk-ec2-role-my-example-env` autorisations de rôle de service personnalisées pour les compartiments S3 dans l'environnement avec un identifiant`my-example-env-ID`.

Example Accorder des autorisations uniquement aux compartiments S3 d'un environnement spécifique

```
{
   "Sid": "BucketAccess",
   "Action": [
      "s3:Get*",
      "s3:List*",
      "s3:PutObject"
    ],
   "Effect": "Allow",
   "Principal": {
      "AWS": "arn:aws:iam::...:role/aws-elasticbeanstalk-ec2-role-my-example-env"
     },
   "Resource": [
      "arn:aws:s3:::elasticbeanstalk-my-region-account-id-12345",
      "arn:aws:s3:::elasticbeanstalk-my-region-account-id-12345/resources/environments/
my-example-env-ID/*"
    ]
}
```

> **Note**
>
> L'ARN de la ressource doit inclure l'ID de l'environnement Elastic Beanstalk (et non le nom de l'environnement). Vous pouvez obtenir l'identifiant de l'environnement depuis la console Elastic Beanstalk sur la page de présentation de l'environnement. Vous pouvez également utiliser la commande AWS CLI describe-environments pour obtenir ces informations.

Pour plus d'informations afin de vous aider à mettre à jour les autorisations du bucket S3 pour vos environnements Elastic Beanstalk, consultez les ressources suivantes :

- Utilisation d'Elastic Beanstalk avec Amazon S3 dans ce guide
- Types de ressources définis par Amazon S3 dans le guide de référence des autorisations de service
- Format ARN dans le guide de l'utilisateur IAM

Utilisation d'Elastic Beanstalk avec Amazon RDS

Cette section explique comment utiliser Elastic Beanstalk avec Amazon Relational Database Service (Amazon RDS) pour configurer, exploiter et dimensionner une base de données relationnelle. Nous expliquons certains concepts relatifs à la configuration et formulons des recommandations. Nous vous expliquerons ensuite le processus de création et de connexion à un Amazon RDS.

Deux options s'offrent à vous pour commencer :

- Créez une nouvelle base de données dans Amazon RDS.
- Commencez avec une base de données précédemment créée par Elastic Beanstalk et découplée par la suite d'un environnement Beanstalk. Pour de plus amples informations, veuillez consulter the section called “Base de données”.

Sélectionnez l'approche

Vous pouvez utiliser l'une ou l'autre approche pour exécuter une instance de base de données dans Amazon RDS et configurer votre application pour qu'elle s'y connecte lors du lancement. Vous pouvez connecter plusieurs environnements à une base de données.

> **Note**
>
> Si vous n'avez encore jamais utilisé d'instance de base de données avec votre application, nous vous recommandons d'abord d'ajouter votre base de données à un environnement de test à l'aide de la console Elastic Beanstalk. Cette opération vous permet de vérifier que votre application peut lire les propriétés de l'environnement, créer une chaîne de connexion et se connecter à une instance de base de données sans le travail de configuration supplémentaire requis pour une base de données autonome. Pour de plus amples informations, veuillez consulter Ajout d'une base de données à votre environnement Elastic Beanstalk.

Configurer un groupe de sécurité

Pour permettre aux EC2 instances Amazon de votre environnement de se connecter à une base de données externe, configurez un groupe de sécurité supplémentaire pour le groupe Auto Scaling associé à votre environnement. Vous pouvez attacher le même groupe de sécurité attaché à votre instance de base de données. Vous pouvez également utiliser un groupe de sécurité distinct. Si vous attachez un groupe de sécurité différent, vous devez configurer le groupe de sécurité attaché à votre base de données pour autoriser l'accès entrant à partir de ce groupe de sécurité.

> **Note**
>
> Vous pouvez connecter votre environnement à une base de données en ajoutant une règle au groupe de sécurité attaché à votre base de données. Cette règle doit autoriser l'accès entrant à partir du groupe de sécurité généré automatiquement qu'Elastic Beanstalk attache au groupe Auto Scaling pour votre environnement. Toutefois, sachez qu'en créant cette règle, vous créez également une dépendance entre les deux groupes de sécurité. Par conséquent, lorsque vous essayez de résilier l'environnement, Elastic Beanstalk ne peut pas supprimer le groupe de sécurité de l'environnement, car le groupe de sécurité de la base de données en dépend.

Configuration de la connexion à la base de données

Une fois que vous avez lancé votre instance de base de données et configuré les groupes de sécurité, vous pouvez transmettre les informations de connexion, telles que le point de terminaison et le mot de passe, à votre application via les propriétés d'environnement. Il s'agit du même mécanisme que celui utilisé en arrière-plan par Elastic Beanstalk lorsque vous exécutez une instance de base de données dans votre environnement.

Pour bénéficier d'une couche de sécurité supplémentaire, vous pouvez stocker vos informations de connexion dans Amazon S3 et configurer Elastic Beanstalk pour les récupérer au cours du déploiement. Avec les fichiers de configuration (`.ebextensions`), vous pouvez configurer les instances de votre environnement afin de récupérer en toute sécurité des fichiers depuis Amazon S3 lorsque vous déployez votre application.

Rubriques

- Lancement et connexion d'une instance Amazon RDS externe dans un VPC par défaut
- Stockage des informations d'identification Amazon RDS dans AWS Secrets Manager

- Nettoyage d'une instance Amazon RDS externe

Lancement et connexion d'une instance Amazon RDS externe dans un VPC par défaut

Les procédures suivantes décrivent le processus de connexion d'une instance Amazon RDS externe à un VPC par défaut. Le processus est le même si vous utilisez un VPC personnalisé. à l'exception des obligations supplémentaires suivantes : votre environnement et l'instance DB doivent se situer dans le même sous-réseau ou dans des sous-réseaux qui sont autorisés à communiquer entre eux. Pour plus d'informations sur la configuration d'un VPC personnalisé à utiliser avec Elastic Beanstalk, consultez Utilisation d'Elastic Beanstalk avec Amazon VPC.

> **Note**
>
> - Une alternative au lancement d'une nouvelle instance de base de données consiste à commencer par une base de données précédemment créée par Elastic Beanstalk, puis découplée d'un environnement Beanstalk. Pour de plus amples informations, veuillez consulter the section called “Base de données”. Si vous choisissez cette option, vous ne devez pas terminer la procédure de lancement d'une nouvelle base de données. Toutefois, vous devez effectuer les procédures suivantes décrites au sein de cette rubrique.
> - Si vous commencez avec une base de données créée par Elastic Beanstalk et découplée ensuite d'un environnement Beanstalk, vous pouvez ignorer les premières étapes et poursuivre avec les étapes regroupées sous Pour modifier les règles de trafic entrant sur le groupe de sécurité de votre instance RDS.
> - Si vous envisagez d'utiliser la base de données que vous découplez pour un environnement de production, vérifiez que le type de stockage utilisé par la base de données est adapté à votre charge de travail. Pour plus d'informations, consultez Stockage d'instance de base de données et Modification d'une instance de base de données dans le Guide de l'utilisateur Amazon RDS.

Pour lancer une instance DB RDS dans un VPC par défaut

1. Ouvrez la console RDS.
2. Dans le panneau de navigation, choisissez Databases (Bases de données).
3. Choisissez Create database (Créer une base de données).

4. Choisissez Standard Create (Création standard).

> **Important**
>
> Ne choisissez pas Easy Create (Création facile). Si vous choisissez cette option, vous ne pouvez pas configurer les paramètres nécessaires pour lancer cette base de données RDS.

5. Sous Additional Configuration (Configuration supplémentaire), pour Initial database name (Nom initial de la base de données), tapez **ebdb**.
6. Vérifiez les paramètres par défaut et ajustez ces paramètres en fonction de vos exigences spécifiques. Prêtez attention aux options suivantes :
   - DB instance class (Classe d'instance de base de données) : choisissez une taille d'instance avec un niveau approprié de puissance d'UC et de mémoire pour votre charge de travail.
   - Déploiement multi-AZ : pour une haute disponibilité, définissez ce paramètre sur Créer un Replica/Reader nœud Aurora dans une autre zone AZ.
   - Master username (Identifiant principal) et Master password (Mot de passe principal) : nom d'utilisateur et mot de passe de la base de données. Notez les valeurs de ces paramètres, car vous en aurez besoin par la suite.
7. Vérifiez les paramètres par défaut pour les autres options, puis cliquez sur Create database (Créer une base de données).

Ensuite, modifiez le groupe de sécurité attaché à votre instance de base de données pour autoriser le trafic entrant sur le port approprié. Il s'agit du même groupe de sécurité que celui que vous attacherez plus tard à votre environnement Elastic Beanstalk. Par conséquent, la règle que vous ajoutez accordera une autorisation d'accès entrant aux autres ressources du même groupe de sécurité.

Pour modifier les règles de trafic entrant sur le groupe de sécurité associé à votre instance RDS

1. Ouvrez la console Amazon RDS.
2. Choisissez Databases (Bases de données).
3. Choisissez le nom de votre instance de base de données pour en afficher les détails.
4. Dans la section Connectivity (Connectivité), prenez note des Subnets (Sous-réseaux), des Security groups (Groupes de sécurité) et du Endpoint (Point de terminaison) affichés sur cette page. Ainsi, vous pourrez utiliser ces informations ultérieurement.

5. Sous Security (Sécurité), vous voyez le groupe de sécurité associé à l'instance de base de données. Ouvrez le lien pour afficher le groupe de sécurité dans la EC2 console Amazon.
6. Dans les détails du groupe de sécurité, choisissez l'onglet Inbound (Entrant).
7. Choisissez Edit (Modifier).
8. Choisissez Add Rule (Ajouter une règle).
9. Pour Type, choisissez le moteur de base de données utilisé par votre application.
10. Pour Source, entrez **`sg-`** pour afficher la liste des groupes de sécurité disponibles. Choisissez le groupe de sécurité associé au groupe Auto Scaling utilisé avec votre environnement Elastic Beanstalk. Cela permet aux EC2 instances Amazon présentes dans l'environnement d'accéder à la base de données.

11. Choisissez Enregistrer.

Ensuite, ajoutez le groupe de sécurité de l'instance de base de données à votre environnement en cours d'exécution. Dans cette procédure, Elastic Beanstalk réapprovisionne toutes les instances de votre environnement avec le groupe de sécurité supplémentaire attaché.

Pour ajouter un groupe de sécurité à votre environnement

- Effectuez l'une des actions suivantes :
  - Pour ajouter un groupe de sécurité à l'aide de la console Elastic Beanstalk
    a. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
    b. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
    c. Dans le panneau de navigation, choisissez Configuration.

d. Dans la catégorie de configuration Instances, choisissez Edit (Modifier).

e. Sous groupes EC2 de sécurité, choisissez le groupe de sécurité à associer aux instances, en plus du groupe de sécurité d'instance créé par Elastic Beanstalk.

f. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

g. Lisez l'avertissement, puis choisissez Confirm (Confirmer).

- Pour ajouter un groupe de sécurité à l'aide d'un fichier de configuration, utilisez l'exemple de fichier `securitygroup-addexisting.config`.

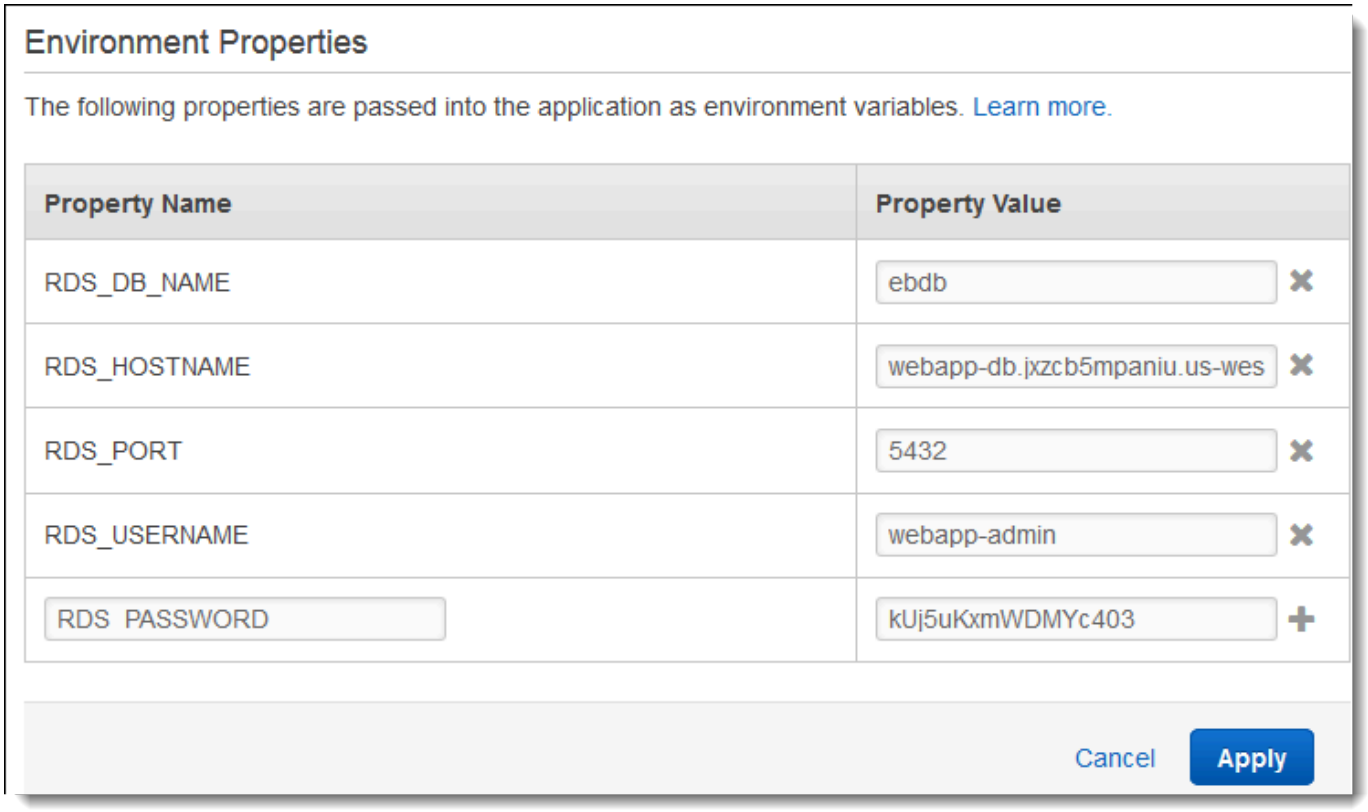
Ensuite, transmettez les informations de connexion à votre environnement à l'aide des propriétés de l'environnement. Lorsque vous ajoutez une instance de base de données à votre environnement avec la console Elastic Beanstalk, Elastic Beanstalk utilise des propriétés d'environnement comme RDS_HOSTNAME pour transmettre les informations de connexion à votre application. Vous pouvez utiliser les mêmes propriétés. De cette façon, vous utilisez le même code d'application avec les instances de base de données intégrées et les instances de base de données externes. Vous pouvez également choisir vos propres noms de propriété.

Pour configurer les propriétés d'environnement pour une instance de base de données Amazon RDS

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Dans la section Environment properties (Propriétés de l'environnement), définissez les variables lues par votre application pour créer une chaîne de connexion. Pour assurer la compatibilité avec les environnements disposant d'une instance DB RDS intégrée, utilisez les noms et valeurs suivants : Vous pouvez trouver toutes les valeurs, à l'exception de votre mot de passe, dans la console RDS.

Nom de la propriété	Description	Valeur de la propriété
`RDS_HOSTNAME`	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité

Nom de la propriété	Description	Valeur de la propriété
		et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instanc e de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, **ebdb**.	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.



6. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Si vous n'avez pas programmé votre application pour lire les propriétés d'environnement et créer une chaîne de connexion, consultez l'une des rubriques suivantes afin d'obtenir des instructions pour le langage de votre choix :

- Java SE – Connexion à une base de données (plateformes Java SE)
- Java avec Tomcat – Connexion à une base de données (plateformes Tomcat)
- Node.js – Connexion à une base de données
- .NET – Connexion à une base de données
- PHP – Connexion à une base de données à l'aide d'un PDO ou d'un My SQLi
- Python – Connexion à une base de données
- Ruby – Connexion à une base de données

Enfin, selon le moment auquel votre application lit les variables d'environnement, vous pouvez être amené à redémarrer le serveur d'application sur les instances de votre environnement.

Pour redémarrer les serveurs d'applications de votre environnement

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Choisissez Actions (Actions), puis Restart App Server(s) (Redémarrer les serveurs d'application).

Stockage des informations d'identification Amazon RDS dans AWS Secrets Manager

Cette rubrique explique AWS Secrets Manager comment améliorer votre niveau de sécurité pour la récupération des informations d'identification avec Elastic Beanstalk. Il fournit également des références à des ressources spécifiques qui peuvent vous aider à configurer les informations d'identification de votre application Elastic Beanstalk.

AWS Secrets Manager vous aide à améliorer votre niveau de sécurité en vous permettant de stocker et de récupérer des informations d'identification chiffrées. Le stockage des informations d'identification dans Secrets Manager permet d'éviter toute compromission possible par quiconque peut inspecter votre application ou les composants qui y sont liés. Votre code peut appeler le service Secrets Manager au moment de l'exécution pour récupérer les informations d'identification de manière dynamique. Secrets Manager propose également des fonctionnalités telles que des composants de mise en cache des secrets côté client pour les langages d'exécution, notamment Python, Go et Java.

Pour plus d'informations, consultez les rubriques suivantes dans le Guide de l'utilisateur AWS Secrets Manager .

- Comment Amazon RDS utilise AWS Secrets Manager
- Création d'un secret AWS Secrets Manager de base de données
- Récupérez les secrets de AWS Secrets Manager

Nettoyage d'une instance Amazon RDS externe

Lorsque vous connectez une instance Amazon RDS externe à votre environnement Elastic Beanstalk, l'instance de base de données ne dépend pas du cycle de vie de votre

environnement et n'est par conséquent pas supprimée lorsque vous résiliez votre environnement. Afin de vous assurer que les informations personnelles que vous pouvez avoir stockées dans l'instance de base de données ne sont pas inutilement conservées, supprimez les enregistrements dont vous n'avez plus besoin. Vous pouvez également supprimer l'instance de base de données.

Utilisation d'Elastic Beanstalk avec Amazon S3

Cette rubrique explique comment Elastic Beanstalk utilise Amazon Simple Storage Service (Amazon S3) et les types d'objets qu'il stocke dans des compartiments S3. Il indique également quels objets vous devez supprimer manuellement après avoir mis fin à votre environnement Elastic Beanstalk et fournit des instructions pour ce faire.

Le compartiment de comptes client Amazon S3 d'Elastic Beanstalk

Elastic Beanstalk crée un compartiment Amazon `elasticbeanstalk-`*`region`*`-`*`account-id` S3 chiffré nommé pour chaque région dans laquelle vous créez des environnements. Ce compartiment appartient à votre AWS compte. Elastic Beanstalk stocke les fichiers de configuration temporaires et d'autres objets nécessaires au bon fonctionnement de votre application dans ce compartiment. Elastic Beanstalk ACLs doit être activé pour les buckets gérés par des services et active donc la liste de contrôle d'accès (ACL) de ce bucket.

Sachez qu'Amazon S3 désactive les listes de contrôle d'accès aux compartiments (ACLs) par défaut. En outre, la rubrique de présentation des ACL du guide de l'utilisateur Amazon S3 recommande de rester ACLs désactivée, sauf dans des cas d'utilisation spécifiques. Les buckets gérés par le service Elastic Beanstalk entrent dans un cas d'utilisation qui doit être activé. ACLs Pour garantir la sécurité, les déploiements d'Elastic Beanstalk font en sorte que ce bucket soit la propriété du compte exécutant l'application.

Elastic Beanstalk conserve le chiffrement par défaut fourni par les buckets Amazon S3. Pour plus d'informations sur le chiffrement des compartiments, consultez le chiffrement par défaut d'Amazon S3 dans le guide de l'utilisateur d'Amazon Simple Storage Service.

Contenu du compartiment de comptes client Elastic Beanstalk Amazon S3

Le tableau suivant répertorie certains objets qu'Elastic Beanstalk stocke dans le compartiment de votre compte client. Le tableau indique également les objets qui doivent être supprimés manuellement. Pour éviter des frais de stockage inutiles et pour vous assurer qu'aucune information personnelle n'est conservée, veillez à supprimer manuellement ces objets lorsque vous n'en avez plus besoin.

Objet	Stocké à quel moment ?	Supprimé à quel moment ?
Versions de l'application	Lorsque vous créez un environnement ou déployez votre code d'application dans un environnement existant, Elastic Beanstalk stocke une version de l'application dans Amazon S3 et l'associe à l'environnement.	Pendant la suppression de l'application, et conformément au Cycle de vie des versions.
Bundles source	Lorsque vous chargez une nouvelle version de l'application à l'aide de la console Elastic Beanstalk ou de l'interface de ligne de commande (CLI) EB, Elastic Beanstalk en stocke une copie dans Amazon S3, et la définit comme le bundle de fichiers source de votre environnement.	Manuellement. Lorsque vous supprimez une version d'application, vous pouvez choisir Delete versions from Amazon S3 (Supprimer les versions d'Amazon S3) pour supprimer également le bundle source connexe. Pour en savoir plus, consultez Gestion des versions d'application.
Plateformes personnalisées	Lorsque vous créez une plateforme personnalisée, Elastic Beanstalk stocke temporairement les données associées dans Amazon S3.	Une fois que vous avez terminé la création de la plateforme personnalisée.
Les fichiers journaux	Vous pouvez demander à Elastic Beanstalk de récupérer les fichiers journaux d'instance (journaux de processus ou de groupe) et de les stocker dans Amazon S3. Vous pouvez également activer la rotation des journaux et configurer votre environnement de sorte à publier automatiquement les journaux dans Amazon S3 après leur rotation.	Journaux de processus et de groupe : 15 minutes après leur création. Journaux ayant subi une rotation : Manuellement.
Configurations enregistrées	Manuellement.	Manuellement.

Suppression d'objets dans le compartiment Elastic Beanstalk Amazon S3

Lorsque vous arrêtez un environnement ou supprimez une application, Elastic Beanstalk supprime la plupart des objets connexes d'Amazon S3. Pour réduire les coûts de stockage d'une application en cours d'exécution, supprimez systématiquement les objets dont votre application n'a pas besoin. En outre, prêtez une attention particulière aux objets que vous devez supprimer manuellement, comme indiqué dans Contenu du compartiment de comptes client Elastic Beanstalk Amazon S3. Pour être sûr que des informations privées ne sont pas inutilement conservées, supprimez ces objets lorsque vous n'en avez plus besoin.

- Supprimez les versions d'application que vous ne pensez plus utiliser dans votre application. Lorsque vous supprimez une version de l'application, vous pouvez sélectionner Delete versions from Amazon S3 (Supprimer les versions d'Amazon S3) pour supprimer également le bundle de fichiers source connexe. Il s'agit d'une copie du code source et des fichiers de configuration de votre application, chargée par Elastic Beanstalk sur Amazon S3 lorsque vous avez déployé une application ou chargé une version de l'application. Pour savoir comment supprimer une version d'application, consultez Gestion des versions d'application.
- Supprimez les journaux ayant fait l'objet d'une rotation, dont vous n'avez pas besoin. Vous pouvez également les télécharger ou les déplacer vers Amazon Glacier pour une analyse plus approfondie.
- Supprimez les configurations enregistrées que vous n'allez plus utiliser dans vos environnements.

Suppression du compartiment Elastic Beanstalk Amazon S3

Lorsqu'Elastic Beanstalk crée un compartiment, il crée également une stratégie de compartiment qu'il applique au nouveau compartiment. Cette politique sert deux objectifs :

- Permettre aux environnements d'écrire dans le compartiment.
- Empêcher la suppression accidentelle du compartiment.

En raison de la stratégie qu'Elastic Beanstalk applique aux compartiments qu'il crée pour vos environnements, vous n'êtes pas autorisé à les supprimer, à moins que vous ne supprimiez délibérément la politique de compartiment au préalable. Vous pouvez supprimer la stratégie de compartiment de la section Autorisations des propriétés de compartiment dans la console Amazon S3.

> **Warning**
>
> Nous vous recommandons de supprimer certains objets inutiles de votre compartiment Amazon S3 Elastic Beanstalk, au lieu de supprimer le compartiment dans son intégralité. Si vous supprimez un compartiment créé par Elastic Beanstalk dans votre compte, et que vous avez toujours des applications existantes et des environnements en cours d'exécution dans la région correspondante, vos applications risquent de ne plus fonctionner correctement. Par exemple :
>
> - Lorsqu'un environnement évolue, Elastic Beanstalk doit être en mesure de trouver la version de l'application de l'environnement dans le compartiment Amazon S3 et de l'utiliser pour démarrer de nouvelles instances Amazon. EC2
> - Lorsque vous créez une plateforme personnalisée, Elastic Beanstalk utilise le stockage Amazon S3 temporaire au cours du processus de création.
>
> Pour plus d'informations sur les implications de la suppression d'un compartiment S3, consultez les considérations répertoriées dans la section Supprimer un compartiment du guide de l'utilisateur Amazon S3.

Pour supprimer un compartiment de stockage Elastic Beanstalk (console)

La procédure générale de suppression d'un compartiment S3 est également décrite dans la section Suppression d'un compartiment du guide de l'utilisateur Amazon S3. Étant donné que nous supprimons un compartiment créé par Elastic Beanstalk dans la procédure suivante, nous incluons des étapes supplémentaires pour supprimer d'abord la stratégie du compartiment.

1. Ouvrez la console Amazon S3.
2. Ouvrez la page du compartiment de stockage Elastic Beanstalk en choisissant le nom du compartiment.
3. Sélectionnez l’onglet Autorisations.
4. Choisissez Stratégie de compartiment.
5. Sélectionnez Delete (Supprimer).
6. Revenez à la page principale de la console Amazon S3, puis sélectionnez le compartiment de stockage Elastic Beanstalk.
7. Choisissez Supprimer le compartiment.

8. Confirmez que vous voulez supprimer le compartiment en saisissant le nom de ce dernier dans le champ de texte, puis choisissez Supprimer le compartiment.

Utilisation d'Elastic AWS Secrets Manager AWS Systems Manager Beanstalk avec et Parameter Store

Cette rubrique explique comment utiliser AWS Systems Manager Parameter Store avec votre environnement Elastic Beanstalk pour stocker AWS Secrets Manager et récupérer en toute sécurité des informations sensibles, telles que les informations d'identification et les clés d'API. Votre application peut récupérer les secrets et paramètres stockés directement à partir de ces magasins, à l' APIs aide des outils de ligne de commande de ces services.

Elastic Beanstalk permet également de référencer les données de Secrets Manager et de Systems Manager Parameter Store dans des variables d'environnement. Il s'agit d'une option sécurisée permettant à votre application d'accéder de manière native aux secrets et aux paramètres stockés par ces services sans avoir à gérer les appels d'API qui leur sont adressés.

Rubriques

- Extraction de secrets et de paramètres dans les variables d'environnement Elastic Beanstalk
- Autorisations IAM requises pour qu'Elastic Beanstalk puisse accéder aux secrets et aux paramètres
- Utilisation AWS Secrets Manager et magasin AWS Systems Manager de paramètres
- Résolution des problèmes liés à l'intégration avec les variables d'environnement Elastic Beanstalk

Extraction de secrets et de paramètres dans les variables d'environnement Elastic Beanstalk

Elastic Beanstalk peut récupérer des AWS Secrets Manager valeurs AWS Systems Manager depuis et Parameter Store lors du démarrage de l'instance et les affecter à des variables d'environnement pour que votre application puisse les utiliser.

Les points suivants résument la configuration, la synchronisation et l'accès pour utiliser les variables d'environnement comme secrets :

- Configurez vos variables d'environnement pour stocker des secrets en spécifiant les Amazon Resource Names (ARNs) pour les secrets et les paramètres qu'elles stockeront.

- Lorsque des valeurs secrètes sont mises à jour ou pivotées dans Secrets Manager ou Systems Manager Parameter Store, vous devez actualiser manuellement vos variables d'environnement.
- Les variables d'environnement secrets sont disponibles pour les commandes du conteneur ebextension et les hooks de plateforme.

Versions de plateforme prises en charge

Les versions de plate-forme publiées le 26 mars 2025 ou après cette date prennent en charge AWS Secrets Manager les secrets et les AWS Systems Manager paramètres du magasin de paramètres configurés en tant que variables d'environnement.

> **Note**
>
> À l'exception des plateformes Docker et ECS, les versions de la plateforme Amazon Linux 2 ne prennent pas en charge les valeurs de variables multilignes. Pour plus d'informations sur la prise en charge des variables multilignes, consultezValeurs multilignes.

Rubriques

- Tarification
- Configurer les secrets en tant que variables d'environnement Elastic Beanstalk
- Extraction de clés JSON à partir des secrets de Secrets Manager
- Bonnes pratiques pour la synchronisation des secrets avec les variables d'environnement Elastic Beanstalk
- Valeurs multilignes dans les variables d'environnement Amazon Linux 2

Tarification

Les frais standard s'appliquent pour l'utilisation de Secrets Manager et de Systems Manager Parameter Store. Pour plus d'informations sur les tarifs, consultez les sites Web suivants :

- AWS Secrets Manager tarification
- AWS Systems Manager tarification (sélectionnez Parameter Store dans la liste des contenus)

Elastic Beanstalk ne facture aucun frais pour que votre application fasse référence à des secrets d'environnement via des variables d'environnement. Toutefois, des frais standard s'appliquent aux demandes qu'Elastic Beanstalk adresse à ces services en votre nom.

Configurer les secrets en tant que variables d'environnement Elastic Beanstalk

Vous pouvez utiliser la console Elastic Beanstalk, les fichiers de configuration qu'elle contient, AWS la CLI et le SDK pour configurer les secrets AWS et les paramètres `.ebextensions` en tant que variables d'environnement.

Rubriques

- Conditions préalables
- Utilisation de la console
- Configuration à l'aide de fichiers dans .ebextensions
- Configuration à l'aide du AWS CLI
- Configuration à l'aide du AWS SDK

Conditions préalables

Avant de configurer vos variables d'environnement pour référencer des secrets, vous devez d'abord suivre les étapes suivantes.

Procédure générale préalable à la configuration des variables d'environnement

1. Créez les secrets du Secrets Manager ou les paramètres du Parameter Store pour stocker vos données sensibles. Pour plus d'informations, consultez l'une des rubriques suivantes ou les deux :
   - Création de secrets dans the section called “Utilisation de Secrets Manager”
   - Création de paramètres dans the section called “Utilisation du Systems Manager Parameter Store”
2. Configurez les autorisations IAM requises pour les instances EC2 de votre environnement afin de récupérer les secrets et les paramètres. Pour de plus amples informations, veuillez consulter Autorisations IAM requises.

Utilisation de la console

Vous pouvez utiliser la console Elastic Beanstalk pour configurer les secrets en tant que variables d'environnement.

Pour configurer les secrets en tant que variables d'environnement dans la console Elastic Beanstalk

1. Ouvrez la console Elastic Beanstalk, puis dans la liste des régions, sélectionnez votre. Région AWS
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Mises à jour, surveillance et journalisation, sélectionnez Modifier.
5. Faites défiler la page jusqu'à Variables d'environnement d'exécution.
6. Sélectionnez Ajouter une variable d'environnement.
7. Pour Source, sélectionnez Secrets Manager ou SSM Parameter Store.

   > **Note**
   >
   > Pour plus d'informations sur l'option Texte brut de la liste déroulante, consultezConfiguration des propriétés de l'environnement (variables d'environnement).

8. Dans Nom de la variable d'environnement, entrez le nom de la variable d'environnement qui doit contenir le secret ou la valeur du paramètre.
9. Pour la valeur de la variable d'environnement, entrez l'ARN du paramètre Systems Manager Parameter Store ou le secret Secrets Manager. Pendant le démarrage de l'instance, Elastic Beanstalk initie la valeur de la variable que vous avez saisie à l'étape 8 à la valeur stockée dans cette ressource ARN.

   La console vérifie si la valeur que vous entrez est un format ARN valide pour le magasin que vous avez sélectionné à l'étape 7. Cependant, cela ne valide pas l'existence de la ressource spécifiée par l'ARN ni si vous disposez des autorisations IAM requises pour y accéder.
10. Si vous devez ajouter d'autres variables, répétez les étapes 6 à 9.
11. Pour enregistrer les modifications, cliquez sur Appliquer en bas de la page.

Configuration à l'aide de fichiers dans .ebextensions

Vous pouvez utiliser les fichiers de configuration d'Elastic Beanstalk pour configurer les secrets en tant que variables d'environnement. Utilisez l'espace de patte : tige de haricot élastique : application : secrets environnementaux noms pour définir les propriétés de l'environnement.

> **Note**
>
> Secrets Manager ajoute automatiquement 6 caractères aléatoires aux noms des secrets au format ARN pour garantir leur unicité.

Example.ebextensions/options.config pour les secrets d'environnement (syntaxe abrégée)

```
option_settings:
  aws:elasticbeanstalk:application:environmentsecrets:
    MY_SECRET: arn:aws:secretsmanager:us-east-1:111122223333:secret:mysecret-AbCd12
    MY_PARAMETER: arn:aws:ssm:us-east-1:111122223333:parameter/myparam
```

Example.ebextensions/options.config pour les secrets d'environnement (syntaxe standard)

```
option_settings:
  - namespace: aws:elasticbeanstalk:application:environmentsecrets
    option_name: MY_SECRET
    value: arn:aws:secretsmanager:us-east-1:111122223333:secret:mysecret-AbCd12
  - namespace: aws:elasticbeanstalk:application:environmentsecrets
    option_name: MY_PARAMETER
    value: arn:aws:ssm:us-east-1:111122223333:parameter/myparam
```

Configuration à l'aide du AWS CLI

Vous pouvez utiliser l'interface de ligne de AWS commande (AWS CLI) pour configurer les secrets sous forme de variables d'environnement Elastic Beanstalk. Cette section fournit des exemples de commandes create-environment et update-environment avec l'espace de noms. patte : tige de haricot élastique : application : secrets environnementaux Lorsqu'Elastic Beanstalk démarre les instances EC2 pour les environnements auxquels ces commandes font référence, il initialise les variables d'environnement avec le secret extrait et les valeurs des paramètres. Il récupère ces valeurs dans le magasin ARNs de paramètres Secrets Manager et Systems Manager respectivement.

Les deux exemples suivants utilisent la commande create-environment pour ajouter un secret et un paramètre, configurés sous forme de variables d'environnement nommées `MY_SECRET` et. `MY_PARAMETER`

Example de create-environment avec des secrets configurés en tant que variables d'environnement (options d'espace de noms en ligne)

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--platform-arn "arn:aws:elasticbeanstalk:::platform/Node.js 24 running on 64bit Amazon
 Linux 2023" \
--option-settings \
Namespace=aws:autoscaling:launchconfiguration,OptionName=IamInstanceProfile,Value=aws-
elasticbeanstalk-ec2-role \
Namespace=aws:elasticbeanstalk:application:environmentsecrets,OptionName=MY_SECRET,Value=arn:aw
east-1:111122223333:secret:mysecret-AbCd12 \
Namespace=aws:elasticbeanstalk:application:environmentsecrets,OptionName=MY_PARAMETER,Value=arn
east-1:111122223333:parameter/myparam
```

À titre de solution de rechange, utilisez un fichier `options.json` pour spécifier les options de l'espace de noms au lieu de les inclure en ligne.

Example de create-environment avec des secrets configurés en tant que variables d'environnement (options d'espace de noms dans le fichier) **`options.json`**

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--platform-arn "arn:aws:elasticbeanstalk:::platform/Node.js 24 running on 64bit Amazon
 Linux 2023" \
--option-settings file://options.json
```

Example

```
### example options.json ###
[
  {
```

```
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "IamInstanceProfile",
    "Value": "aws-elasticbeanstalk-ec2-role"
  },
  {
    "Namespace": "aws:elasticbeanstalk:application:environmentsecrets",
    "OptionName": "MY_SECRET",
    "Value": "arn:aws:secretsmanager:us-east-1:111122223333:secret:mysecret-AbCd12"
  },
  {
    "Namespace": "aws:elasticbeanstalk:application:environmentsecrets",
    "OptionName": "MY_PARAMETER",
    "Value": "arn:aws:ssm:us-east-1:111122223333:parameter/myparam"
  }
]
```

L'exemple suivant configure des variables d'environnement, nommées `MY_SECRET` et`MY_PARAMETER`, pour stocker un secret et un paramètre pour un environnement existant. La commande update-environment transmet les options avec la même syntaxe que la `create-environment` commande, soit en ligne, soit avec un fichier. `options.json` L'exemple suivant illustre la commande utilisant le même `options.json` fichier que celui utilisé dans l'exemple précédent.

Example de l'environnement de mise à jour avec des secrets configurés en tant que variables d'environnement (options d'espace de noms dans le fichier) **`options.json`**

```
aws elasticbeanstalk update-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--platform-arn "arn:aws:elasticbeanstalk:::platform/Node.js 24 running on 64bit Amazon
 Linux 2023" \
--option-settings file://options.json
```

Configuration à l'aide du AWS SDK

Vous pouvez configurer les secrets et les paramètres en tant que variables d'environnement à l'aide du AWS SDKs. Comme pour les `create-environment` AWS CLI commandes `update-environment` et mentionnées dans la section précédente, vous pouvez utiliser les actions de

l'UpdateEnvironmentAPI CreateEnvironmentet. Utilisez le paramètre de `OptionSettings` requête pour spécifier les options de l'espace de patte : tige de haricot élastique : application : secrets environnementaux noms.

Extraction de clés JSON à partir des secrets de Secrets Manager

Les versions de plateforme publiées le 13 janvier 2026 ou après cette date permettent d'extraire des champs spécifiques des secrets de Secrets Manager au format JSON en ajoutant deux points et le nom de la clé JSON à l'ARN secret. Cela vous permet de référencer des paires clé-valeur individuelles au sein d'un secret plutôt que le secret entier.

Syntaxe

Pour extraire une clé JSON spécifique d'un secret, ajoutez `:json-key-name` à l'ARN du secret :

```
arn:aws:secretsmanager:region:account:secret:secret-name-XXXXXX:json-key-name
```

L'extraction de clés JSON peut être configurée en utilisant les mêmes méthodes que les secrets d'environnement classiques : console, fichiers de configuration dans `.ebextensions`, AWS CLI ou AWS SDKs.

Limitations

- L'extraction de clés JSON n'est prise en charge que pour les secrets de Secrets Manager, et non pour les paramètres du magasin de paramètres de Systems Manager.
- Seules les clés JSON de haut niveau sont prises en charge. L'accès par clé imbriquée (par exemple,`config.database.host`) et l'indexation par tableau (par exemple,`servers[0]`) ne sont pas pris en charge. Si vous devez accéder à des valeurs imbriquées, extrayez l'objet parent et analysez-le dans le code de votre application. Les objets imbriqués et les tableaux auxquels on accède à l'aide de la clé de niveau supérieur sont sérialisés au format JSON.
- Les noms de clé JSON ne peuvent pas contenir de caractères deux-points (`:`). Les deux-points sont réservés comme délimiteurs dans la syntaxe de l'ARN. Si le nom d'une clé JSON contient deux points, seule la partie située avant le premier point est utilisée comme nom de clé, et tous les caractères situés après les deux points sont ignorés.
- La plate-forme Docker gérée par ECS utilise la syntaxe native ECS pour référencer les secrets. Pour plus d'informations, consultez la section Transmettre les secrets de Secrets Manager via les variables d'environnement Amazon ECS dans le manuel Amazon Elastic Container Service Developer Guide.

Exemple

L'exemple suivant montre comment différents types de valeurs sont extraits d'un secret JSON. Compte tenu de ce secret :

```
{
  "stringKey": "value1",
  "numberKey": 42,
  "objectKey": {
    "nested": "data"
  },
  "arrayKey": ["item1", "item2"]
}
```

Vous pouvez configurer l'extraction dans `.ebextensions` :

Example Exemple de .ebextensions avec extraction de clés JSON

```
option_settings:
  aws:elasticbeanstalk:application:environmentsecrets:
    STRING_VAR: arn:aws:secretsmanager:us-east-1:111122223333:secret:mysecret-
AbCd12:stringKey
    NUMBER_VAR: arn:aws:secretsmanager:us-east-1:111122223333:secret:mysecret-
AbCd12:numberKey
    OBJECT_VAR: arn:aws:secretsmanager:us-east-1:111122223333:secret:mysecret-
AbCd12:objectKey
    ARRAY_VAR: arn:aws:secretsmanager:us-east-1:111122223333:secret:mysecret-
AbCd12:arrayKey
```

Cela se traduit par les variables d'environnement suivantes :

- `STRING_VAR=value1`
- `NUMBER_VAR=42`
- `OBJECT_VAR={"nested":"data"}`
- `ARRAY_VAR=["item1","item2"]`

Bonnes pratiques pour la synchronisation des secrets avec les variables d'environnement Elastic Beanstalk

Cette rubrique recommande les meilleures pratiques pour que votre application utilise les secrets d'environnement avec Secrets Manager ou le magasin de paramètres Systems Manager. Votre application Elastic Beanstalk ne recevra pas automatiquement les valeurs mises à jour si les données du magasin secret sont mises à jour ou pivotées. Elastic Beanstalk extrait les secrets des variables d'environnement uniquement au moment du démarrage de l'instance.

Actualisation de vos variables d'environnement

Pour que votre environnement Elastic Beanstalk récupère les dernières valeurs des secrets dans leurs magasins de secrets, nous vous recommandons d'exécuter l'opération or. `UpdateEnvironment` `RestartAppServer` Vous pouvez exécuter ces opérations à l'aide de la console Elastic Beanstalk AWS CLI, de ou de l'API Elastic Beanstalk. Pour plus d'informations, consultez les AWS CLI exemples d'Elastic Beanstalk ou l'API Reference.AWS Elastic Beanstalk

Gestion des effets de mise à l'échelle automatique sur la synchronisation secrète

Si un événement de redimensionnement ou un remplacement d'instance survient après les mises à jour de la banque secrète, la nouvelle instance qui apparaît contiendra les dernières valeurs secrètes de Secrets Manager ou de Systems Manager Parameter Store. Un tel événement peut se produire même si toutes les autres instances de l'environnement n'ont pas été actualisées pour récupérer les nouveaux secrets.

> **Important**
>
> Vous devez vous assurer que votre application est capable d'utiliser deux valeurs secrètes différentes pour la même variable d'environnement. Cela permet de gérer les événements dans lesquels une mise à jour secrète a lieu dans Secrets Manager ou Systems Manager Parameter Store, suivie d'une mise à l'échelle ou d'un remplacement d'instance dans votre environnement, tandis que les autres instances sont en attente d'une actualisation des variables d'environnement. Pendant la période d'attente pour l'actualisation, toutes les instances d'environnement n'auront pas les mêmes valeurs pour les variables d'environnement de la banque secrète.

Un exemple d'un tel cas d'utilisation est la rotation des informations d'identification de base de données. Lorsqu'un événement de scalabilité fait suite à la rotation des informations d'identification,

les secrets d'environnement référencés par les instances nouvellement démarrées contiennent les informations d'identification de base de données mises à jour. Cependant, les secrets d'environnement référencés par les instances existantes conservent l'ancienne valeur jusqu'à ce qu'ils soient actualisés par les `RestartAppServer` opérations `UpdateEnvironment` or.

Valeurs multilignes dans les variables d'environnement Amazon Linux 2

Les valeurs multilignes sont composées de plusieurs lignes et incluent un caractère de nouvelle ligne. À l'exception des plateformes Docker et Docker basées sur ECS, les plateformes qui s'exécutent sur Amazon Linux 2 ne prennent pas en charge les valeurs multilignes pour les variables d'environnement

> **Note**
>
> Elastic Beanstalk échouera au déploiement des environnements concernés s'il détecte une valeur multiligne.

Les options suivantes peuvent constituer des solutions de contournement ou des solutions au problème multiligne :

- Mettez à niveau votre environnement Amazon Linux 2 vers Amazon Linux 2023. Pour de plus amples informations, veuillez consulter Migration depuis Amazon Linux 2 vers Amazon Linux 2023.
- Supprimez les caractères de nouvelle ligne de vos valeurs secrètes. Un exemple d'approche consiste à encoder vos valeurs en Base64 avant de les stocker dans le magasin secret. Votre application devra alors décoder à nouveau la valeur dans son format d'origine lorsqu'elle la référence à partir de la variable secrète d'environnement.
- Concevez le code de votre application pour récupérer les données directement depuis Secrets Manager ou Systems Manager Parameter Store. Pour plus d'informations, voir Extraction de secrets dans Utilisation de Secrets Manager ou Extraction de paramètres. Utilisation du Systems Manager Parameter Store

Autorisations IAM requises pour qu'Elastic Beanstalk puisse accéder aux secrets et aux paramètres

Vous devez accorder les autorisations nécessaires aux instances EC2 de votre environnement pour récupérer les secrets et les paramètres de AWS Systems Manager Parameter AWS Secrets Manager Store. Les autorisations sont accordées aux instances EC2 via un rôle de profil d'instance EC2.

Les sections suivantes répertorient les autorisations spécifiques que vous devez ajouter à un profil d'instance EC2, en fonction du service que vous utilisez. Suivez les étapes indiquées dans Mettre à jour la politique d'autorisations pour un rôle dans le guide de l'utilisateur IAM pour ajouter ces autorisations.

> **Autorisations IAM pour la plateforme Docker gérée par ECS**
>
> La plate-forme Docker gérée par ECS nécessite des autorisations IAM supplémentaires à celles fournies dans cette rubrique. Pour plus d'informations sur toutes les autorisations requises pour que votre environnement de plateforme Docker géré par ECS prenne en charge l'intégration des variables d'environnement Elastic Beanstalk avec les secrets, consultez. Format ARN du rôle d'exécution

Rubriques

- Autorisations IAM requises pour Secrets Manager
- Autorisations IAM requises (Systems Manager Parameter Store)

Autorisations IAM requises pour Secrets Manager

Les autorisations suivantes permettent d'accéder à l'extraction de secrets chiffrés depuis le AWS Secrets Manager magasin :

- responsable des secrets : GetSecretValue
- kms:Decrypt

L'autorisation de déchiffrer un secret n' AWS KMS key est requise que si votre secret utilise une clé gérée par le client au lieu de la clé par défaut. L'ajout de votre clé ARN personnalisée ajoute l'autorisation de déchiffrer la clé gérée par le client.

Example politique avec Secrets Manager et autorisations clés KMS

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:secretsmanager:us-east-1:111122223333:secret:my-secret",
                "arn:aws:kms:us-east-1:111122223333:key/my-key"
            ]
        }
    ]
}
```

Autorisations IAM requises (Systems Manager Parameter Store)

Les autorisations suivantes permettent d'accéder aux paramètres chiffrés depuis le AWS Systems Manager Parameter Store :

- SMS : GetParameter
- kms:Decrypt

L'autorisation de déchiffrer un n' AWS KMS key est requise que pour les types de `SecureString` paramètres qui utilisent une clé gérée par le client au lieu d'une clé par défaut. L'ajout de votre clé ARN personnalisée ajoute l'autorisation de déchiffrer la clé gérée par le client. Les types de paramètres habituels qui ne sont pas chiffrés `String` et `StringList` qui n'ont pas besoin de AWS KMS key.

Example politique avec Systems Manager et autorisations AWS KMS clés

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameter",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-1:111122223333:parameter/my-parameter",
                "arn:aws:kms:us-east-1:111122223333:key/my-key"
            ]
        }
    ]
}
```

Utilisation AWS Secrets Manager et magasin AWS Systems Manager de paramètres

Cette rubrique fournit une brève introduction à AWS Systems Manager Parameter Store, des AWS Secrets Manager informations sur les prix et des références pour en savoir plus sur la création et la récupération de secrets, à l'aide de la console et des options de programmation.

À propos de Secrets Manager

AWS Secrets Manager vous aide à gérer, à récupérer et à alterner les secrets tout au long de leur cycle de vie. Parmi les données secrètes que vous pouvez gérer, citons les informations d'identification de base de données, les informations d'identification d'application, les OAuth jetons et les clés d'API. Secrets Manager vous permet de configurer un calendrier de rotation automatique pour vos secrets.

À propos de Systems Manager Parameter Store

Parameter Store est un outil de AWS Systems Manager. Il fournit un stockage hiérarchique sécurisé pour la gestion des données de configuration et la gestion des secrets. Vous pouvez gérer les

données de configuration importantes sous forme de valeurs de paramètres. Les exemples de données que vous pouvez gérer avec Parameter Store incluent Amazon Machine Image (AMI) IDs, les codes de licence, les mots de passe et les chaînes de base de données.

> **Tarification**
>
> Les frais standard s'appliquent pour l'utilisation de Secrets Manager et de Systems Manager Parameter Store. Pour plus d'informations sur les tarifs, consultez les sites Web suivants :
>
> - AWS Secrets Manager tarification
> - AWS Systems Manager tarification (sélectionnez Parameter Store dans la liste des contenus)

Rubriques

- Utiliser le Gestionnaire de Secrets pour créer et récupérer des secrets
- Utilisation du magasin de paramètres de Systems Manager pour créer et récupérer des paramètres

Utiliser le Gestionnaire de Secrets pour créer et récupérer des secrets

Vous pouvez créer et récupérer les secrets de Secrets Manager à l'aide de la AWS Secrets Manager console AWS CLI, du SDK ou du AWS SDK. Consultez les ressources suivantes pour en savoir plus sur les différentes méthodes de création et de récupération des secrets de Secrets Manager.

Création de secrets

- Console — Créez un AWS Secrets Manager secret (console) dans le guide de l'AWS Secrets Manager utilisateur
- AWS CLI — AWS CLI dans le guide de AWS Secrets Manager l'utilisateur
- AWS SDK — AWS SDK dans le guide de l'utilisateur AWS Secrets Manager

Retrouver des secrets

- Console — Obtenez une valeur secrète (console) dans le guide de l'AWS Secrets Manager utilisateur
- AWS CLI — Obtenir une valeur secrète (AWS CLI) dans le guide de AWS Secrets Manager l'utilisateur

- AWS SDK — Exemples de code à utiliser par Secrets Manager AWS SDKs dans la bibliothèque de codes d'exemples de code du AWS SDK
- Autres méthodes — Obtenez des secrets AWS Secrets Manager dans le guide de AWS Secrets Manager l'utilisateur

Pour plus d'informations AWS Secrets Manager, voir Qu'est-ce que c'est AWS Secrets Manager ? dans le guide de AWS Secrets Manager l'utilisateur.

Utilisation du magasin de paramètres de Systems Manager pour créer et récupérer des paramètres

Vous pouvez créer et récupérer les paramètres du Parameter Store à l'aide de la AWS Systems Manager console AWS CLI, du SDK ou du AWS SDK. Consultez les ressources suivantes pour en savoir plus sur les différentes méthodes de création et de récupération des paramètres du magasin de paramètres.

Création de paramètres

- Console — Création d'un paramètre Systems Manager (console) dans le guide de AWS Systems Manager l'utilisateur
- AWS CLI — Création d'un paramètre Systems Manager (AWS CLI) dans le guide de AWS Systems Manager l'utilisateur
- AWS SDK — À utiliser PutParameter avec un AWS SDK ou AWS CLI dans la bibliothèque de codes d'exemples de code du AWS SDK

Récupération des paramètres

- Console — Recherche d'un paramètre (console) dans le guide de l'AWS Systems Manager utilisateur
- AWS CLI — À utiliser GetParameter avec un AWS SDK ou AWS CLI dans la bibliothèque de codes d'exemples de code du AWS SDK
- AWS SDK — À utiliser GetParameter avec un AWS SDK ou AWS CLI dans la bibliothèque de codes d'exemples de code du AWS SDK

Pour plus d’informations, veuillez consulter la rubrique AWS Systems Manager Parameter Store dans le Guide de l’utilisateur AWS Systems Manager .

Résolution des problèmes liés à l'intégration avec les variables d'environnement Elastic Beanstalk

> **Essayez Amazon Q Developer CLI pour un dépannage assisté par l'IA**
>
> L'interface de ligne de commande Amazon Q Developer peut vous aider à résoudre rapidement les problèmes d'environnement. La Q CLI fournit des solutions en vérifiant l'état de l'environnement, en examinant les événements, en analysant les journaux et en posant des questions de clarification. Pour plus d'informations et des instructions détaillées, consultez la section Résolution des problèmes liés aux environnements Elastic Beanstalk avec Amazon Q Developer CLI dans les blogs. AWS

Cette section fournit des conseils pour résoudre les problèmes liés aux secrets dans votre environnement Elastic Beanstalk.

Événement : le déploiement de l'instance n'a pas réussi à obtenir un ou plusieurs secrets

Ce message indique qu'Elastic Beanstalk n'a pas pu récupérer un ou plusieurs des secrets spécifiés lors du déploiement de votre application.

- Vérifiez que les ressources spécifiées par les valeurs ARN dans la configuration de vos variables d'environnement existent.
- Vérifiez que votre rôle de profil d'instance Elastic Beanstalk EC2 dispose des autorisations IAM requises pour accéder aux ressources.
- Si cet événement a été déclenché par l'`RestartAppServer`opération, une fois le problème résolu, relancez l'`RestartAppServer`appel pour le résoudre.
- Si l'événement a été déclenché par un `UpdateEnvironment` appel, recommencez l'`UpdateEnvironment`opération.

Pour des exemples de ces commandes, consultez les AWS CLI exemples d'Elastic Beanstalk. Pour plus d'informations sur les actions d'API associées à ces opérations, consultez la référence des AWS Elastic Beanstalk API.

Événement : le déploiement de l'instance a détecté une ou plusieurs valeurs d'environnement multilignes, qui ne sont pas prises en charge pour cette plate-forme

Les variables multilignes ne sont pas prises en charge sur les plateformes Amazon Linux 2, à l'exception des plateformes Docker et Docker gérées par ECS. Pour connaître les options disponibles pour continuer, consultezValeurs multilignes.

Événement : CreateEnvironment échoue lorsqu'un secret est spécifié

En cas d'`CreateEnvironment`échec et que vous avez des secrets en tant que variables d'environnement, vous devez résoudre le problème sous-jacent, puis l'utiliser `UpdateEnvironment` pour terminer la configuration de l'environnement. Ne pas utiliser`RestartAppServer`, car cela ne suffira pas à améliorer l'environnement dans cette situation. Pour des exemples de ces commandes, consultez les AWS CLI exemples d'Elastic Beanstalk. Pour plus d'informations sur les actions d'API associées à ces opérations, consultez la référence des AWS Elastic Beanstalk API.

Utilisation d'Elastic Beanstalk avec Amazon VPC

Cette rubrique explique les avantages de l'utilisation de points de terminaison VPC avec Elastic Beanstalk et les différents types de configurations que vous pouvez implémenter.

Vous pouvez utiliser un Amazon Virtual Private Cloud (Amazon VPC) pour créer un réseau sécurisé pour votre application Elastic Beanstalk et les ressources associées. AWS Lorsque vous créez votre environnement, vous choisissez les VPC, sous-réseaux et groupes de sécurité utilisés pour vos instances d'application et votre équilibreur de charge. Vous pouvez utiliser la configuration de VPC de votre choix, à condition qu'elle respecte les conditions suivantes.

Exigences du VPC

- Accès Internet – Les instances peuvent avoir accès à Internet via l'une des méthodes suivantes.
  - Sous-réseau public – Les instances ont une adresse IP publique et utilisent une passerelle Internet pour accéder à Internet.
  - Sous-réseau privé – Les instances utilisent un périphérique NAT pour accéder à Internet.

> **Note**
>
> Si vous configurez des points de terminaison d'un VPC dans votre VPC pour vous connecter à la fois aux services `elasticbeanstalk` et `elasticbeanstalk-health`, l'accès Internet est facultatif et est uniquement requis si votre application en a spécifiquement besoin. Sans points de terminaison de VPC, votre VPC doit avoir accès à Internet.

Le VPC par défaut qu'Elastic Beanstalk met en place pour vous fournir un accès Internet.

Elastic Beanstalk ne prend pas en charge les paramètres de proxy comme `HTTPS_PROXY` pour la configuration d'un proxy web.

- NTP – Les instances dans votre environnement Elastic Beanstalk utilisent le protocole NTP (Network Time) pour synchroniser l'horloge système. Si les instances ne peuvent pas communiquer sur le port UDP 123, l'horloge peut se désynchroniser, ce qui entraîne des problèmes avec les rapports d'intégrité d'Elastic Beanstalk. Assurez-vous que vos groupes de sécurité VPC et votre réseau ACLs autorisent le trafic UDP entrant et sortant sur le port 123 pour éviter ces problèmes.

Le elastic-beanstalk-samplesréférentiel fournit des CloudFormation modèles que vous pouvez utiliser pour créer un VPC à utiliser avec vos environnements Elastic Beanstalk.

Pour créer des ressources à l'aide d'un CloudFormation modèle

1. Clonez le référentiel d'exemples ou téléchargez un modèle à l'aide des liens que vous trouvez dans le fichier README.
2. Ouvrez la CloudFormation console.
3. Sélectionnez Créer la pile.
4. Choisissez Télécharger un modèle sur Amazon S3.
5. Choisissez Charger le fichier et chargez le fichier de modèle depuis votre ordinateur local.
6. Choisissez Suivant et suivez les instructions pour créer une pile avec les ressources dans le modèle.

Lorsque la création de la pile est terminée, consultez l'onglet Sorties pour trouver l'ID et le sous-réseau du VPC. IDs Utilisez ces ressources pour configurer le VPC dans la catégorie de configuration Réseau de l'assistant de création d'un environnement.

Rubriques

- VPC public
- VPC public/privé
- VPC privé
- Exemple : Lancement d'une application Elastic Beanstalk dans un VPC avec des hôtes bastion

- Exemple : Lancement d'un Elastic Beanstalk dans un VPC avec Amazon RDS
- Utilisation d'Elastic Beanstalk avec les points de terminaison VPC
- Utilisation des stratégies de point de terminaison pour contrôler l'accès avec des points de terminaison de VPC

VPC public

CloudFormation modèle — vpc-public.yaml

Paramètres (charge équilibrée)

- Visibilité de l'équilibreur de charge – Public
- Sous-réseaux de l'équilibreur de charge – Public pour les deux
- Adresse IP publique de l'instance – Activée
- Sous-réseaux de l'instance – Public pour les deux
- Groupes de sécurité de l'instance – Ajouter le groupe de sécurité par défaut

Paramètres (une seule instance)

- Sous-réseaux de l'instance – Public pour un des deux
- Groupes de sécurité de l'instance – Ajouter le groupe de sécurité par défaut

Une conception élémentaire de VPC public uniquement inclut un ou plusieurs sous-réseaux publics, une passerelle Internet et un groupe de sécurité par défaut qui autorise le trafic entre les différentes ressources du VPC. Lorsque vous créez un environnement dans le VPC, Elastic Beanstalk crée des ressources supplémentaires qui varient en fonction du type d'environnement.

Ressources en matière de VPC

- Instance unique – Elastic Beanstalk crée un groupe de sécurité pour l'instance d'application qui autorise le trafic sur le port 80 depuis Internet, et attribue une adresse IP Elastic à l'instance pour qu'elle soit associée à une adresse IP publique. Le nom de domaine de l'environnement est résolu avec l'adresse IP publique de l'instance.
- Équilibrage de charge – Elastic Beanstalk crée un groupe de sécurité pour l'équilibreur de charge qui autorise le trafic sur le port 80 depuis Internet, et un groupe de sécurité pour les instances

d'application qui autorisent le trafic depuis le groupe de sécurité de l'équilibreur de charge. Le nom de domaine de l'environnement est résolu avec le nom de domaine public de l'équilibreur de charge.

Cette option est similaire à la façon dont Elastic Beanstalk gère la mise en réseau lorsque vous utilisez le VPC par défaut. La sécurité d'un sous-réseau public dépend des groupes de sécurité d'instance et d'équilibreur de charge créés par Elastic Beanstalk. C'est là la configuration la moins coûteuse, car aucun passerelle NAT n'est requise.

VPC public/privé

CloudFormation modèle — vpc-privatepublic.yaml

Paramètres (charge équilibrée)

- Visibilité de l'équilibreur de charge – Public
- Sous-réseaux de l'équilibreur de charge – Public pour les deux
- Adresse IP publique de l'instance – Désactivée
- Sous-réseaux de l'instance – Privé pour les deux
- Groupes de sécurité de l'instance – Ajouter le groupe de sécurité par défaut

Pour une sécurité optimale, ajoutez des sous-réseaux privés à votre VPC pour créer une configuration publique-privée. Cette configuration nécessite un équilibreur de charge et une passerelle NAT dans les sous-réseaux publics, et vous permet d'exécuter vos instances d'application, votre base de données et les autres ressources dans des sous-réseaux privés. Les instances des sous-réseaux privés peuvent uniquement communiquer avec Internet via l'équilibreur de charge et la passerelle NAT.

VPC privé

CloudFormation modèle — vpc-private.yaml

Paramètres (charge équilibrée)

- Visibilité de l'équilibreur de charge – Privé
- Sous-réseaux de l'équilibreur de charge – Les deux sous-réseaux privés

- Adresse IP publique de l'instance – Désactivée
- Sous-réseaux de l'instance – Privé pour les deux
- Groupes de sécurité de l'instance – Ajouter le groupe de sécurité par défaut

Pour les applications internes qui ne doivent pas avoir d'accès depuis Internet, vous pouvez exécuter toutes les applications dans des sous-réseaux privés et configurer l'équilibreur de charge de façon à ce qu'il soit interne (remplacez la valeur de Load balancer visibility (Visibilité de l'équilibreur de charge) par Internal (Interne). Ce modèle crée un VPC sans sous-réseaux publics et sans passerelle Internet. Utilisez cette configuration pour les applications qui doivent être accessibles à partir du même VPC ou d'un VPN associé uniquement.

Exécution d'un environnement Elastic Beanstalk dans un VPC privé

Lorsque vous créez votre environnement Elastic Beanstalk dans un VPC privé, l'environnement n'a pas accès à Internet. Votre application peut avoir besoin d'accéder au service Elastic Beanstalk ou à d'autres services. Votre environnement peut utiliser des rapports d'intégrité améliorés et, dans ce cas, les instances d'environnement envoient des informations d'intégrité au service d'intégrité amélioré. Et le code Elastic Beanstalk sur les instances d'environnement envoie du trafic AWS vers d'autres services, et d'autres types de trafic vers des sites autres que les points de terminaison (par exemple, pour télécharger des packages de dépendances pour votre application).AWS Voici quelques étapes que vous devrez peut-être emprunter dans ce cas, pour vous assurer que votre environnement fonctionne correctement.

- Configurez des points de terminaison d'un VPC pour Elastic Beanstalk – Elastic Beanstalk et son service d'intégrité amélioré prennent en charge les points de terminaison d'un VPC, ce qui garantit que le trafic vers ces services reste à l'intérieur du réseau Amazon et ne nécessite pas d'accès Internet. Pour de plus amples informations, veuillez consulter the section called “Points de terminaison d’un VPC”.
- Configurez les points de terminaison VPC pour des services supplémentaires : les instances Elastic Beanstalk envoient du trafic vers plusieurs autres services en votre AWS nom : Amazon Simple Storage Service (Amazon S3), Amazon Simple Queue Service (Amazon SQS) et Amazon Logs. AWS CloudFormation CloudWatch Vous devez également configurer les points de terminaison de VPC pour ces services. Pour de plus amples informations sur les points de terminaison VPC, y compris les liens par service, veuillez consulter les points de terminaison VPC dans le Guide de l'utilisateur Amazon VPC.

> **Note**
>
> Certains AWS services, notamment Elastic Beanstalk, prennent en charge les points de terminaison VPC dans un nombre limité de régions. AWS Lorsque vous concevez votre solution VPC privée, vérifiez qu'Elastic Beanstalk et les autres services dépendants mentionnés ici prennent en charge les points de terminaison VPC dans la région de votre choix. AWS

- Fournir une image Docker privée – Dans un environnement Docker, le code sur les instances de l'environnement peut essayer d'extraire votre image Docker configurée à partir d'Internet lors de la création de l'environnement et échouer. Pour éviter cet échec, créez une image Docker personnalisée sur votre environnement ou utilisez une image Docker stockée dans Amazon Elastic Container Registry (Amazon ECR) et configurez un point de terminaison VPC pour le service Amazon ECR.
- Activer les noms DNS : le code Elastic Beanstalk sur les instances d'environnement envoie du trafic AWS à tous les services via leurs points de terminaison publics. Pour vous assurer que ce trafic passe bien, choisissez l'option Enable DNS name (Activer le nom DNS) lorsque vous configurez tous les points de terminaison d'un VPC d'interface. Cela ajoute une entrée DNS dans votre VPC qui mappe le point de terminaison du service public au point de terminaison de VPC d'interface.

> **Important**
>
> Si votre VPC n'est pas privé et dispose d'un accès Internet public, et si Enable DNS name (Activer le nom DNS) est désactivé pour n'importe quel point de terminaison de VPC, le trafic vers le service correspondant passe par l'Internet public. Ce n'est probablement pas votre intention. Ce problème est facile à détecter avec un VPC privé, car il empêche ce trafic de passer et vous recevez des erreurs. Cependant, avec un VPC public, vous n'obtenez aucune indication.

- Inclure les dépendances de l'application – Si votre application a des dépendances telles que des packages d'exécution de langage, elle peut essayer de les télécharger et de les installer à partir d'Internet lors de la création de l'environnement et échouer. Pour éviter cet échec, incluez tous les packages de dépendance dans le bundle source de votre application.
- Utiliser une version de plateforme actuelle – Assurez-vous que votre environnement utilise une version de plateforme qui a été publiée le 24 février 2020 ou plus tard. Plus précisément, utilisez

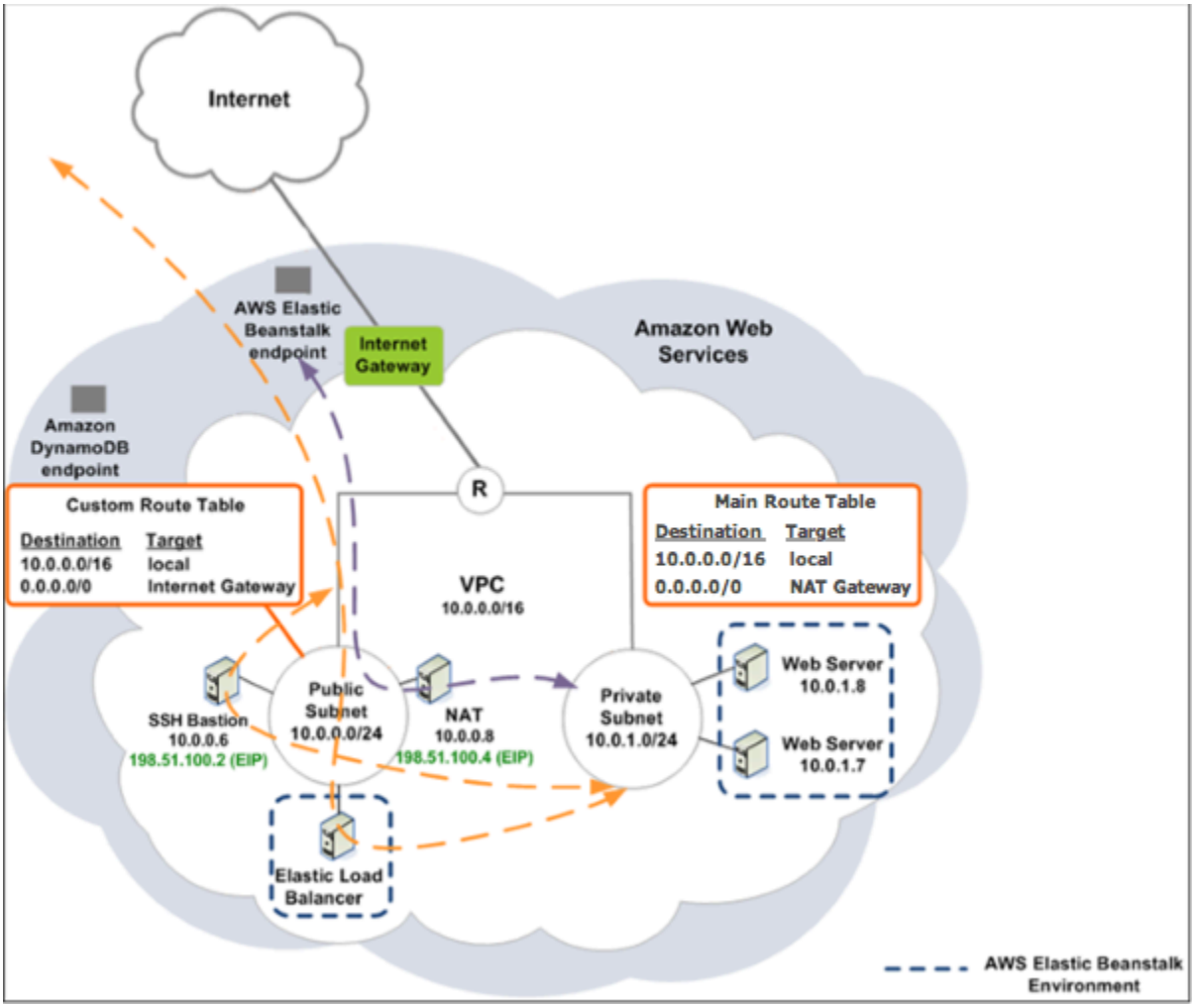
une version de plateforme qui a été publiée avec ou après l'une de ces deux mises à jour : Linux Update 2020-02-28, Windows Update 2020-02-24.

> **Note**
>
> La raison pour laquelle il vous faut une version de plateforme mise à jour est que les anciennes versions présentaient un problème qui empêcherait les entrées DNS créées par l'option Enable DNS name (Activer le nom DNS) de fonctionner correctement pour Amazon SQS.

Exemple : Lancement d'une application Elastic Beanstalk dans un VPC avec des hôtes bastion

Cette section explique comment déployer une application Elastic Beanstalk au sein d'un VPC à l'aide d'un hôte bastion et explique pourquoi vous devez implémenter cette topologie.

Si vos EC2 instances Amazon sont situées dans un sous-réseau privé, vous ne pourrez pas vous y connecter à distance. Pour vous connecter à vos instances, vous pouvez configurer des serveurs bastions dans le sous-réseau public pour servir de proxy. Par exemple, vous pouvez définir des acheminеurs de port SSH ou des passerelles RDP dans le sous-réseau public pour définir des serveurs proxy du trafic à destination de vos serveurs de base de données et en provenance de votre propre réseau. Cette section fournit un exemple sur la manière de créer d'un VPC avec un sous-réseau privé et public. Les instances sont trouvent à l'intérieur du sous-réseau privé et l'hôte bastion, la passerelle NAT ainsi que l'équilibreur de charge se trouvent à l'intérieur du sous-réseau public. Votre infrastructure ressemblera au schéma suivant.



Pour déployer une application Elastic Beanstalk à l'intérieur d'un VPC à l'aide d'un hôte bastion, suivez les étapes décrites dans les sous-sections suivantes.

Étapes

- Création d'un VPC avec des sous-réseaux public et privé
- Création et configuration du groupe de sécurité de l'hôte bastion
- Mise à jour du groupe de sécurité de l'instance
- Création d'un hôte bastion

Création d'un VPC avec des sous-réseaux public et privé

Réalisez toutes les procédures décrites dans VPC public/privé. Lors du déploiement de l'application, vous devez spécifier une paire de EC2 clés Amazon pour les instances afin de pouvoir vous y

connecter à distance. Pour plus d'informations sur la façon de spécifier la paire de clés d'instance, consultez Les EC2 instances Amazon pour votre environnement Elastic Beanstalk.

Création et configuration du groupe de sécurité de l'hôte bastion

Créez un groupe de sécurité pour l'hôte Bastion et ajoutez des règles qui autorisent le trafic SSH entrant depuis Internet et le trafic SSH sortant vers le sous-réseau privé qui contient les instances Amazon. EC2

Pour créer le groupe de sécurité de l'hôte bastion

1. Ouvrez la console Amazon VPC à l'adresse https://console.aws.amazon.com/vpc/.
2. Dans le panneau de navigation, choisissez Groupes de sécurité.
3. Sélectionnez Créer un groupe de sécurité.
4. Dans la boîte de dialogue Créer un groupe de sécurité, entrez les informations suivantes et choisissez Oui, créer.

   Balise Nom (facultatif)

   Entrez une étiquette de nom pour le groupe de sécurité.

   Nom du groupe

   Entrez le nom du groupe de sécurité.

   Description

   Entrez une description pour le groupe de sécurité.

   VPC

   Sélectionnez votre VPC.

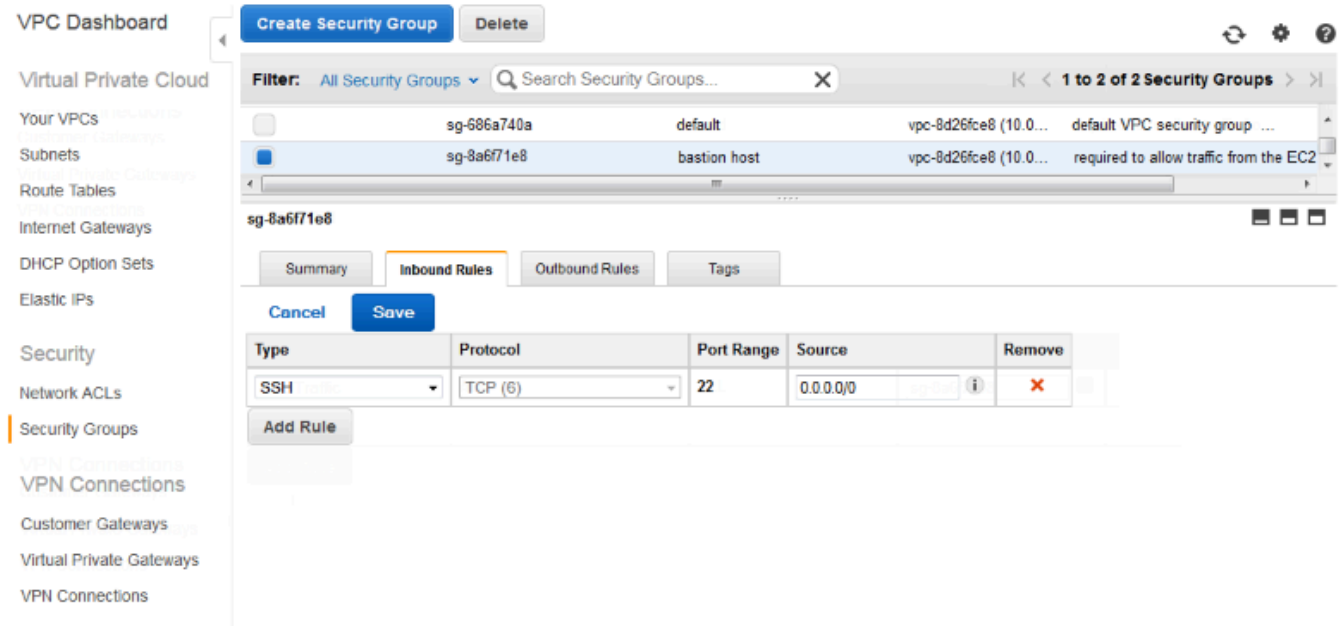
   Le groupe de sécurité est créé et apparaît sur la page Groupes de sécurité. Notez qu'il possède un ID (par exemple, `sg-xxxxxxxx`). Vous devrez peut-être activer la colonne ID du groupe en cliquant sur Afficher/Masquer dans l'angle supérieur droit de la page.

Pour configurer le groupe de sécurité de l'hôte bastion

1. Dans la liste des groupes de sécurité, cochez la case correspondant au groupe de sécurité que vous venez de créer pour votre hôte bastion.

2. Sous l'onglet Règles entrantes, choisissez Modifier.
3. Si nécessaire, sélectionnez Add another rule (Ajouter une autre règle).
4. Si votre hôte bastion est une instance Linux, sous Type, sélectionnez SSH.

   Si votre hôte bastion est une instance Windows, sous Type, sélectionnez RDP.
5. Entrez la plage CIDR source souhaitée dans le champ Source et choisissez Save (Enregistrer).

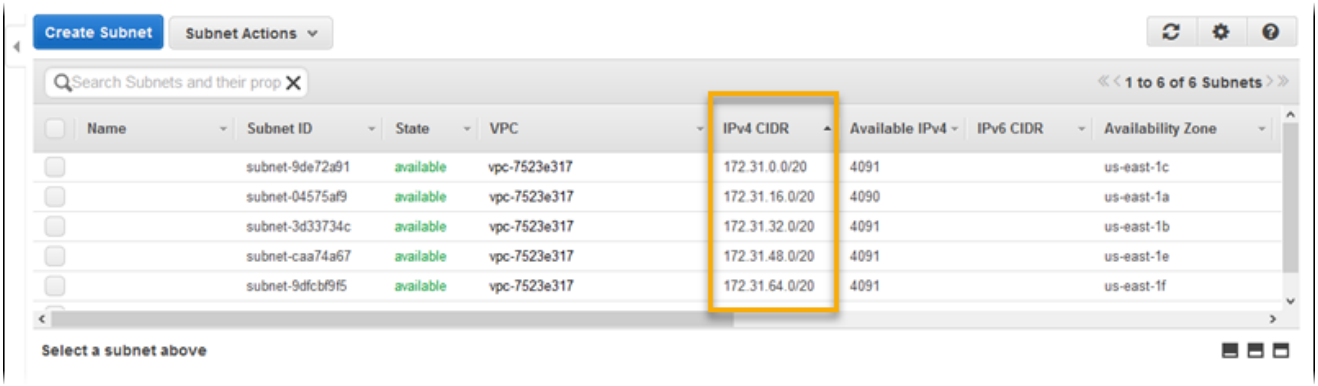


6. Sous l'onglet Outbound Rules (Règles sortantes), choisissez Edit (Modifier).
7. Si nécessaire, sélectionnez Add another rule (Ajouter une autre règle).
8. Sous Type, sélectionnez le type que vous avez spécifiée pour la règle entrante.
9. Dans le champ Source, entrez la plage d'adresses CIDR du sous-réseau des hôtes dans le sous-réseau privé du VPC.

   Pour la trouver :

   a. Ouvrez la console Amazon VPC à l'adresse https://console.aws.amazon.com/vpc/.
   b. Dans le panneau de navigation, choisissez Subnets (Sous-réseaux).
   c. Notez la valeur sous IPv4 CIDR pour chaque zone de disponibilité dans laquelle vous avez des hôtes vers lesquels vous souhaitez que l'hôte bastion établisse un pont.

> **Note**
>
> Si vous avez des hôtes dans plusieurs zones de disponibilité, créez une règle sortante pour chacun de ces zones de disponibilité.



10. Choisissez Enregistrer.

Mise à jour du groupe de sécurité de l'instance

Par défaut, le groupe de sécurité que vous avez créé pour vos instances ne permet pas le trafic entrant. Même si Elastic Beanstalk modifiera le groupe par défaut pour les instances pour autoriser le trafic SSH, vous devez modifier votre groupe de sécurité d'instance personnalisée pour autoriser le trafic RDP si vos instances sont des instances Windows.

Pour mettre à jour le groupe de sécurité de l'instance pour RDP

1. Dans la liste des groupes de sécurité, cochez la case correspondant au groupe de sécurité de l'instance.
2. Dans l'onglet Entrant, choisissez Modifier.
3. Si nécessaire, sélectionnez Add another rule (Ajouter une autre règle).
4. Entrez les valeurs suivantes, puis sélectionnez Save (Enregistrer).

   Type

   ```
   RDP
   ```

Protocole

`TCP`

Plage de ports

`3389`

Source

Entrez l'ID du groupe de sécurité de l'hôte bastion (par exemple, `sg-8a6f71e8`) et choisissez Save (Enregistrer).

Création d'un hôte bastion

Pour créer un hôte bastion, vous lancez une EC2 instance Amazon dans votre sous-réseau public qui fera office d'hôte bastion.

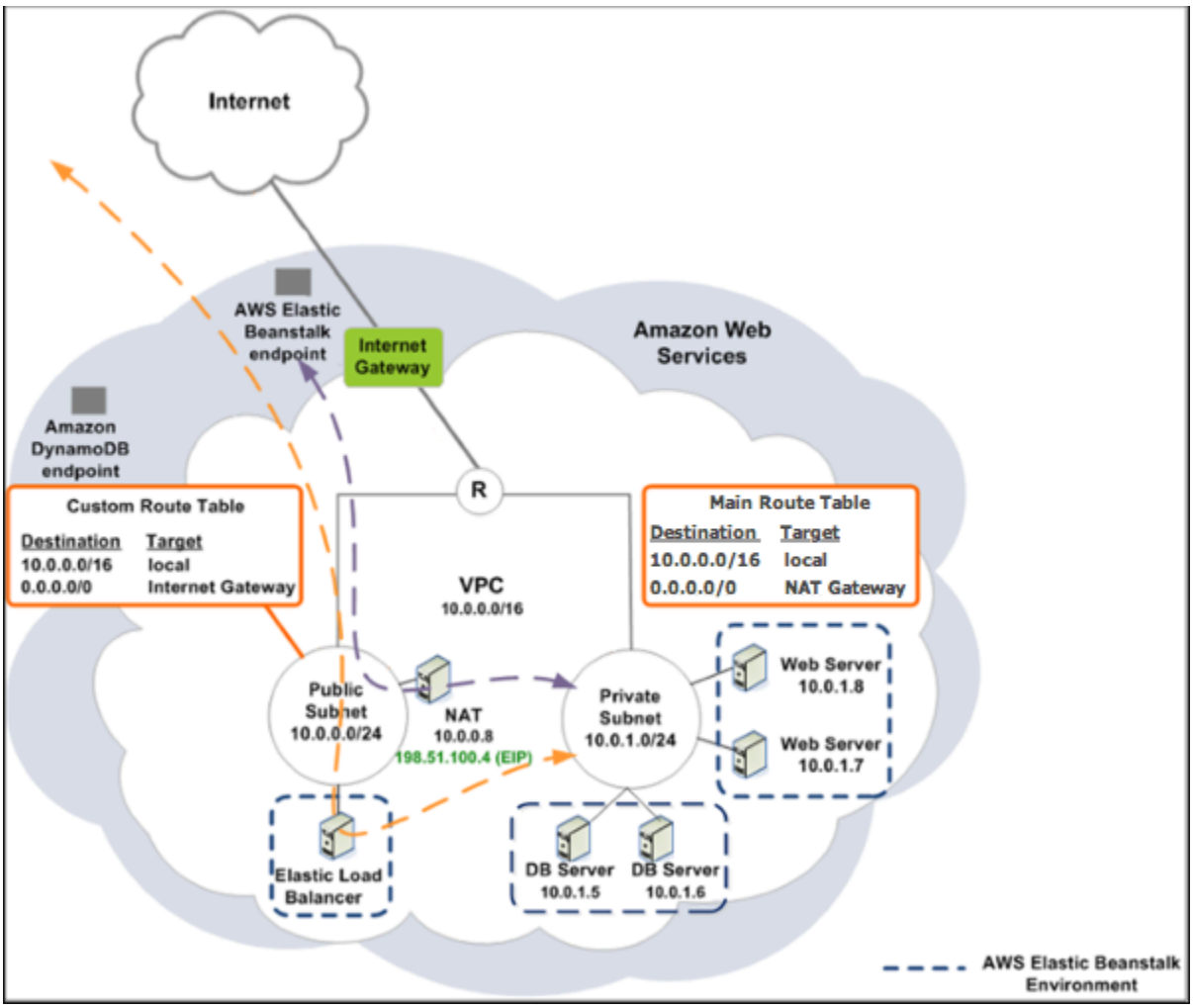
Pour plus d'informations sur la configuration d'un hôte bastion pour les instances Windows dans le sous-réseau privé, consultez la section Contrôle de l'accès réseau aux EC2 instances à l'aide d'un serveur Bastion.

Pour de plus amples informations sur la configuration d'un hôte bastion pour les instances Linux sur le sous-réseau privé, veuillez consulter l'article de blog Connexion sécurisée aux instances Linux exécutées dans un VPC privé Amazon.

Exemple : Lancement d'un Elastic Beanstalk dans un VPC avec Amazon RDS

Cette section vous guide à travers les tâches de déploiement d'une application Elastic Beanstalk avec Amazon RDS dans un VPC à l'aide d'une passerelle NAT.

Votre infrastructure ressemblera au schéma suivant.



> **Note**
>
> Si vous n'avez pas utilisé une instance de base de données avec votre application auparavant, essayez d'en ajouter une à un environnement de test et d'établir une connexion à une instance de base de données externe avant d'ajouter une configuration VPC au mélange.

Création d'un VPC avec des sous-réseaux public et privé

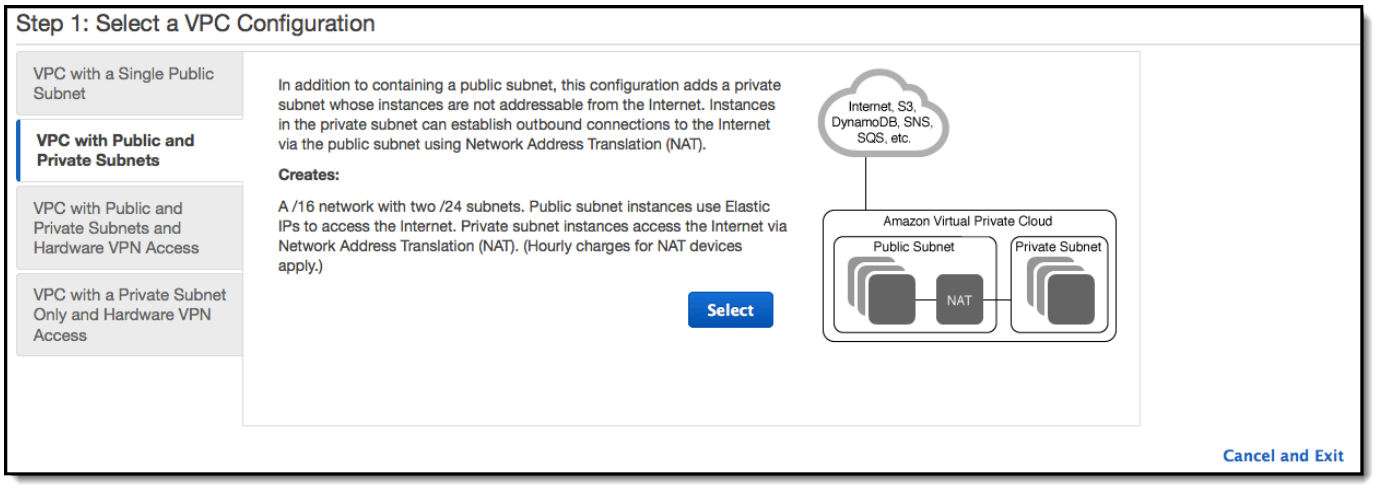
Vous pouvez utiliser la console Amazon VPC pour créer un VPC.

Pour créer un VPC

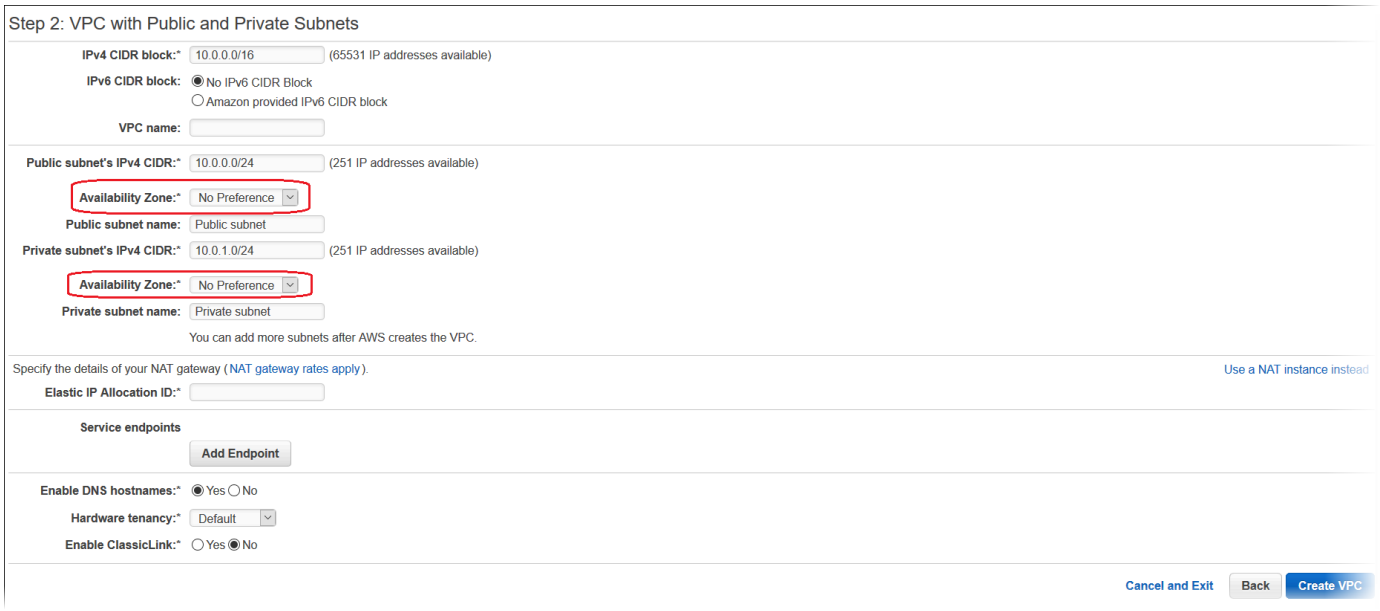
1. Connectez-vous à la console Amazon VPC.

2. Dans le panneau de navigation, choisissez Tableau de bord du VPC. Ensuite, choisissez Créer VPC.

3. Choisissez VPC avec des sous-réseaux publics et privés, puis Sélectionner.



4. Votre équilibreur de charge Elastic Load Balancing et vos EC2 instances Amazon doivent se trouver dans la même zone de disponibilité pour pouvoir communiquer entre eux. Choisissez la même zone de disponibilité dans chaque liste Zone de disponibilité.



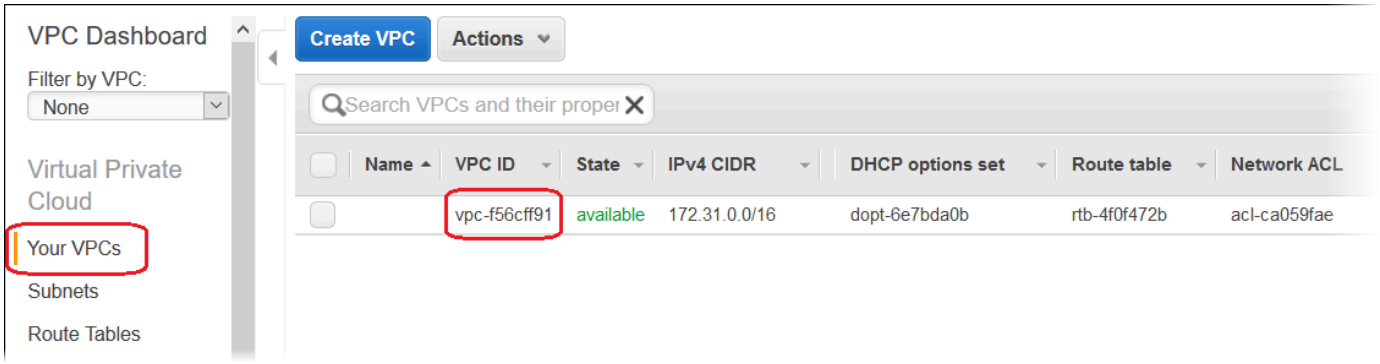
5. Choisissez une adresse IP Elastic pour votre passerelle NAT.

6. Sélectionnez Create VPC (Créer un VPC).

   L'Assistant commence à créer le VPC, les sous-réseaux et la passerelle Internet. Il met également à jour la table de routage principale et crée une table de routage personnalisée. Enfin, il crée une passerelle NAT dans le sous-réseau public.

> **Note**
>
> Vous pouvez choisir de lancer une instance NAT dans le sous-réseau public au lieu d'une passerelle NAT. Pour de plus amples informations, veuillez consulter Scénario 2 : VPC avec des sous-réseaux publics et privés (NAT) dans le Guide de l'utilisateur Amazon VPC.

7. Une fois que le VPC est créé avec succès, vous obtenez un ID de VPC. Vous avez besoin de cette valeur pour l'étape suivante. Pour afficher votre ID VPC, choisissez Your VPCs dans le volet gauche de la console Amazon VPC.



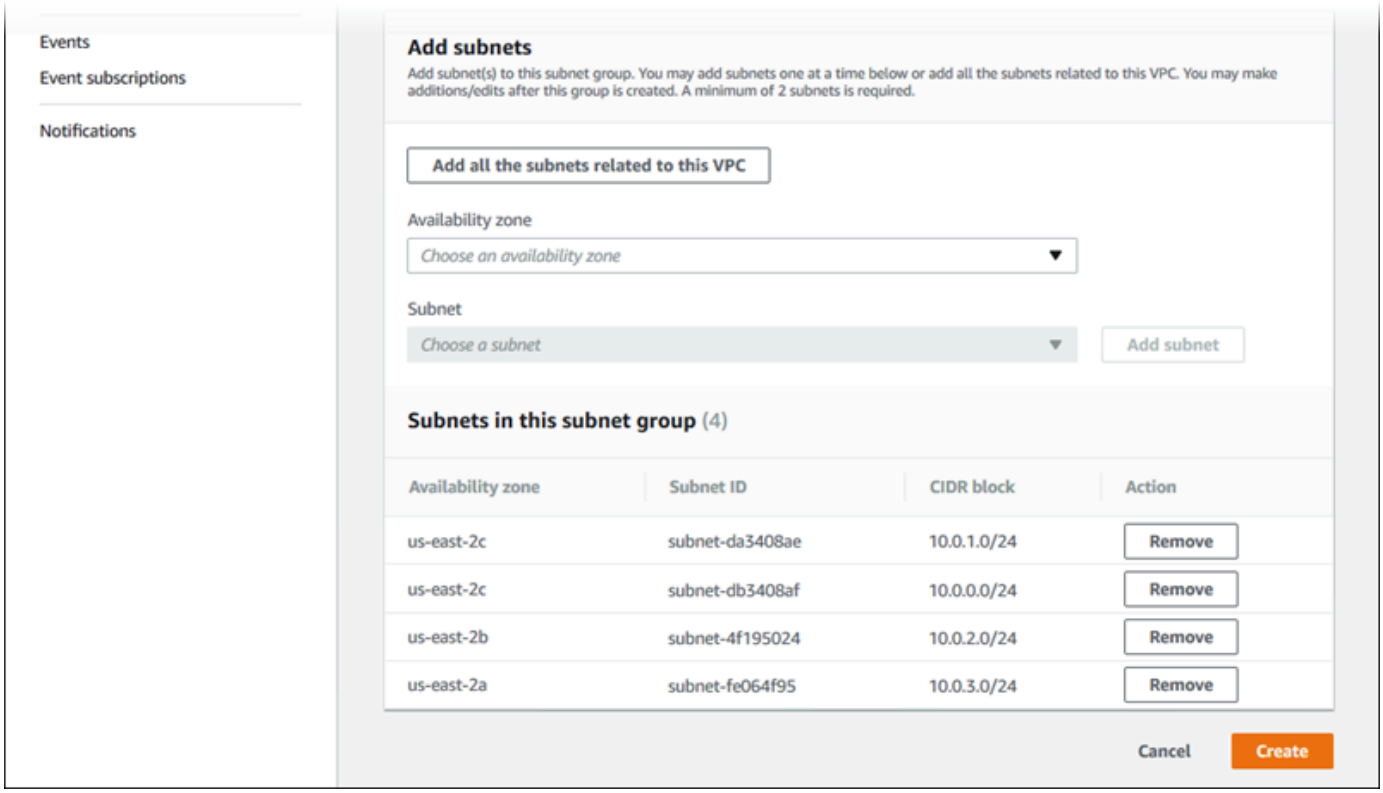
Création d'un groupe de sous-réseaux de base de données

Un groupe de sous-réseaux DB pour un VPC est un ensemble de sous-réseaux (généralement privés) que vous pouvez désigner pour vos instances backend DB RDS. Chaque groupe de sous-réseaux de base de données doit avoir au moins un sous-réseau pour chaque zone de disponibilité d'une région donnée AWS . Pour de plus amples informations, veuillez consulter Création d'un sous-réseau dans votre VPC.

Création d'un groupe de sous-réseaux de base de données

1. Ouvrez la console Amazon RDS.
2. Dans le panneau de navigation, choisissez Subnet groups (Groupes de sous-réseaux).
3. Choisissez Create DB Subnet Group (Créer groupe de sous-réseaux de base de données).
4. Choisissez Name (Nom), puis saisissez le nom de votre groupe de sous-réseaux DB.
5. Choisissez Description, puis décrivez votre groupe de sous-réseaux DB.
6. Pour VPC, choisissez l'ID du VPC que vous avez créé précédemment.

7. Dans Add subnets (Ajouter des sous-réseaux), choisissez Add all the subnets related to this VPC (Ajouter tous les sous-réseaux associés à ce VPC).



8. Lorsque vous avez terminé, sélectionnez Create.

   Votre nouveau groupe de sous-réseaux DB s'affiche dans la liste des groupes de sous-réseaux de base de données de la console Amazon RDS. Vous pouvez le choisir pour afficher des détails, tels que tous les sous-réseaux associés à ce groupe, dans le volet des détails en bas de la page.

Déploiement sur Elastic Beanstalk

Une fois que vous avez configuré votre VPC, vous pouvez y créer votre environnement et déployer votre application sur Elastic Beanstalk. Pour ce faire, vous pouvez utiliser la console Elastic Beanstalk, les boîtes à outils AWS CLI, l'EB CLI ou AWS l'API Elastic Beanstalk. Si vous utilisez la console Elastic Beanstalk, il vous suffit de télécharger votre fichier `.war` ou `.zip` et de sélectionner les paramètres VPC dans l'assistant. Elastic Beanstalk crée ensuite votre environnement dans votre VPC et déploie votre application. Vous pouvez également utiliser les AWS boîtes à outils AWS CLI, la CLI EB ou l'API Elastic Beanstalk pour déployer votre application. Pour ce faire, vous devez définir vos paramètres VPC dans un fichier de configuration et déployer ce fichier avec votre groupe source. Cette rubrique fournit des instructions pour les deux méthodes.

Déploiement avec la console Elastic Beanstalk

La console Elastic Beanstalk vous guide à travers la création de votre nouvel environnement dans votre VPC. Vous devez fournir un fichier `.war` (pour les applications Java) ou un fichier `.zip` (pour toutes les autres applications). Sur la page Configuration VPC de l'assistant d'environnement Elastic Beanstalk, vous devez effectuer les sélections suivantes :

VPC

Sélectionnez votre VPC.

Groupe de sécurité VPC

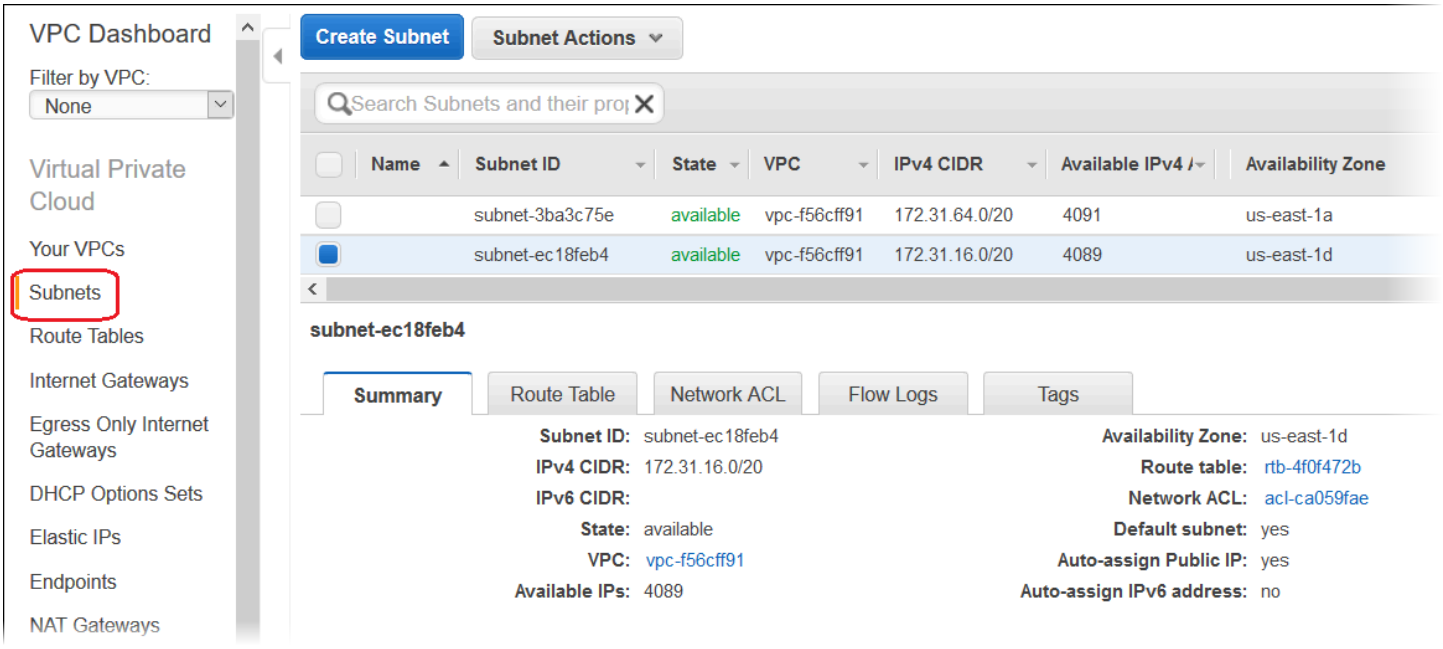
Sélectionnez le groupe de sécurité d'instance que vous avez créé précédemment.

Visibilité ELB

Sélectionnez `External` si votre équilibreur de charge doit être publiquement disponible, ou `Internal` si l'équilibreur de charge ne doit être disponible que dans votre VPC.

Sélectionnez les sous-réseaux pour votre équilibreur de charge et EC2 vos instances. Assurez-vous de sélectionner le sous-réseau public pour l'équilibreur de charge et le sous-réseau privé pour vos instances Amazon. EC2 Par défaut, l'assistant de création de VPC crée le sous-réseau public dans `10.0.0.0/24` et le sous-réseau privé dans `10.0.1.0/24`.

Vous pouvez consulter votre sous-réseau IDs en choisissant Subnets dans la console Amazon VPC.



Déploiement à l'aide des AWS boîtes à outils, de la CLI EB ou AWS CLI de l'API

Lorsque vous déployez votre application sur Elastic Beanstalk à l'aide des boîtes à outils, de AWS la AWS CLI CLI EB ou de l'API, vous pouvez spécifier les paramètres de vos options VPC dans un fichier et les déployer avec votre bundle source. Pour plus d'informations, consultez Personnalisation d'environnement avancée avec fichiers de configuration (`.ebextensions`).

Lorsque vous mettez à jour les paramètres de l'option, vous devez spécifier au moins les éléments suivants :

- VPCId—Contient l'ID du VPC.
- Subnets (Sous-réseaux) – Contient l'ID du sous-réseau du groupe Auto Scaling. Dans cet exemple, il s'agit de l'ID du sous-réseau privé.
- ELBSubnets: contient l'ID du sous-réseau de l'équilibreur de charge. Dans cet exemple, il s'agit de l'ID du sous-réseau public.
- SecurityGroups—Contient l'ID des groupes de sécurité.
- DBSubnets: contient l'ID des sous-réseaux de base de données.

> **Note**
>
> Lorsque vous utilisez des sous-réseaux de base de données, vous devez créer des sous-réseaux supplémentaires dans votre VPC pour couvrir toutes les zones de disponibilité de la région. AWS

Le cas échéant, vous pouvez également spécifier les informations suivantes :

- ELBScheme— Spécifiez `internal` de créer un équilibreur de charge interne dans votre VPC afin que votre application Elastic Beanstalk ne soit pas accessible depuis l'extérieur de votre VPC.

Voici un exemple des paramètres d'options que vous pouvez utiliser lors du déploiement de votre application Elastic Beanstalk à l'intérieur d'un VPC. Pour de plus amples informations sur les paramètres d'options VPC (y compris les exemples pour savoir comment les spécifier, les valeurs par défaut et les valeurs valides), veuillez consulter le tableau d'espace de noms aws:ec2:vpc dans Options de configuration.

```
option_settings:
```

```
  - namespace: aws:autoscaling:launchconfiguration
    option_name: EC2KeyName
    value: ec2keypair

  - namespace: aws:ec2:vpc
    option_name: VPCId
    value: vpc-170647c

  - namespace: aws:ec2:vpc
    option_name: Subnets
    value: subnet-4f195024

  - namespace: aws:ec2:vpc
    option_name: ELBSubnets
    value: subnet-fe064f95

  - namespace: aws:ec2:vpc
    option_name: DBSubnets
    value: subnet-fg148g78

  - namespace: aws:autoscaling:launchconfiguration
    option_name: InstanceType
    value: m1.small

  - namespace: aws:autoscaling:launchconfiguration
    option_name: SecurityGroups
    value: sg-7f1ef110
```

> **Note**
>
> Lorsque vous utilisez des sous-réseaux de base de données, assurez-vous que votre VPC comporte des sous-réseaux couvrant toutes les zones de disponibilité de la région. AWS

Utilisation d'Elastic Beanstalk avec les points de terminaison VPC

Cette rubrique explique les avantages qu'un point de terminaison VPC peut offrir à votre application Elastic Beanstalk. Il fournit également des instructions pour créer un point de terminaison VPC d'interface vers un service Elastic Beanstalk.

Un point de terminaison VPC vous permet de connecter de manière privée votre VPC aux services pris en charge AWS et aux services de point de terminaison VPC alimentés par AWS PrivateLink,

sans avoir besoin d'une passerelle Internet, d'un périphérique NAT, d'une connexion VPN ou d'une connexion. Direct Connect

Les instances de votre VPC ne requièrent pas d'adresses IP publiques pour communiquer avec les ressources du service. Le trafic entre votre VPC et les autres services ne quitte pas le réseau Amazon. Pour des informations complètes sur les points de terminaison d'un VPC, veuillez consulter Points de terminaison d'un VPC dans le Guide de l'utilisateur Amazon VPC.

AWS Elastic Beanstalk supports AWS PrivateLink, qui fournit une connectivité privée au service Elastic Beanstalk et élimine l'exposition du trafic à l'Internet public. Pour permettre à votre application d'envoyer des demandes à Elastic AWS PrivateLink Beanstalk à l'aide de, vous configurez un type de point de terminaison VPC appelé point de terminaison VPC d'interface (point de terminaison d'interface). Pour de plus amples informations, consultez Points de terminaison VPC (AWS PrivateLink) dans le Guide de l'utilisateur Amazon VPC.

> **Note**
>
> Elastic AWS PrivateLink Beanstalk prend en charge et interface les points de terminaison VPC dans un nombre limité de. Régions AWS Nous nous efforçons d'étendre le support à d'autres Régions AWS personnes dans un futur proche.

IPv6 support avec configuration à double pile

Elastic Beanstalk prend en charge le trafic entrant sur et. IPv4 IPv6 Cette section décrit les points de terminaison publics qui prennent en charge les deux IPv4 IPv6 et explique également comment configurer vos points de terminaison VPC Elastic Beanstalk pour prendre en charge le trafic à double pile.

Points de terminaison publics

Le service Elastic Beanstalk comporte deux ensembles de points de terminaison composés des points de terminaison les plus anciens et des points de terminaison IPv4 les plus récents dotés d'une capacité de double pile. Les deux ensembles de points de terminaison respectent les normes AWS de dénomination :

- IPv4 les points de terminaison utilisent le format de domaine pour le point de terminaison de service général `amazonaws.com` : `elasticbeanstalk.region.amazonaws.com`

- Les points de terminaison à double pile utilisent le format de domaine pour le point de terminaison de service général `api.aws` : `elasticbeanstalk.region.api.aws`

Les points de terminaison relatifs à l'état du service et à la norme FIPS portent des noms d'hôtes différents, mais ils suivent le même modèle de nom de domaine. Pour obtenir la liste des points de terminaison, voir les points de terminaison du service Elastic Beanstalk dans le. Référence générale d'Amazon Web Services

Demandes adressées à Elastic Beanstalk

Lorsque vous envoyez des demandes au service Elastic Beanstalk AWS CLIavec le SDK ou le SDK, vous pouvez IPv4 spécifier un point de AWS terminaison ou un point de terminaison à double pile. Le AWS SDK AWS CLI et utilise les points de IPv4 terminaison uniquement par défaut si aucune URL de point de terminaison n'est spécifiée.

L'exemple suivant illustre l' AWS CLI envoi d'une demande à un point de terminaison à double pile :

Example

```
aws elasticbeanstalk list-available-solution-stacks \
    --endpoint-url "https://elasticbeanstalk.us-east-1.api.aws"
```

L'exemple suivant montre que le SDK AWS Python envoie une requête à un point de terminaison à double pile :

Example

```
import boto3

dual_stack_eb_client = boto3.client(
    service_name='elasticbeanstalk',
    region_name='us-east-1',
    endpoint_url='https://elasticbeanstalk.us-east-1.api.aws';
)

print(dual_stack_eb_client.list_available_solution_stacks())
```

Points de terminaison VPC pour double pile IPs

Pour configurer vos points de terminaison VPC Elastic Beanstalk afin de prendre en charge le trafic à double pile, spécifiez dualstack pour le paramètre de type d'adresse IP du point de terminaison VPC.

Vous pouvez spécifier ce champ via AWS CLIle AWS SDK ou la AWS PrivateLink console. Pour obtenir des instructions à ce sujet dans la AWS PrivateLink console, consultez la section Créer un point de terminaison VPC dans le AWS PrivateLink guide.

> **Note**
>
> Vous devez spécifier le type d'adresse IP du point de terminaison VPC sous la forme d'une pile IPv4ou d'une pile double. À l'heure actuelle, les points de terminaison VPC Elastic Beanstalk ne prennent pas en charge le type d'adresse IPv6IP, ce qui indiquerait une prise en charge uniquement. IPv6 L'option dualstack autorise à la fois le protocole IPv6 Internet IPv4 et le protocole Internet.

L'exemple suivant montre comment créer un point de terminaison VPC à double pile avec : AWS CLI

Example

```
aws ec2 create-vpc-endpoint \
    --vpc-id "vpc-example"
    --service-name "com.amazonaws.us-east-1.elasticbeanstalk"
    --ip-address-type "dualstack"
```

Configuration d'un point de terminaison VPC pour Elastic Beanstalk

Pour créer le point de terminaison d'un VPC d'interface pour le service Elastic Beanstalk dans votre VPC, suivez la procédure Création d'un point de terminaison d'interface.

- Pour Nom du service, choisissez com.amazonaws. ***region***. tige de haricot élastique.
- Pour le type d'adresse IP, choisissez Dualstack IPv4ou Dualstack. À l'heure actuelle, les points de terminaison VPC Elastic Beanstalk ne prennent pas en charge le type d'adresse IPv6IP, ce qui indiquerait une prise en charge uniquement. IPv6 L'option Dualstack permet à la fois le protocole IPv6 Internet IPv4 et le protocole Internet.

Si votre VPC est configuré avec un accès public à Internet, votre application peut toujours accéder à Elastic Beanstalk via Internet en utilisant le point de terminaison public ou le point de terminaison public. `elasticbeanstalk.region.amazonaws.com` `elasticbeanstalk.region.api.aws` Pour empêcher cela, assurez-vous que Enable DNS name (Activer le nom DNS) est activé lors de la

création du point de terminaison (true par défaut). Cela ajoute une entrée DNS dans votre VPC qui mappe le point de terminaison du service public au point de terminaison de VPC d'interface.

Configuration d'un point de terminaison de VPC pour l'intégrité améliorée

Si vous avez activé les rapports de santé améliorés pour votre environnement, vous pouvez AWS PrivateLink également configurer l'envoi d'informations de santé améliorées. Les informations d'intégrité améliorées sont envoyées par le `healthd` démon, un composant Elastic Beanstalk sur vos instances d'environnement, à un service d'intégrité amélioré Elastic Beanstalk distinct. Pour créer un point de terminaison d'un VPC d'interface pour ce service dans votre VPC, suivez la procédure Création d'un point de terminaison d'interface.

- Pour Nom du service, choisissez com.amazonaws. ***region***. elasticbeanstalk-health.
- Pour le type d'adresse IP, choisissez Dualstack IPv4ou Dualstack. À l'heure actuelle, les points de terminaison VPC Elastic Beanstalk ne prennent pas en charge le type d'adresse IPv6IP, ce qui indiquerait une prise en charge uniquement. IPv6 L'option Dualstack permet à la fois le protocole IPv6 Internet IPv4 et le protocole Internet.

> **Important**
>
> Le `healthd` daemon envoie des informations de santé améliorées au point de terminaison public `elasticbeanstalk-health.`*`region`*`.amazonaws.com` ou`elasticbeanstalk-health.`*`region`*`.api.aws`. Si votre VPC est configuré avec un accès Internet public et que Enable DNS name (Activer nom DNS) est désactivé pour le point de terminaison de VPC, des informations d'intégrité améliorée circulent sur l'Internet public. Ce n'est probablement pas votre intention lorsque vous configurez un point de terminaison d'un VPC d'intégrité améliorée. Assurez-vous que Enable DNS name (Activer nom DNS) est activé (true par défaut).

Utilisation de points de terminaison d'un VPC dans un VPC privé

Un VPC privé, ou un sous-réseau privé dans un VPC, n'a pas d'accès Internet public. Vous pouvez exécuter votre environnement Elastic Beanstalk dans un VPC privé et configurer des points de terminaison de VPC d'interface pour une sécurité renforcée. Dans ce cas, sachez que votre environnement peut essayer de se connecter à Internet pour d'autres raisons que celle de contacter le service Elastic Beanstalk. Pour en savoir plus sur l'exécution d'un environnement dans un VPC

privé, consultez the section called “Exécution d'un environnement Elastic Beanstalk dans un VPC privé”.

Utilisation des stratégies de point de terminaison pour contrôler l'accès avec des points de terminaison de VPC

Cette rubrique explique comment vous pouvez associer une politique aux points de terminaison VPC afin de contrôler l'accès à votre application (votre service) et à votre environnement Elastic Beanstalk.

Une politique de point de terminaison est une politique de ressources Gestion des identités et des accès AWS (IAM) qui contrôle l'accès du point de terminaison au service spécifié. La stratégie de point de terminaison est spécifique au point de terminaison. Elle est distincte des stratégies IAM d'utilisateur ou d'instance que votre environnement peut avoir et ne les remplace pas.

Par défaut, un point de terminaison de VPC permet un accès complet au service auquel il est associé. Lorsque vous créez ou modifiez un point de terminaison, vous pouvez y associer une politique de point de terminaison pour contrôler l'accès à des ressources spécifiques associées au service. Pour plus d'informations sur la création et l'utilisation de politiques de point de terminaison VPC, consultez la section Contrôler l'accès aux points de terminaison VPC à l'aide de politiques de point de terminaison dans le Guide.AWS PrivateLink

> **Note**
>
> Lorsque vous créez des politiques de point de terminaison restrictives, vous devrez peut-être ajouter des autorisations spécifiques aux ressources requises, afin que l'accès à ces ressources ne soit pas bloqué par la politique de point de terminaison. Cela garantit que votre environnement continue de se déployer et de fonctionner correctement.

Dans l’exemple suivant, on refuse à tous les utilisateurs l'autorisation de mettre fin à un environnement via le point de terminaison de VPC et on autorise un accès complet à toutes les autres actions.

```
{
    "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
```

```
            "Principal": "*"
        },
        {
            "Action": "elasticbeanstalk:TerminateEnvironment",
            "Effect": "Deny",
            "Resource": "*",
            "Principal": "*"
        }
    ]
}
```

Autorisations de compartiment Amazon S3 requises pour les politiques de point de terminaison VPC restrictives

Si vous ajoutez des restrictions à vos politiques de point de terminaison VPC, vous devez inclure des autorisations de compartiment Amazon S3 spécifiques pour garantir que votre environnement continue à se déployer et à fonctionner correctement. Cette section explique les compartiments S3 requis et inclut des exemples de politiques.

Rubriques

- Des compartiments S3 qui stockent des actifs pour gérer les plateformes d'environnement
- Buckets S3 appartenant à CloudFormation
- Compartiments S3 détenus par les comptes clients pour stocker le code source et d'autres éléments
- Des compartiments S3 appartenant à des comptes clients pour prendre en charge l'authentification du registre Docker
- Mise à jour de votre politique de point de terminaison VPC

Des compartiments S3 qui stockent des actifs pour gérer les plateformes d'environnement

Le service Elastic Beanstalk possède des compartiments S3 qui stockent les actifs associés à une pile de solutions (version de plateforme). Ces actifs incluent les fichiers de configuration, l'exemple d'application et les types d'instances disponibles. Lorsqu'Elastic Beanstalk crée et gère votre environnement, il extrait les informations requises pour chaque version de plate-forme dans le compartiment d'actifs correspondant. Région AWS

ARN du compartiment S3

`arn:aws:s3:::elasticbeanstalk-samples-`*`region`*

Amazon Linux 2 et versions ultérieures

- `arn:aws:s3:::elasticbeanstalk-platform-assets-region`

> **Note**
>
> Le nom du bucket suit une convention différente pour la région BJS. La chaîne public-beta-cn-north-1 est utilisée à la place de*region*. Par exemple, `arn:aws:s3:::elasticbeanstalk-platform-assets-public-beta-cn-north-1`.

Windows Server, Amazon Linux (AMI), Amazon Linux 2 et versions ultérieures

- `arn:aws:s3:::elasticbeanstalk-env-resources-region`
- `arn:aws:s3:::elasticbeanstalk-region`

> **Note**
>
> Les noms de bucket pour les buckets platform-assets et env-resources suivent des conventions différentes dans certaines régions. Consultez la section sur les modèles de dénomination des compartiments spécifiques à la région ci-dessous pour plus de détails.

Modèles d'ARN de bucket spécifiques à une région

Région	Type de godet	Motif Bucket ARN
me-central-1	actifs de la plateforme	`arn:aws:s3:::elasticbeanstalk-platform-assets-me-central-1-f08b818c`
	ressources environnementales	`arn:aws:s3:::elasticbeanstalk-env-resources-me-central-1-f08b818c`

Opérations

GetObject

Exemple de politique de point de terminaison VPC

L'exemple suivant montre comment fournir l'accès aux compartiments S3 requis pour les opérations d'Elastic Beanstalk dans la région USA Est (Ohio) (us-east-2). L'exemple répertorie tous les buckets pour les plateformes Amazon Linux et Windows Server. Mettez à jour votre politique pour inclure uniquement les buckets qui s'appliquent au système d'exploitation de votre environnement.

Exemple de stratégie

> **Important**
>
> Dans cette politique, nous vous recommandons d'éviter d'utiliser des caractères génériques (*) à la place des régions spécifiques. Par exemple, utilisez `arn:aws:s3:::cloudformation-waitcondition-us-east-2/*` et n'utilisez pas `arn:aws:s3:::cloudformation-waitcondition-*/*`. L'utilisation de caractères génériques pourrait fournir l'accès aux compartiments S3 vers lesquels vous ne prévoyez pas d'accorder l'accès. Si vous souhaitez utiliser la politique pour plusieurs régions, nous vous recommandons de répéter le premier `Statement` bloc pour chaque région.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRequestsToAWSResources",
            "Effect": "Allow",
            "Principal": {"AWS": "*"},
            "Action": ["s3:GetObject"],
            "Resource": [
                "arn:aws:s3:::elasticbeanstalk-platform-assets-us-east-2/*",
                "arn:aws:s3:::elasticbeanstalk-env-resources-us-east-2/*",
                "arn:aws:s3:::elasticbeanstalk-env-resources-us-east-2/*",
                "arn:aws:s3:::elasticbeanstalk-samples-us-east-2/*"
            ]
```

```
        }
    ]
}
```

Buckets S3 appartenant à CloudFormation

Elastic CloudFormation Beanstalk permet de créer des ressources pour votre environnement. CloudFormation possède des compartiments S3 dans chacun d'eux Région AWS pour surveiller les réponses aux conditions d'attente.

Des services tels qu'Elastic Beanstalk CloudFormation communiquent avec eux en envoyant des demandes à une URL Amazon S3 présignée pour le compartiment S3 qui en est le propriétaire. CloudFormation CloudFormation crée l'URL Amazon S3 présignée à l'aide du principal `cloudformation.amazonaws.com` de service.

Pour des informations plus détaillées, consultez la section Considérations relatives aux points de terminaison CloudFormation VPC dans le guide de l'AWS CloudFormation utilisateur. Pour en savoir plus sur le présigné URLs, consultez la section Travailler avec le présigné URLs dans le guide de l'utilisateur Amazon S3.

ARN du compartiment S3

- `arn:aws:s3:::cloudformation-waitcondition-`*`region`*

  Lors de l'utilisation de conditions d'attente, les noms de région contiennent des tirets. Par exemple, us-west-2.
- `arn:aws:s3:::cloudformation-custom-resource-response-`*`region`*

  Lors de l'utilisation de ressources personnalisées, les noms de région ne contiennent pas de tirets. Par exemple, uswest2.

Opérations

GetObject

Exemple de politique de point de terminaison VPC

L'exemple suivant montre comment fournir l'accès aux compartiments S3 requis pour les opérations d'Elastic Beanstalk dans la région USA Est (Ohio) (us-east-2).

Exemple de stratégie

> **Important**
>
> Dans cette politique, nous vous recommandons d'éviter d'utiliser des caractères génériques (*) à la place des régions spécifiques. Par exemple, utilisez `arn:aws:s3:::cloudformation-waitcondition-us-east-2/*` et n'utilisez pas `arn:aws:s3:::cloudformation-waitcondition-*/*`. L'utilisation de caractères génériques pourrait fournir l'accès aux compartiments S3 vers lesquels vous ne prévoyez pas d'accorder l'accès. Si vous souhaitez utiliser la politique pour plusieurs régions, nous vous recommandons de répéter le premier `Statement` bloc pour chaque région.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRequestsToCloudFormation",
            "Effect": "Allow",
            "Principal": {"AWS": "*"},
            "Action": ["s3:GetObject"],
            "Resource": [
                "arn:aws:s3:::cloudformation-waitcondition-us-east-2/*",
                "arn:aws:s3:::cloudformation-custom-resource-response-us-east-2/
*"
            ]
         }
    ]
}
```

Compartiments S3 détenus par les comptes clients pour stocker le code source et d'autres éléments

Ce bucket appartient au compte AWS client propriétaire de l'environnement. Il stocke les ressources spécifiques à votre environnement, telles que le code source et les journaux demandés.

ARN du compartiment S3

`arn:aws:s3:::elasticbeanstalk-region-account-id`

Opérations

- GetObject
- GetObjectAcl
- PutObject
- PutObjectAcl
- ListBucket

Exemple de politique de point de terminaison VPC

L'exemple suivant montre comment fournir l'accès aux compartiments S3 requis pour les opérations d'Elastic Beanstalk dans la région USA Est (Ohio) (us-east-2) et pour l'exemple, l'identifiant de compte 123456789012. AWS

Exemple de stratégie

> **Important**
>
> Dans cette politique, nous vous recommandons d'éviter d'utiliser des caractères génériques (*) à la place des régions spécifiques. Par exemple, utilisez `arn:aws:s3:::cloudformation-waitcondition-us-east-2/*` et n'utilisez pas `arn:aws:s3:::cloudformation-waitcondition-*/*`. L'utilisation de caractères génériques pourrait fournir l'accès aux compartiments S3 vers lesquels vous ne prévoyez pas d'accorder l'accès. Si vous souhaitez utiliser la politique pour plusieurs régions, nous vous recommandons de répéter le premier `Statement` bloc pour chaque région.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRequestsToCustomerItems",
            "Effect": "Allow",
            "Principal": {"AWS": "*"},
            "Action": [
               "s3:GetObject",
```

```
                "s3:GetObjectAcl",
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::elasticbeanstalk-us-east-2-123456789012/*"
            ]
        }
    ]
}
```

Des compartiments S3 appartenant à des comptes clients pour prendre en charge l'authentification du registre Docker

Ce bucket s'applique uniquement aux environnements basés sur la plateforme Docker. Le compartiment stocke un fichier utilisé pour s'authentifier auprès d'un registre Docker privé qui réside sur un compartiment S3 provisionné par le client. Pour plus d'informations, consultez la section Utilisation du fichier `Dockerrun.aws.json` v3 dans le chapitre de ce guide consacré à la plateforme Docker.

ARN du compartiment S3

L'ARN varie en fonction du compte client.

L'ARN du compartiment S3 a le format suivant : `arn:aws:s3:::`*`bucket-name`*

Opérations

GetObject

Exemple de politique de point de terminaison VPC

L'exemple suivant montre comment fournir l'accès à un compartiment S3 nommé amzn-s3-demo-bucket1.

Exemple de stratégie

JSON

```
  {
  "Version":"2012-10-17",
  "Statement": [
     {
        "Sid": "AllowRequestsToDockerRegistryAuth",
         "Effect": "Allow",
         "Action": [
            "s3:GetObject"
          ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket1"
      ]
    }
  ]
}
```

Mise à jour de votre politique de point de terminaison VPC

Étant donné qu'un point de terminaison VPC n'est associé qu'à une seule stratégie, vous devez combiner toutes les autorisations dans une seule stratégie. L'exemple suivant regroupe tous les exemples précédents en un seul.

Pour plus d'informations sur la création et l'utilisation de politiques de point de terminaison VPC, consultez la section Contrôler l'accès aux points de terminaison VPC à l'aide de politiques de point de terminaison dans le Guide.AWS PrivateLink

Comme dans les exemples précédents, l'exemple suivant montre comment fournir l'accès aux compartiments S3 nécessaires aux opérations d'Elastic Beanstalk dans la région USA Est (Ohio) (us-east-2). Il inclut également des buckets dont l'identifiant de AWS compte est 123456789012 et le nom d'exemple du bucket amzn-s3-demo-bucket1.

> **Important**
>
> Dans cette politique, nous vous recommandons d'éviter d'utiliser des caractères génériques (*) à la place des régions spécifiques. Par exemple, utilisez `arn:aws:s3:::cloudformation-waitcondition-us-east-2/*` et n'utilisez pas `arn:aws:s3:::cloudformation-waitcondition-*/*`. L'utilisation de caractères génériques pourrait fournir l'accès aux compartiments S3 vers lesquels vous ne prévoyez pas

d'accorder l'accès. Si vous souhaitez utiliser la politique pour plusieurs régions, nous vous recommandons de répéter le premier `Statement` bloc pour chaque région.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRequestsToAWSResources",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::elasticbeanstalk-platform-assets-us-east-2/*",
                "arn:aws:s3:::elasticbeanstalk-env-resources-us-east-2/*",
                "arn:aws:s3:::elasticbeanstalk-samples-us-east-2/*"
            ]
        },
        {
            "Sid": "AllowRequestsToCloudFormation",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::cloudformation-waitcondition-us-east-2/*",
                "arn:aws:s3:::cloudformation-custom-resource-response-us-east-2/
*"
            ]
        },
        {
            "Sid": "AllowRequestsToCustomerItems",
            "Effect": "Allow",
            "Action": [
               "s3:GetObject",
               "s3:GetObjectAcl",
              "s3:PutObject",
               "s3:PutObjectAcl",
               "s3:ListBucket"
```

```
            ],
            "Resource": [
                "arn:aws:s3:::elasticbeanstalk-us-east-2-123456789012/*"
            ]
        },
        {
            "Sid": "AllowRequestsToDockerRegistryAuth",
            "Effect": "Allow",
            "Action": [
            "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1"
            ]
        }
    ]
}
```

AWS Elastic Beanstalk sécurité

Consultez ce chapitre pour en savoir plus sur les tâches de sécurité dont Elastic Beanstalk est responsable, ainsi que sur les configurations de sécurité à prendre en compte lorsque vous utilisez Elastic Beanstalk pour atteindre vos objectifs de sécurité et de conformité.

La sécurité du cloud AWS est la priorité absolue. En tant que AWS client, vous bénéficiez d'un centre de données et d'une architecture réseau conçus pour répondre aux exigences des entreprises les plus sensibles en matière de sécurité.

La sécurité est une responsabilité partagée entre vous AWS et vous. Le modèle de responsabilité partagée décrit ceci comme la sécurité du cloud et la sécurité dans le cloud.

Sécurité du cloud : AWS est chargée de protéger l'infrastructure qui exécute tous les services proposés dans le AWS cloud et de vous fournir des services que vous pouvez utiliser en toute sécurité. Notre responsabilité en matière de sécurité est notre priorité absolue AWS, et l'efficacité de notre sécurité est régulièrement testée et vérifiée par des auditeurs tiers dans le cadre des programmes de AWS conformité. Consultez les AWS services concernés par les programmes d' AWS assurance pour obtenir des informations relatives à Elastic Beanstalk.

Sécurité dans le cloud — Votre responsabilité est déterminée par le AWS service que vous utilisez et par d'autres facteurs, notamment la sensibilité de vos données, les exigences de votre organisation et les lois et réglementations applicables. Cette documentation est conçue pour vous aider à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation d'Elastic Beanstalk.

Rubriques

- Protection des données dans Elastic Beanstalk
- Gestion des identités et des accès pour Elastic Beanstalk
- Journalisation et surveillance dans Elastic Beanstalk
- Validation de la conformité pour Elastic Beanstalk
- Résilience dans Elastic Beanstalk
- Sécurité de l'infrastructure dans Elastic Beanstalk
- Configuration et analyse des vulnérabilités dans Elastic Beanstalk
- Bonnes pratiques de sécurité pour Elastic Beanstalk

Protection des données dans Elastic Beanstalk

Le modèle de responsabilité AWS partagée de s'applique à la protection des données dans AWS Elastic Beanstalk. Comme décrit dans ce modèle, AWS est chargé de protéger l'infrastructure mondiale qui gère tous les AWS Cloud. La gestion du contrôle de votre contenu hébergé sur cette infrastructure relève de votre responsabilité. Vous êtes également responsable des tâches de configuration et de gestion de la sécurité des Services AWS que vous utilisez. Pour plus d'informations sur la confidentialité des données, consultez Questions fréquentes (FAQ) sur la confidentialité des données. Pour en savoir plus sur la protection des données en Europe, consultez le billet de blog Modèle de responsabilité partagée d'AWS et RGPD (Règlement général sur la protection des données) sur le Blog de sécuritéAWS .

À des fins de protection des données, nous vous recommandons de protéger les Compte AWS informations d'identification et de configurer les utilisateurs individuels avec AWS IAM Identity Center ou Gestion des identités et des accès AWS (IAM). Ainsi, chaque utilisateur se voit attribuer uniquement les autorisations nécessaires pour exécuter ses tâches. Nous vous recommandons également de sécuriser vos données comme indiqué ci-dessous :

- Utilisez l'authentification multifactorielle (MFA) avec chaque compte.
- SSL/TLS À utiliser pour communiquer avec AWS les ressources. Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Configurez l'API et la journalisation de l'activité des utilisateurs avec AWS CloudTrail. Pour plus d'informations sur l'utilisation des CloudTrail sentiers pour capturer AWS des activités, consultez la section Utilisation des CloudTrail sentiers dans le guide de AWS CloudTrail l'utilisateur.
- Utilisez des solutions de AWS chiffrement, ainsi que tous les contrôles de sécurité par défaut qu'ils contiennent Services AWS.
- Utilisez des services de sécurité gérés avancés tels qu'Amazon Macie, qui contribuent à la découverte et à la sécurisation des données sensibles stockées dans Amazon S3.
- Si vous avez besoin de modules cryptographiques validés par la norme FIPS 140-3 pour accéder AWS via une interface de ligne de commande ou une API, utilisez un point de terminaison FIPS. Pour plus d'informations sur les points de terminaison FIPS disponibles, consultez Norme FIPS (Federal Information Processing Standard) 140-3.

Nous vous recommandons fortement de ne jamais placer d'informations confidentielles ou sensibles, telles que les adresses e-mail de vos clients, dans des balises ou des champs de texte libre tels que le champ Nom. Cela inclut lorsque vous travaillez avec Elastic Beanstalk Services AWS ou autre à

l'aide de la console, de l'API ou. AWS CLI AWS SDKs Toutes les données que vous entrez dans des balises ou des champs de texte de forme libre utilisés pour les noms peuvent être utilisées à des fins de facturation ou dans les journaux de diagnostic. Si vous fournissez une adresse URL à un serveur externe, nous vous recommandons fortement de ne pas inclure d'informations d'identification dans l'adresse URL permettant de valider votre demande adressée à ce serveur.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à AWS Elastic Beanstalk sécurité.

Rubriques

- Protection des données à l'aide du chiffrement
- Confidentialité du trafic inter-réseaux

Protection des données à l'aide du chiffrement

Vous pouvez utiliser différentes formes de chiffrement des données pour protéger vos données Elastic Beanstalk. La protection des données fait référence à la protection des données en transit (lorsqu'elles voyagent vers et depuis Elastic Beanstalk) et au repos (lorsqu'elles sont stockées AWS dans des centres de données).

Chiffrement en transit

Vous pouvez obtenir la protection des données en transit de deux manières : chiffrer la connexion à l'aide du protocole SSL (Secure Sockets Layer) ou utiliser le chiffrement côté client (où l'objet est chiffré avant d'être envoyé). Les deux méthodes sont valides pour protéger les données de votre application. Pour sécuriser la connexion, chiffrez-la à l'aide de SSL chaque fois que votre application, ses développeurs et administrateurs, ainsi que ses utilisateurs finaux, envoient ou reçoivent des objets. Pour plus d'informations sur le chiffrement du trafic web vers et à partir de votre application, consultez the section called “HTTPS”.

Le chiffrement côté client n'est pas une méthode valide pour protéger votre code source dans les versions d'application et les bundles de fichiers source que vous chargez. Elastic Beanstalk ayant besoin d'accéder à ces objets, ils ne peuvent pas être chiffrés. Par conséquent, veillez à sécuriser la connexion entre votre environnement de développement ou de déploiement et Elastic Beanstalk.

Chiffrement au repos

Pour protéger les données de votre application au repos, découvrez la protection des données dans le service de stockage utilisé par votre application. Par exemple, consultez Protection des

données dans Amazon RDS dans le Guide de l'utilisateur Amazon RDS, Protection des données dans Amazon S3 dans le Guide de l'utilisateur Amazon Simple Storage Service, ou Chiffrement des données et métadonnées dans EFS dans le Guide de l'utilisateur Amazon Elastic File System.

Elastic Beanstalk stocke divers objets dans un bucket Amazon Simple Storage Service (Amazon S3) chiffré qu'il crée AWS pour chaque région dans laquelle vous créez des environnements. Dans la mesure où Elastic Beanstalk conserve le chiffrement par défaut fourni par Amazon S3, il crée des buckets Amazon S3 chiffrés. Pour en savoir plus, consultez the section called “Amazon S3”. Vous fournissez certains des objets stockés, comme les versions d'application et les bundles de fichiers source par exemple, puis les envoyez à Elastic Beanstalk. Elastic Beanstalk génère d'autres objets, comme des fichiers journaux par exemple. Outre les données stockées par Elastic Beanstalk, votre application and/or peut transférer des données de magasin dans le cadre de son fonctionnement.

Pour protéger les données stockées sur les volumes Amazon Elastic Block Store (Amazon EBS) attachés aux instances de votre environnement, activez le chiffrement Amazon EBS par défaut dans votre AWS compte et votre région. Lorsque cette option est activée, tous les nouveaux volumes Amazon EBS et leurs instantanés sont automatiquement chiffrés à l'aide AWS Key Management Service de clés. Pour plus d'informations, consultez la section Chiffrement par défaut dans le guide de l'utilisateur Amazon EBS.

Pour en savoir plus sur la protection des données, consultez le billet de blog Modèle de responsabilité partagée AWS et RGPD sur le Blog sur la sécurité d’AWS .

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à AWS Elastic Beanstalk sécurité.

Confidentialité du trafic inter-réseaux

Vous pouvez utiliser Amazon Virtual Private Cloud (Amazon VPC) pour créer des limites entre les ressources de votre application Elastic Beanstalk et contrôler le trafic entre celles-ci, votre réseau sur site et Internet. Pour plus d'informations, consultez the section called “Amazon VPC”.

Pour de plus amples informations sur la sécurité dans Amazon VPC, veuillez consulter Sécurité dans le Guide de l'utilisateur Amazon VPC.

Pour en savoir plus sur la protection des données, consultez le billet de blog Modèle de responsabilité partagée AWS et RGPD sur le Blog sur la sécurité d’AWS .

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à AWS Elastic Beanstalk sécurité.

Gestion des identités et des accès pour Elastic Beanstalk

Gestion des identités et des accès AWS (IAM) est un AWS service qui permet à un administrateur de contrôler en toute sécurité l'accès aux AWS ressources. Des administrateurs IAM contrôlent les personnes qui s'authentifient (sont connectées) et sont autorisées (disposent d'autorisations) à utiliser des ressources AWS Elastic Beanstalk . IAM est un AWS service que vous pouvez utiliser sans frais supplémentaires.

Pour de plus amples informations sur l'utilisation d'IAM, veuillez consulter Utilisation d'Elastic Beanstalk avec Gestion des identités et des accès AWS.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à AWS Elastic Beanstalk sécurité.

AWS politiques gérées pour AWS Elastic Beanstalk

Une politique AWS gérée est une politique autonome créée et administrée par AWS. AWS les politiques gérées sont conçues pour fournir des autorisations pour de nombreux cas d'utilisation courants afin que vous puissiez commencer à attribuer des autorisations aux utilisateurs, aux groupes et aux rôles.

N'oubliez pas que les politiques AWS gérées peuvent ne pas accorder d'autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles sont accessibles à tous les AWS clients. Nous vous recommandons de réduire encore les autorisations en définissant des politiques gérées par le client qui sont propres à vos cas d'utilisation.

Vous ne pouvez pas modifier les autorisations définies dans les politiques AWS gérées. Si les autorisations définies dans une politique AWS gérée sont AWS mises à jour, la mise à jour affecte toutes les identités principales (utilisateurs, groupes et rôles) auxquelles la politique est attachée. AWS est le plus susceptible de mettre à jour une politique AWS gérée lorsqu'une nouvelle Service AWS est lancée ou lorsque de nouvelles opérations d'API sont disponibles pour les services existants.

Pour plus d'informations, consultez Politiques gérées par AWS dans le Guide de l'utilisateur IAM.

Elastic AWS Beanstalk met à jour les politiques gérées

Consultez les détails des mises à jour apportées aux politiques AWS gérées pour Elastic Beanstalk depuis le 1er mars 2021.

Pour consulter la source JSON d'une stratégie gérée spécifique, consultez le Guide de référence des politiques AWS gérées.

Modifier	Description	Date
AWSElasticBeanstalkManagedUpdatesServiceRolePolicy—Politique existante mise à jour	Cette politique a été mise à jour pour permettre à Elastic Beanstalk d'effectuer des mises à jour gérées lorsqu'elle est activée. Propagation des balises vers les modèles de lancement Pour plus d'informations, veuillez consulter Stratégies de rôles de service gérées.	17 mars 2026
AWSElasticBeanstalkWebTier—Politique existante mise à jour	Cette politique a été mise à jour pour permettre à Elastic Beanstalk d'utiliser Amazon Bedrock pour l'analyse de l'environnement basée sur l'IA. Pour de plus amples informations, veuillez consulter Gestion des profils d'instance Elastic Beanstalk.	11 mars 2026
AWSElasticBeanstalkWorkerTier—Politique existante mise à jour	Cette politique a été mise à jour pour permettre à Elastic Beanstalk d'utiliser Amazon Bedrock pour l'analyse de l'environnement basée sur l'IA.	11 mars 2026

Modifier	Description	Date
	Pour de plus amples informations, veuillez consulter Gestion des profils d'instance Elastic Beanstalk.	
AWSElasticBeanstalkMulticontainerDocker—Politique existante mise à jour	Cette politique a été mise à jour pour permettre à Elastic Beanstalk d'utiliser Amazon Bedrock pour l'analyse de l'environnement basée sur l'IA. Pour de plus amples informations, veuillez consulter Gestion des profils d'instance Elastic Beanstalk.	11 mars 2026
AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy—Politique existante mise à jour	Cette politique a été mise à jour pour permettre à Elastic Beanstalk d'effectuer des Propagation des balises vers les modèles de lancement mises à jour gérées lorsqu'elle est activée pour une instance unique. Pour plus d'informations, veuillez consulter Stratégies de rôles de service gérées.	27 janvier 2026

Modifier	Description	Date
AdministratorAccess-AWSElastic Beanstalk — Politique existante mise à jour	Cette politique a été mise à jour pour permettre à Elastic Beanstalk de configurer les paramètres de blocage d'accès public et les contrôles de propriété des compartiments sur les compartiments S3. Pour de plus amples informations, veuillez consulter Gestion des stratégies utilisateur Elastic Beanstalk.	12 novembre 2025
AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy—Politique existante mise à jour	Cette politique a été mise à jour pour permettre à Elastic Beanstalk d'effectuer des mises à jour gérées lorsqu'elle est activée. Propagation des balises vers les modèles de lancement Pour plus d'informations, veuillez consulter Stratégies de rôles de service gérées.	27 février 2025

Modifier	Description	Date
AdministratorAccess-AWSElastic Beanstalk — Politique existante mise à jour	Cette politique a été mise à jour pour remplacer l'StringLikeopérateur par l'ArnLikeopérateur chargé d'évaluer les clés de type ARN dans le bloc de condition s. `iam:PolicyArn` Cela permet une application plus sûre. Pour de plus amples informations, veuillez consulter Gestion des stratégies utilisateur Elastic Beanstalk.	11 décembre 2024

Modifier	Description	Date
Les politiques suivantes ont été mises à jour : • AWSElasticBeanstalkInternalMaintenanceRolePolicy • AWSElasticBeanstalkMaintenance • AWSElasticBeanstalkManagedUpdatesInternalServiceRolePolicy • AWSElasticBeanstalkManagedUpdatesServiceRolePolicy • AWSElasticBeanstalkRoleCore	Ces politiques ont été mises à jour pour permettre à Elastic Beanstalk d'ajouter ou de supprimer des balises lors de la création ou de la mise à AWS CloudFormation jour d'une pile ou d'un ensemble de modifications. Pour plus d'informations sur `AWSElasticBeanstalkManagedUpdatesServiceRolePolicy`, consultez Autorisations des rôles liés à un service pour Elastic Beanstalk. Pour plus d'informations sur `AWSElasticBeanstalkRoleCore`, consultez Politiques d'intégration à d'autres services.	30 avril 2024

Modifier	Description	Date
AWSElasticBeanstalkService—Politique existante mise à jour	Cette stratégie a été mise à jour pour permettre à Elastic Beanstalk de baliser les ressources lors de leur création pour Elastic Load Balancing, groupes Auto Scaling (ASG) et Amazon ECS. **Note** Cette stratégie a été précédemment remplacée par `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy` . Cette stratégie ne peux plus être associée à de nouveaux utilisateurs, groupes ou rôles IAM, mais il est possible qu'elle soit toujours associée à des utilisateurs, groupes ou rôles IAM existants.	10 mai 2023

Modifier	Description	Date
	Pour plus d'informations, veuillez consulter Stratégies de rôles de service gérées.	
AWSElasticBeanstalkMulticontainerDocker—Politique existante mise à jour	Cette stratégie a été mise à jour pour permettre à Elastic Beanstalk de baliser les ressources lors de leur création pour Amazon ECS. Pour de plus amples informations, veuillez consulter Gestion des profils d'instance Elastic Beanstalk.	23 mars 2023
AWSElasticBeanstalkRoleECS —Politique existante mise à jour	Cette stratégie a été mise à jour pour permettre à Elastic Beanstalk de baliser les ressources lors de leur création pour Amazon ECS. Pour de plus amples informations, veuillez consulter Politiques d'intégration à d'autres services.	23 mars 2023
AdministratorAccess-AWSElastic Beanstalk — Politique existante mise à jour	Cette stratégie a été mise à jour pour permettre à Elastic Beanstalk de baliser les ressources lors de leur création pour Amazon ECS. Pour de plus amples informations, veuillez consulter Gestion des stratégies utilisateur Elastic Beanstalk.	23 mars 2023

Modifier	Description	Date
AWSElasticBeanstalkManagedUpdatesServiceRolePolicy —Politique existante mise à jour	Cette politique a été mise à jour pour permettre à Elastic Beanstalk d'ajouter des balises aux ressources Amazon ECS lorsqu'il les crée. Pour de plus amples informations, veuillez consulter Autorisations des rôles liés à un service pour Elastic Beanstalk.	23 mars 2023
AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy—Politique existante mise à jour	Cette politique a été mise à jour pour permettre à Elastic Beanstalk d'ajouter des balises aux ressources Amazon ECS lorsqu'il les crée. Pour plus d'informations, veuillez consulter Stratégies de rôles de service gérées.	23 mars 2023
AWSElasticBeanstalkManagedUpdatesServiceRolePolicy—Politique existante mise à jour	Cette politique a été mise à jour pour permettre à Elastic Beanstalk d'ajouter des balises aux groupes Auto Scaling lorsqu'il les crée. Pour de plus amples informations, veuillez consulter Rôle lié à un service de mises à jour gérées.	27 janvier 2023

Modifier	Description	Date
AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy—Politique existante mise à jour	Cette politique a été mise à jour pour permettre à Elastic Beanstalk d'ajouter des balises lors de la création d'un groupe Auto Scaling (ASG). Pour plus d'informations, veuillez consulter Stratégies de rôles de service gérées.	23 janvier 2023
AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy—Politique existante mise à jour	Cette politique a été mise à jour pour permettre à Elastic Beanstalk d'ajouter des balises lors de la création d'un équilibreur de charge élastique (ELB). Pour plus d'informations, veuillez consulter Stratégies de rôles de service gérées.	21 décembre 2022

Modifier	Description	Date
AWSElasticBeanstalkManagedUpdatesServiceRolePolicy—Politique existante mise à jour	Les autorisations ont été ajoutées à cette politique pour permettre à Elastic Beanstalk d'effectuer les opérations ci-dessous lors des mises à jour gérées : • Créer et supprimer des modèles de lancement et des versions de modèles. • Démarrer les instances Amazon EC2 avec des modèles de lancement. • Si un Amazon RDS est présent, récupérez une liste des moteurs de base de données et des informations sur les instances RDS fournies. Pour de plus amples informations, veuillez consulter Rôle lié à un service de mises à jour gérées.	23 août 2022

Modifier	Description	Date
AWSElasticBeanstalkReadOnlyAccess : obsolète GovCloud (États-Unis) Région AWS	Cette stratégie a été remplacée par `AWSElasticBeanstalkReadOnly` . Cette politique sera progressivement supprimée aux GovCloud États-Unis. Région AWS Lorsque cette stratégie est progressivement supprimée , elle ne pourra plus être associée à de nouveaux utilisateurs, groupes ou rôles IAM après le 17 juin 2021. Pour plus d'informations, veuillez consulter Stratégies utilisateur.	17 juin 2021
AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy—Politique existante mise à jour	Cette stratégie a été mise à jour pour permettre à Elastic Beanstalk de lire les attributs des zones de disponibilité EC2. Elle permet à Elastic Beanstalk de fournir une validation plus efficace de votre sélection de type d'instance dans les zones de disponibilité. Pour plus d'informations, veuillez consulter Stratégies de rôles de service gérées.	16 juin 2021

Modifier	Description	Date
AWSElasticBeanstalkFullAccess : obsolète GovCloud (États-Unis) Région AWS	Cette stratégie a été remplacée par `AdministratorAccess-AWSElasticBeanstalk` . Cette politique sera progressivement supprimée aux GovCloud États-Unis. Région AWS Lorsque cette stratégie est progressivement supprimée , elle ne pourra plus être associée à de nouveaux utilisateurs, groupes ou rôles IAM après le 10 juin 2021. Pour plus d'informations, veuillez consulter Stratégies utilisateur.	10 JUIN 2021

Modifier	Description	Date
Les politiques gérées suivantes ont été déconseillées dans toutes les régions de Chine Région AWS : • AWSElasticBeanstalkFullAccess • AWSElasticBeanstalkReadOnlyAccess	La stratégie `AWSElasticBeanstalkFullAccess` a été remplacée par `AdministratorAccess-AWSElasticBeanstalk` . La stratégie `AWSElasticBeanstalkReadOnlyAccess` a été remplacée par `AWSElasticBeanstalkReadOnly` . Ces politiques ont été progressivement supprimées dans toutes les régions de Chine Région AWS. Ces stratégies ne pourront plus être associées à de nouveaux utilisateurs, groupes ou rôles IAM après le 3 juin 2021. Pour plus d'informations, veuillez consulter Stratégies utilisateur.	3 juin 2021

Modifier	Description	Date
AWSElasticBeanstalkService : obsolète	Cette politique a été remplacée par `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy` . Cette stratégie est progressivement supprimée et ne peux plus être associée à de nouveaux utilisateurs, groupes ou rôles IAM. Pour plus d'informations, veuillez consulter Stratégies de rôles de service gérées.	juin 2021 à janvier 2022

Modifier	Description	Date
Les politiques gérées suivantes ont été déconseillées dans tous les Région AWS pays, à l'exception de la Chine et GovCloud (États-Unis) : • AWSElasticBeanstalkFullAccess • AWSElasticBeanstalkReadOnlyAccess	La stratégie `AWSElasticBeanstalkFullAccess` a été remplacée par `AdministratorAccess-AWSElasticBeanstalk` . La stratégie `AWSElasticBeanstalkReadOnlyAccess` a été remplacée par `AWSElasticBeanstalkReadOnly` . Ces politiques ont été progressivement supprimées dans toutes les Région AWS années, à l'exception de la Chine et GovCloud (États-Unis). Ces stratégies ne pourront plus être associées à de nouveaux utilisateurs, groupes ou rôles IAM après le 16 avril 2021. Pour plus d'informations, veuillez consulter Stratégies utilisateur.	16 avril 2021

Modifier	Description	Date
Les stratégies gérées suivantes ont été mises à jour : • AdministratorAccess-AWSElastic Haricot • AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy	Ces deux politiques prennent désormais en charge PassRole les autorisations en Chine Régions AWS. Pour plus d'informations sur `AdministratorAccess-AWSElasticBeanstalk`, veuillez consulter Stratégies utilisateur. Pour plus d'informations sur `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy`, veuillez consulter Stratégies de rôles de service gérées.	9 mars 2021
AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy : nouvelle politique	Elastic Beanstalk a ajouté une nouvelle stratégie en remplacement de la stratégie gérée `AWSElasticBeanstalkService`. Cette nouvelle stratégie gérée améliore la sécurité de vos ressources en appliquant un jeu d'autorisations plus restrictif. Pour plus d'informations, veuillez consulter Stratégies de rôles de service gérées.	3 mars 2021

Modifier	Description	Date
Elastic Beanstalk a commencé à assurer le suivi des modifications	Elastic Beanstalk a commencé à suivre les modifications apportées aux politiques gérées. AWS	1er mars 2021

Journalisation et surveillance dans Elastic Beanstalk

AWS fournit plusieurs outils pour surveiller vos ressources Elastic Beanstalk et répondre aux incidents potentiels. La surveillance est importante pour maintenir la fiabilité, la disponibilité et les performances de AWS Elastic Beanstalk vos AWS solutions. Vous devez collecter des données de surveillance provenant de toutes les parties de votre AWS solution afin de pouvoir corriger plus facilement une défaillance multipoint, le cas échéant.

Pour de plus amples informations sur la surveillance, veuillez consulter Surveillance des environnements dans Elastic Beanstalk.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à AWS Elastic Beanstalk sécurité.

Création de rapports d'intégrité améliorée

La création de rapports d'intégrité améliorée est une fonctionnalité que vous pouvez activer sur votre environnement pour autoriser Elastic Beanstalk à collecter des informations complémentaires sur les ressources de votre environnement. Elastic Beanstalk analyse les informations recueillies pour fournir une meilleure image de l'intégrité globale de l'environnement et aider à l'identification des problèmes pouvant entraîner une indisponibilité de votre application. Pour de plus amples informations, veuillez consulter Rapports et surveillance de l'état de santé améliorés dans Elastic Beanstalk.

Journaux des EC2 instances Amazon

Les EC2 instances Amazon de votre environnement Elastic Beanstalk génèrent des journaux que vous pouvez consulter pour résoudre les problèmes liés à votre application ou à vos fichiers de configuration. Les journaux sont créés par le serveur Web, le serveur d'applications, les scripts de la plateforme Elastic Beanstalk et sont stockés localement sur CloudFormation des instances individuelles. Vous pouvez les récupérer facilement avec la console de gestion d'environnement ou

l'interface de ligne de commande EB. Vous pouvez également configurer votre environnement pour diffuser les journaux sur Amazon CloudWatch Logs en temps réel. Pour de plus amples informations, veuillez consulter Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk.

Notifications de l'environnement

Vous pouvez configurer votre environnement Elastic Beanstalk pour utiliser Amazon Simple Notification Service (Amazon SNS) et vous informer des événements importants qui affectent votre application. Spécifiez une adresse e-mail pendant ou après la création de l'environnement pour recevoir des e-mails en AWS cas d'erreur ou de modification de l'état de santé de votre environnement. Pour de plus amples informations, veuillez consulter Notifications d'environnement Elastic Beanstalk avec Amazon SNS.

CloudWatch Alarmes Amazon

À l'aide d' CloudWatch alarmes, vous observez une seule métrique sur une période que vous spécifiez. Si la métrique dépasse un seuil donné, une notification est envoyée à une rubrique ou AWS Auto Scaling à une politique Amazon SNS. CloudWatch les alarmes n'appellent pas d'actions car elles sont dans un état particulier. Au lieu de cela, les alarmes appellent des actions lorsque l'état a changé et a été maintenu pendant un certain nombre de périodes. Pour de plus amples informations, veuillez consulter Utilisation d'Elastic Beanstalk avec Amazon CloudWatch.

AWS CloudTrail journaux

CloudTrail fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans Elastic Beanstalk. À l'aide des informations collectées par CloudTrail, vous pouvez déterminer la demande envoyée à Elastic Beanstalk, l'adresse IP à partir de laquelle la demande a été faite, l'auteur de la demande, la date à laquelle elle a été faite, ainsi que des informations supplémentaires. Pour de plus amples informations, veuillez consulter Journalisation des appels d'API Elastic Beanstalk avec AWS CloudTrail.

AWS X-Ray débogage

X-Ray est un AWS service qui collecte des données sur les demandes traitées par votre application et les utilise pour créer une carte des services que vous pouvez utiliser pour identifier les problèmes liés à votre application et les opportunités d'optimisation. Vous pouvez utiliser la AWS Elastic Beanstalk console ou un fichier de configuration pour exécuter le daemon X-Ray sur les instances de

votre environnement. Pour de plus amples informations, veuillez consulter Configuration du AWS X-Ray débogage.

Validation de la conformité pour Elastic Beanstalk

La sécurité et la conformité de AWS Elastic Beanstalk sont évaluées par des auditeurs tiers dans le cadre de multiples programmes de AWS conformité. Il s'agit notamment du SOC, du PCI, du FedRAMP, de l'HIPAA et d'autres. AWS fournit une liste fréquemment mise à jour des AWS services concernés par des programmes de conformité spécifiques sur la page AWS Services in Scope by Compliance Program.

Les rapports d'audit tiers peuvent être téléchargés à l'aide de AWS Artifact. Pour plus d'informations, consultez la section Téléchargement de rapports dans AWS Artifact.

Pour plus d'informations sur les programmes de AWS conformité, consultez AWS la section Programmes de conformité.

Votre responsabilité de conformité lors de l'utilisation d'Elastic Beanstalk est déterminée par la sensibilité de vos données, les objectifs de conformité de votre organisation, ainsi que par la législation et la réglementation applicables. Si votre utilisation d'Elastic Beanstalk est soumise au respect de normes telles que HIPAA, PCI ou FedRAMP, fournit des ressources pour vous aider à : AWS

- Guides de démarrage rapide sur la sécurité et la conformité : guides de déploiement qui abordent les considérations architecturales et indiquent les étapes à suivre pour déployer des environnements de référence axés sur la sécurité et sur la conformité sur. AWS
- Architecture axée sur la sécurité et la conformité HIPAA sur Amazon Web Services : livre blanc qui décrit comment les entreprises peuvent créer des applications AWS conformes à la loi HIPAA.
- AWS Ressources de conformité : collection de classeurs et de guides de conformité susceptibles de s'appliquer à votre secteur d'activité et à votre situation géographique.
- AWS Config – Service qui permet d'évaluer comment les configurations de vos ressources se conforment aux pratiques internes, aux normes et aux directives industrielles.
- AWS Security Hub CSPM— Une vue complète de votre état de sécurité interne AWS qui vous permet de vérifier votre conformité aux normes et aux meilleures pratiques du secteur de la sécurité.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à AWS Elastic Beanstalk sécurité.

Résilience dans Elastic Beanstalk

AWS Elastic Beanstalk gère et automatise l'utilisation de l'infrastructure AWS mondiale en votre nom. Lorsque vous utilisez Elastic Beanstalk, vous bénéficiez des mécanismes de disponibilité et de tolérance aux pannes qu'il propose. AWS

L'infrastructure AWS mondiale est construite autour des AWS régions et des zones de disponibilité.

AWS Les régions fournissent plusieurs zones de disponibilité physiquement séparées et isolées, connectées par un réseau à faible latence, à haut débit et hautement redondant.

Avec les zones de disponibilité, vous pouvez concevoir et exploiter des applications et des bases de données qui basculent automatiquement d'une zone de disponibilité à l'autre sans interruption. Les zones de disponibilité sont plus hautement disponibles, tolérantes aux pannes et évolutives que les infrastructures traditionnelles à un ou plusieurs centres de données.

Pour plus d'informations sur AWS les régions et les zones de disponibilité, consultez la section Infrastructure AWS mondiale.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à AWS Elastic Beanstalk sécurité.

Sécurité de l'infrastructure dans Elastic Beanstalk

En tant que service géré, AWS Elastic Beanstalk il est protégé par les procédures de sécurité du réseau AWS mondial décrites dans notre site Web consacré aux meilleures pratiques en matière de sécurité, d'identité et de conformité.

Vous utilisez des appels d'API AWS publiés pour accéder à Elastic Beanstalk via le réseau. Les clients doivent prendre en charge le protocole TLS (Transport Layer Security) 1.2 ou version ultérieure. Les clients doivent également prendre en charge les suites de chiffrement PFS (Perfect Forward Secrecy) comme Ephemeral Diffie-Hellman (DHE) ou Elliptic Curve Ephemeral Diffie-Hellman (ECDHE) La plupart des plateformes modernes prennent en charge ces modes.

En outre, les demandes doivent être signées à l'aide d'un ID de clé d'accès et d'une clé d'accès secrète associée à un principal IAM. Vous pouvez également utiliser AWS Security Token Service

(AWS STS) pour générer des informations d'identification de sécurité temporaires et signer les demandes.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à AWS Elastic Beanstalk sécurité.

Configuration et analyse des vulnérabilités dans Elastic Beanstalk

AWS et nos clients partagent la responsabilité d'atteindre un niveau élevé de sécurité et de conformité des composants logiciels. AWS Elastic Beanstalk vous aide à mettre en œuvre votre version du modèle de responsabilité partagée en fournissant une fonctionnalité de mises à jour gérées. Cette fonction applique automatiquement les mises à jour correctives et mineures correspondant à une version de plateforme prise en charge par Elastic Beanstalk.

Pour de plus amples informations, veuillez consulter Modèle de responsabilité partagée pour la maintenance de la plateforme Elastic Beanstalk.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à AWS Elastic Beanstalk sécurité.

Bonnes pratiques de sécurité pour Elastic Beanstalk

AWS Elastic Beanstalk fournit plusieurs fonctionnalités de sécurité à prendre en compte lors de l'élaboration et de la mise en œuvre de vos propres politiques de sécurité. Les bonnes pratiques suivantes doivent être considérées comme des instructions générales et ne représentent pas une solution de sécurité complète. Étant donné que ces bonnes pratiques peuvent ne pas être appropriées ou suffisantes pour votre environnement, considérez-les comme des considérations utiles et non comme des recommandations.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à AWS Elastic Beanstalk sécurité.

Bonnes pratiques de sécurité préventive

Les contrôles de sécurité préventifs tentent d'éviter les incidents avant qu'ils ne se produisent.

Implémentation d'un accès sur la base du moindre privilège

Elastic Gestion des identités et des accès AWS Beanstalk fournit des politiques gérées (IAM) pour les profils d'instance, les rôles de service et les utilisateurs IAM. Ces stratégies gérées spécifient toutes

les autorisations qui peuvent être nécessaires au bon fonctionnement de votre environnement et de votre application.

Votre application peut ne pas exiger toutes les autorisations de nos stratégies gérées. Vous pouvez les personnaliser et accorder uniquement les autorisations requises pour que les instances de votre environnement, le service Elastic Beanstalk et vos utilisateurs puissent effectuer leurs tâches. C'est particulièrement pertinent pour les stratégies utilisateur, où différents rôles utilisateur peuvent avoir des besoins différents en matière d'autorisations. L'implémentation d'un accès sur la base du moindre privilège est fondamentale pour réduire les risques en matière de sécurité et l'impact que pourraient avoir des d'erreurs ou des actes de malveillance.

Protégez les données sensibles des applications

Lorsque votre application doit accéder à des informations sensibles telles que des informations d'identification, des clés d'API ou des données de configuration, suivez les pratiques suivantes pour garantir la sécurité :

- Récupérez les données sensibles directement à partir de AWS Secrets Manager AWS Systems Manager Parameter Store en utilisant leur code respectif SDKs ou APIs dans le code de votre application. Cela constitue le moyen le plus sûr et le plus flexible d'accéder aux informations sensibles.
- Si vous transmettez des données sensibles depuis AWS Secrets Manager ou AWS Systems Manager Parameter Store en tant que variables d'environnement (voir Récupérez les secrets des variables d'environnement), limitez soigneusement l'accès aux paires de EC2 clés et configurez les rôles IAM appropriés avec des autorisations de moindre privilège pour vos instances.
- N'imprimez, n'enregistrez ou n'exposez jamais de données sensibles dans le code de votre application, car ces valeurs pourraient se retrouver dans des fichiers journaux ou des messages d'erreur visibles par des utilisateurs non autorisés.

Mise à jour régulière de vos plateformes

Elastic Beanstalk publie régulièrement de nouvelles versions de plateforme pour mettre à jour toutes ses plateformes. Les nouvelles versions de plateforme fournissent des mises à jour du système d'exploitation, de l'environnement d'exécution, du serveur d'applications et du serveur web, ainsi que des mises à jour des composants Elastic Beanstalk. Bon nombre de ces mises à jour de plateforme incluent des correctifs de sécurité importants. Assurez-vous que vos environnements Elastic Beanstalk s'exécutent sur une version de plateforme prise en charge (généralement la

dernière version de votre plateforme). Pour en savoir plus, consultez Mise à jour de la version de la plateforme de votre environnement Elastic Beanstalk.

Le moyen le plus simple de maintenir à jour la plateforme de votre environnement consiste à configurer l'environnement pour utiliser les mises à jour gérées de la plateforme.

Appliquer IMDSv2 sur les instances de l'environnement

Les instances Amazon Elastic Compute Cloud (Amazon EC2) de vos environnements Elastic Beanstalk utilisent le service de métadonnées d'instance (IMDS), un composant intégré à l'instance, pour accéder en toute sécurité aux métadonnées des instances. L'IMDS prend en charge deux méthodes pour accéder aux données : IMDSv1 et IMDSv2. IMDSv2 utilise des requêtes orientées session et atténue plusieurs types de vulnérabilités susceptibles d'être utilisées pour tenter d'accéder à l'IMDS. Pour en savoir plus sur les avantages de IMDSv2, consultez la section Améliorations visant à renforcer la défense du service de métadonnées d' EC2 instance.

IMDSv2 est plus sécurisé, c'est donc une bonne idée d'imposer l'utilisation de IMDSv2 sur vos instances. Pour appliquer IMDSv2, assurez-vous que tous les composants de votre application sont pris en charge IMDSv2, puis désactivez-les IMDSv1. Pour de plus amples informations, veuillez consulter the section called “IMDS”.

Bonnes pratiques de sécurité de détection

Les contrôles de sécurité de détection identifient les violations de sécurité après qu'elles se sont produites. Ils peuvent vous aider à détecter une menace ou un incident de sécurité potentiel.

Mise en œuvre de la surveillance

La surveillance joue un rôle important dans le maintien de la fiabilité, de la sécurité, de la disponibilité et des performances de vos solutions Elastic Beanstalk. AWS fournit plusieurs outils et services pour vous aider à surveiller vos AWS services.

Voici quelques exemples d'éléments à surveiller :

- CloudWatch Métriques Amazon pour Elastic Beanstalk — Définissez des alarmes pour les mesures clés d'Elastic Beanstalk et pour les mesures personnalisées de votre application. Pour en savoir plus, consultez Utilisation d'Elastic Beanstalk avec Amazon CloudWatch.
- AWS CloudTrail entrées — Suivez les actions susceptibles d'avoir un impact sur la disponibilité, comme `UpdateEnvironment` ou`TerminateEnvironment`. Pour en savoir plus, consultez Journalisation des appels d'API Elastic Beanstalk avec AWS CloudTrail.

Activer AWS Config

AWS Config fournit une vue détaillée de la configuration des AWS ressources de votre compte. Vous pouvez voir comment les ressources sont liées, obtenir un historique des changements de configuration, et voir comment les configurations et les relations changent au fil du temps.

Vous pouvez l'utiliser AWS Config pour définir des règles qui évaluent les configurations des ressources en termes de conformité des données. AWS Config les règles représentent les paramètres de configuration idéaux pour vos ressources Elastic Beanstalk. Si une ressource enfreint une règle et est signalée comme non conforme, vous AWS Config pouvez être alertée via une rubrique Amazon Simple Notification Service (Amazon SNS). Pour en savoir plus, consultez Trouver et suivre les ressources Elastic Beanstalk avec AWS Config.

Rôles d'Elastic Beanstalk Service, profils d'instance et politiques utilisateur

Les rôles sont des entités que vous créez avec Gestion des identités et des accès AWS (IAM) pour appliquer des autorisations. Certains rôles sont nécessaires au bon fonctionnement de votre environnement Elastic Beanstalk. Vous avez également la possibilité de créer vos propres politiques et rôles personnalisés que vous pouvez attribuer à des utilisateurs ou à des groupes.

Rôles obligatoires pour votre environnement Elastic Beanstalk

Lorsque vous créez un environnement, il vous AWS Elastic Beanstalk invite à fournir les rôles Gestion des identités et des accès AWS (IAM) suivants :

- Rôle de service : Elastic Beanstalk assume un rôle de service pour Services AWS en utiliser d'autres en votre nom.
- Profil d'instance Elastic Beanstalk applique un profil d'instance aux instances Amazon EC2 de votre environnement. Cette action leur permet d'effectuer les tâches requises, telles que la récupération d'informations depuis Amazon Simple Storage Service (Amazon S3) et le téléchargement de journaux vers S3.

Création du rôle de service et du rôle de profil d'instance EC2

Si votre AWS compte ne possède pas de profil d'instance EC2 ni de rôle de service, vous devez en créer un à l'aide du service IAM. Vous pouvez ensuite attribuer le profil d'instance EC2 et le rôle de service aux nouveaux environnements que vous créez. L'assistant de création d'un environnement vous guide vers le service IAM, afin que vous puissiez créer ces rôles avec les autorisations requises.

Politiques et rôles facultatifs pour gérer votre environnement Elastic Beanstalk

Vous pouvez éventuellement créer des politiques utilisateur et les appliquer aux utilisateurs et aux groupes IAM de votre compte. Cela permet aux utilisateurs de créer et de gérer des applications et des environnements Elastic Beanstalk. Vous pouvez également attribuer des politiques gérées par Elastic Beanstalk pour un accès complet et un accès en lecture seule aux utilisateurs ou aux

groupes. Pour plus d'informations sur ces stratégies, consultez the section called “Stratégies utilisateur”.

Vous pouvez créer vos propres profils d'instance et stratégies utilisateur pour des scénarios avancés. Si vos instances ont besoin d'accéder à des services qui ne sont pas inclus dans les politiques par défaut, vous pouvez créer une politique ou ajouter des politiques supplémentaires à celle par défaut. Si la politique gérée est trop permissive pour vos besoins, vous pouvez également créer des politiques d'utilisateur plus restrictives. Pour plus d'informations sur AWS les autorisations, consultez le guide de l'utilisateur IAM.

Rôle de service Elastic Beanstalk

Un rôle de service correspond au rôle IAM exercé par Elastic Beanstalk lorsqu'il appelle d'autres services en votre nom. Par exemple, Elastic Beanstalk utilise un rôle de service lorsqu'il appelle Amazon Elastic Compute Cloud (Amazon EC2), Elastic Load Balancing et Amazon EC2 Auto Scaling pour recueillir des informations. APIs La fonction du service utilisée par Elastic Beanstalk est celle que vous avez spécifiée lorsque vous créez l'environnement Elastic Beanstalk.

Il y a deux politiques gérées attachée à la fonction du service. Ces politiques fournissent les autorisations qui permettent à Elastic Beanstalk d'accéder aux ressources nécessaires pour créer AWS et gérer vos environnements. Elastic Beanstalk fournit une politique gérée pour la surveillance améliorée de l'état, un support Amazon SQS de niveau de travail et une autre qui comporte les autorisations supplémentaires requises pour les mises à jour gérées de la plateforme.

`AWSElasticBeanstalkEnhancedHealth`

Cette politique accorde des autorisations pour qu'Elastic Beanstalk surveille l'état de l'instance et de l'environnement. Cette stratégie inclut également des actions Amazon SQS qui permettent à Elastic Beanstalk de surveiller l'activité de file d'attente pour les environnements de travail. Pour consulter le contenu de cette stratégie gérée, consultez la AWSElasticBeanstalkEnhancedHealthpage du Guide de référence des politiques AWS gérées.

`AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy`

Cette politique octroie des autorisations à Elastic Beanstalk pour mettre à jour les environnements en votre nom afin d'effectuer des mises à jour de plateforme gérées. Pour consulter le contenu de cette stratégie gérée, consultez la AWSElasticBeanstalkManagedUpdatesCustomerRolePolicypage du Guide de référence des politiques AWS gérées.

Groupements d'autorisations de niveau service

Cette politique est groupée en instructions basées sur le jeu d'autorisations fourni.

- *ElasticBeanstalkPermissions*— Ce groupe d'autorisations permet d'appeler les actions du service Elastic Beanstalk (Elastic Beanstalk). APIs
- *AllowPassRoleToElasticBeanstalkAndDownstreamServices* – Ce groupe d'autorisations permet de transmettre n'importe quel rôle à Elastic Beanstalk et à d'autres services aval comme CloudFormation.
- *ReadOnlyPermissions* — Ce groupe d'autorisations sert à collecter des informations sur l'environnement en cours d'exécution.
- **OperationPermissions* — Les groupes avec ce schéma de nommage servent à appeler les opérations nécessaires pour effectuer les mises à jour de la plateforme.
- **BroadOperationPermissions* — Les groupes avec ce schéma de nommage servent à appeler les opérations nécessaires pour effectuer les mises à jour de la plateforme. Ils incluent également des autorisations étendues pour la prise en charge d'environnements hérités.
- **TagResource*— Les groupes dotés de ce modèle de dénomination sont destinés aux appels qui utilisent le tag-on-create APIs pour attacher des balises à des ressources créées dans un environnement Elastic Beanstalk.

Vous pouvez créer un environnement Elastic Beanstalk en utilisant l'une des approches suivantes. Chaque section décrit la manière dont l'approche gère la fonction du service.

Console Elastic Beanstalk

Si vous créez un environnement en utilisant la console Elastic Beanstalk, ce dernier vous invite à créer une fonction du service nommée `aws-elasticbeanstalk-service-role`. Lorsqu'il est créé via Elastic Beanstalk, cette fonction inclut une politique de confiance qui permet à Elastic Beanstalk d'assumer la fonction du service. Les deux politiques gérées décrites plus haut dans cette rubrique sont également associées à la fonction.

Interface de ligne de commande Elastic Beanstalk (EB CLI)

Vous pouvez créer un environnement en utilisant la commande the section called “eb create” de l'interface de ligne de commande Elastic Beanstalk (EB CLI). Si vous ne spécifiez pas de fonction du service via l'option `--service-role`. Elastic Beanstalk crée la même fonction du service par défaut `aws-elasticbeanstalk-service-role`. Si le rôle de service par défaut existe déjà,

Elastic Beanstalk l'utilise pour le nouvel environnement. Lorsqu'il est créé via Elastic Beanstalk, cette fonction inclut une politique de confiance qui permet à Elastic Beanstalk d'assumer la fonction du service. Les deux politiques gérées décrites plus haut dans cette rubrique sont également associées au rôle.

API Elastic Beanstalk

Vous pouvez créer un environnement en utilisant l'action `CreateEnvironment` de l'API Elastic Beanstalk. Si vous ne spécifiez pas de fonction du service, Elastic Beanstalk crée un rôle lié au service de surveillance. Il s'agit d'un type unique de rôle de service prédéfini par Elastic Beanstalk afin d'inclure toutes les autorisations dont le service a besoin pour appeler d'autres personnes en votre nom. Services AWS Le rôle lié à un service est associé à votre compte. Elastic Beanstalk le crée une seule fois, et le réutilise lors de la création d'environnements supplémentaires. Vous pouvez également utiliser IAM pour créer à l'avance le rôle lié au service de surveillance de votre compte. Lorsque votre compte dispose d'un rôle lié à un service de surveillance, vous pouvez l'utiliser pour créer un environnement à l'aide de la console ou de l'API Elastic Beanstalk, ou bien via l'interface de ligne de commande EB. Pour obtenir des instructions sur l'utilisation des rôles liés à un service avec les environnements Elastic Beanstalk, veuillez consulter Utilisation des rôles liés à un service pour Elastic Beanstalk.

Pour de plus amples informations sur les rôles de service, veuillez consulter Gestion des rôles de service Elastic Beanstalk.

Profil d'instance Elastic Beanstalk

Un profil d'instance est un rôle IAM qui est appliqué aux instances Amazon EC2 lancées dans votre environnement Elastic Beanstalk. Lorsque vous créez un environnement Elastic Beanstalk, vous spécifiez le profil d'instance utilisé lorsque vos instances EC2 effectuent les actions suivantes :

- récupèrent les versions de l'application depuis Amazon Simple Storage Service (Amazon S3) ;
- écrivent des journaux dans Amazon S3 ;
- dans des environnements intégrés AWS X-Ray, chargent des données de débogage dans X-Ray
- dans les environnements Docker gérés par Amazon ECS, coordonez les déploiements de conteneurs avec Amazon Elastic Container Service (Amazon ECS) ;
- lisent à partir d'une file d'attente Amazon Simple Queue Service (Amazon SQS) (dans les environnements de travail) ;
- effectuent des choix principaux avec Amazon DynamoDB (dans les environnements de travail) ;

- Dans les environnements de travail, publiez les indicateurs de santé des instances sur Amazon CloudWatch

Politiques gérées

Elastic Beanstalk fournit un ensemble de stratégies gérées qui permettent aux instances EC2 de votre environnement d'effectuer les opérations requises. Les stratégies gérées requises pour les cas d'utilisation de base sont les suivantes.

- `AWSElasticBeanstalkWebTier`
- `AWSElasticBeanstalkWorkerTier`
- `AWSElasticBeanstalkMulticontainerDocker`

Si votre application Web nécessite l'accès à d'autres services supplémentaires Services AWS, ajoutez des instructions ou des politiques gérées au profil d'instance qui autorisent l'accès à ces services. Pour de plus amples informations, veuillez consulter Ajout d'autorisations au profil d'instance par défaut.

Création d'un profil d'instance EC2

Si votre AWS compte ne possède pas de profil d'instance EC2, vous devez en créer un à l'aide du service IAM. Vous pouvez ensuite attribuer le profil d'instance EC2 aux nouveaux environnements que vous créez. Les étapes de création d'un environnement de la console Elastic Beanstalk vous permettent d'accéder à la console IAM, afin que vous puissiez créer un profil d'instance EC2 avec les autorisations requises.

Vous pouvez également créer un profil d'instance EC2 en accédant directement à la console IAM, sans passer par la console Elastic Beanstalk. Pour connaître les étapes détaillées de création d'un profil d'instance Elastic Beanstalk EC2 dans la console IAM, consultez. Création d'un profil d'instance

Stratégie utilisateur Elastic Beanstalk

Pour éviter d'utiliser votre compte racine ou de partager les informations d'identification, créez des utilisateurs IAM pour chaque utilisateur qui exploite Elastic Beanstalk. En tant que bonne pratique de sécurité, n'accordez à ces utilisateurs que les autorisations d'accès aux services et fonctions dont ils ont besoin.

Elastic Beanstalk exige des autorisations non seulement pour les actions de ses propres API, mais aussi pour plusieurs autres AWS . Elastic Beanstalk utilise les autorisations utilisateur pour lancer des ressources dans un environnement. Ces ressources incluent des instances EC2, un équilibreur de charge Elastic Load Balancing et un groupe Auto Scaling. Elastic Beanstalk utilise également les autorisations des utilisateurs pour enregistrer des journaux et des modèles sur Amazon Simple Storage Service (Amazon S3), envoyer des notifications à Amazon SNS, attribuer des profils d'instance et publier des métriques sur. CloudWatch Elastic CloudFormation Beanstalk nécessite des autorisations pour orchestrer les déploiements et les mises à jour des ressources. Ce service a également besoin d'autorisations Amazon RDS pour créer des bases de données si nécessaire, et d'autorisations Amazon SQS pour créer des files d'attente pour les environnements de travail.

Pour plus d'informations sur les stratégies utilisateur, consultez Gestion des stratégies utilisateur Elastic Beanstalk.

Tutoriels et exemples

Des didacticiels spécifiques au langage et au framework sont répartis dans le guide du AWS Elastic Beanstalk développeur. Nous actualisons cette liste en permanence en y ajoutant les didacticiels que nous créons ou mettons à jour. Les mises à jour les plus récentes sont affichées en premier.

Ces didacticiels s'adressent aux utilisateurs de niveau intermédiaire et peuvent ne pas contenir d'instructions pour des procédures de base, telles que l'inscription à AWS. Si c'est la première fois que vous utilisez AWS Elastic Beanstalk, consultez la procédure pas à pas pour démarrer afin de mettre en place votre premier environnement Elastic Beanstalk.

- Ruby on Rails - Déploiement d'une application rails sur Elastic Beanstalk
- Ruby et Sinatra - Déploiement d'une application sinatra sur Elastic Beanstalk
- Configuration haute disponibilité PHP et MySQL - Déploiement d'une application PHP haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk
- PHP et Laravel - Déploiement d'une application Laravel sur Elastic Beanstalk
- PHP et CakePHP - Déploiement d'une application CakePHP sur Elastic Beanstalk
- Configuration haute disponibilité PHP et Drupal - Déploiement d'un site web Drupal haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk
- Configuration PHP et WordPress HA - Déploiement d'un WordPress site Web à haute disponibilité avec une base de données Amazon RDS externe sur Elastic Beanstalk
- Configuration haute disponibilité Node.js avec DynamoDB - Déploiement d'une application Node.js avec DynamoDB vers Elastic Beanstalk
- ASP.NET Core - QuickStart: Déployer une application ASP.NET sur Elastic Beanstalk
- Python et Flask - Déploiement d'une application Flask sur Elastic Beanstalk
- Python et Django - Déploiement d'une application Django sur Elastic Beanstalk
- Node.js et Express - Déploiement d'une application Node.js Express sur Elastic Beanstalk
- Docker, PHP et nginx - Création d'un environnement Docker géré par ECS avec la console Elastic Beanstalk

Vous pouvez utiliser les liens suivants afin de télécharger les exemples d'application utilisés par Elastic Beanstalk lorsque vous créez un environnement sans fournir de bundle de fichiers source :

- Docker – docker.zip

- Docker multiconteneur — 2.zip docker-multicontainer-v
- Docker préconfiguré (Glassfish) — 1.zip docker-glassfish-v
- Go – go.zip
- Corretto – corretto.zip
- Tomcat – tomcat.zip
- .NET Core sous Linux — dotnet-core-linux.zip
- .NET Core — dotnet-asp-windows.zip
- Node.js – nodejs.zip
- PHP – php.zip
- Python – python.zip
- Ruby – ruby.zip

Des exemples d'applications plus complexes qui montrent l'utilisation de frameworks, de bibliothèques et d'outils Web supplémentaires sont disponibles sous forme de projets open source sur GitHub :

- Équilibrage de charge WordPress (didacticiel) : fichiers de configuration permettant de l'installer WordPress en toute sécurité et de l'exécuter dans un environnement Elastic Beanstalk à charge équilibrée.
- Drupal à charge équilibrée (tutoriel) – Fichiers de configuration et instructions pour l'installation de Drupal en toute sécurité et son exécution dans un environnement Elastic Beanstalk à charge équilibrée.
- Scorekeep - API RESTful Web qui utilise le framework Spring et fournit une interface pour créer et gérer des utilisateurs, des sessions et des jeux. AWS SDK pour Java Cette API est regroupée avec une application web Angular 1.5 qui utilise l'API sur HTTP. Inclut des branches présentant une intégration avec Amazon Cognito et Amazon Relational Database Service. AWS X-Ray

  L'application utilise des fonctionnalités de la plateforme Java SE pour télécharger des dépendances et des instances intégrées, ce qui réduit la taille du bundle de fichiers source. L'application inclut également des fichiers de configuration nginx qui remplacent la configuration par défaut pour servir l'application web frontale de manière statique sur le port 80 via le proxy et acheminer les requêtes vers des chemins sous `/api` vers l'API s'exécutant sur `localhost:5000`.
- Y a-t-il des serpents ? - Application Tomcat qui montre l'utilisation de RDS dans une application Web Java EE dans Elastic Beanstalk. Le projet montre l'utilisation des fichiers de configuration

Servlets JSPs, Simple Tag Support, Tag Files, JDBC, SQL, Log4J, Bootstrap, Jackson et Elastic Beanstalk.

- Locust Load Generator - Ce projet décrit comment utiliser les fonctionnalités de la plateforme Java SE pour installer et exécuter Locust, un outil de test de charge développé en Python. Ce projet inclut des fichiers de configuration qui installent et configurent Locust, un script de construction qui configure une table DynamoDB et un Procfile qui exécute Locust.
- Share Your Thoughts (tutoriel) - Application PHP qui décrit l'utilisation de MySQL sur Amazon RDS, Composer et les fichiers de configuration.
- A New Startup (tutoriel) - Exemple d'application Node.js qui montre l'utilisation de DynamoDB, du SDK JavaScript pour Node.js, de AWS la gestion des packages npm et des fichiers de configuration.

Migration des applications IIS vers Elastic Beanstalk

AWS Elastic Beanstalk fournit un chemin de migration rationalisé pour vos applications Windows exécutées sur Internet Information Services (IIS). La fonctionnalité de migration décrite dans ce chapitre réduit considérablement le temps et la complexité généralement associés aux migrations vers le cloud, en vous aidant à maintenir les fonctionnalités des applications et l'intégrité de la configuration pendant la transition vers AWS.

L'eb migrateopération

Utilisez la eb migrate commande dans l'interface de ligne de commande Elastic Beanstalk (EB CLI) pour découvrir, empaqueter et déployer automatiquement vos applications IIS sur le. AWS Cloud Le processus préserve les fonctionnalités de l'application et préserve vos configurations, notamment les liaisons, les pools d'applications et les paramètres d'authentification.

Les étapes suivantes résument le processus effectué par l'`eb migrate`opération pour faire passer votre application vers AWS Cloud :

1. Découvrez les sites IIS et leurs configurations.
2. Package du contenu et de la configuration de l'application.
3. Créez l'environnement et l'application Elastic Beanstalk.
4. Déployez l'application avec des paramètres préservés.

Options d'exécution du flux de travail et de l'emplacement

La eb migrate commande fournit des options pour des flux de travail de migration et des emplacements d'exécution flexibles. Par défaut, exécutez la commande sur le serveur cible qui contient l'application que vous souhaitez migrer vers Elastic Beanstalk. Si vous ne pouvez pas exécuter de commandes directement sur le serveur d'applications, utilisez l'`remote`option permettant d'exécuter la commande depuis un hôte bastion qui se connecte au serveur cible contenant votre application et vos configurations. Pour effectuer la migration en deux étapes, vous pouvez également générer le package de migration sans le déployer à l'aide de l'`archive-only`option, puis le déployer ultérieurement à votre convenance en utilisant l'`archive`option.

Pour des informations de référence sur la eb migrate commande, consultezthe section called “eb migrate”.

Rubriques

Les rubriques suivantes fournissent des informations détaillées sur la migration des applications IIS vers Elastic Beanstalk :

- the section called “Conditions préalables”- Découvrez les logiciels, les accès et les autorisations nécessaires pour migrer vos applications Windows vers AWS Elastic Beanstalk des environnements.
- the section called “Glossaire des migrations”- Découvrez comment les composants IIS sont mappés aux ressources Elastic Beanstalk
- the section called “Cartographie de la migration entre IIS et Elastic Beanstalk”- Découvrez comment les composants IIS sont mappés aux ressources Elastic Beanstalk
- the section called “Migrations IIS de base”- Apprenez à effectuer des migrations de base
- the section called “Scénarios de migration avancés”- Gérez des scénarios de migration complexes
- the section called “Configuration de sécurité”- Configurer les paramètres de sécurité lors de la migration
- the section called “Configuration réseau”- Gérez les configurations du réseau et des ports
- the section called “Dépannage et diagnostic”- Résoudre les problèmes de migration courants
- the section called “Options de migration : EB CLI ou MGN”- Comparez deux options de migration principales.

Conditions préalables

Avant d'utiliser la eb migrate commande, assurez-vous que votre environnement répond aux exigences suivantes :

Installation et version d'IIS

Le serveur à partir duquel vous effectuez la migration doit exécuter Internet Information Services (IIS) version 7.0 ou ultérieure. IIS 10.0 sur Windows Server 2016 ou version ultérieure fournit l'environnement le plus compatible pour la migration.

Pour vérifier votre version d'IIS, exécutez la commande suivante :

```
PS C:\migrations_workspace> Get-ItemProperty "HKLM:\SOFTWARE\Microsoft\InetStp\"
...
SetupString             : IIS 10.0
VersionString           : Version 10.0
...
```

Configuration requise pour Windows Server

Votre environnement source doit exécuter Windows Server 2016 ou version ultérieure pour une compatibilité optimale. Elastic Beanstalk prend en charge les versions de Windows Server suivantes en tant que plateformes cibles :

- Windows Server 2025
- Windows Server 2022
- Windows Server 2019
- Windows Server 2016

Installation de la CLI EB

- Flux de travail par défaut (sans `--remote` option) :
  - Python et l'interface de ligne de commande Elastic Beanstalk (EB CLI) doivent être installés sur le serveur qui contient l'application que vous souhaitez migrer vers Elastic Beanstalk. Bien que cela ne soit pas obligatoire, nous vous recommandons d'installer l'interface de ligne de commande EB dans un `virtualenv` bac à sable, comme décrit dansthe section called “Installer dans virtualenv”.
- En utilisant l'`--remote`option :
  - Python et l'interface de ligne de commande Elastic Beanstalk (EB CLI) doivent être installés sur votre hôte bastion. Bien que cela ne soit pas obligatoire, nous vous recommandons d'installer l'interface de ligne de commande EB dans un `virtualenv` bac à sable, comme décrit dansthe section called “Installer dans virtualenv”.

Autorisations nécessaires

Vous avez besoin des informations d'identification et autorisations suivantes :

- Privilèges d'administrateur sur le serveur IIS source ou sur l'hôte Bastion (si vous utilisez l'`--remote`option).
- AWS informations d'identification avec autorisations pour créer et gérer les ressources Elastic Beanstalk

Déploiement Web 3.6

L'outil Microsoft Web Deploy (version 3.6 ou ultérieure) doit être installé sur votre serveur source ou sur l'hôte Bastion (si vous utilisez `--remote` cette option). Cet outil est utilisé eb migrate pour empaqueter vos applications.

Pour vérifier l'installation, exécutez la commande suivante :

:

```
PS C:\migrations_workspace> Get-ItemProperty "HKLM:\SOFTWARE\Microsoft\IIS
 Extensions\MSDeploy\3" -Name InstallPath

InstallPath  : C:\Program Files\IIS\Microsoft Web Deploy V3\
...
```

Pour les instructions d'installation, consultez la section Installation et configuration de Web Deploy sur IIS 8.0 ou version ultérieure sur le site Web de documentation du produit Microsoft Windows.

Exigences relatives au réseau

- Flux de travail par défaut (sans `--remote` option) :
  - Votre serveur source doit disposer d'un accès Internet sortant aux AWS services.
- En utilisant l'`--remote`option :
  - Votre serveur source doit disposer d'un accès Internet sortant aux AWS services.
  - Configurez les règles d'entrée des groupes de sécurité appropriées qui autorisent une connexion réseau sortante depuis votre hôte Bastion et une connexion entrante vers la machine distante. Assurez-vous que l'adresse IP de l'hôte bastion est autorisée via TCP sur le port 22 pour accéder à la machine distante.
  - Assurez-vous que votre client SSH est installé et fonctionne sur votre machine distante ainsi que sur votre hôte Bastion.
  - Assurez-vous que la configuration de votre pare-feu contient les règles appropriées qui ouvrent le port 22 ou autorisent la connexion au client.
  - Testez votre connexion en utilisant manuellement une connexion SSH sur l'hôte distant depuis l'hôte bastion avant de tenter la migration.

Glossaire des migrations

Ce glossaire fournit les définitions des termes et concepts clés liés à IIS, à Elastic Beanstalk et à la migration des applications IIS vers Elastic Beanstalk.

Termes relatifs à Windows, IIS et .NET

IIS

Internet Information Services, un logiciel de serveur Web développé par Microsoft pour une utilisation avec Windows Server. IIS héberge des sites Web, des applications Web et des services Web, fournissant ainsi une plate-forme pour exécuter ASP.NET et d'autres technologies Web. Lors de la migration vers Elastic Beanstalk, les sites IIS et leurs configurations sont empaquetés et déployés sur des instances Windows Server dans le cloud. AWS

Les versions 7.0 et ultérieures d'IIS sont prises en charge pour la migration, IIS 10.0 sur Windows Server 2016 ou version ultérieure fournissant l'environnement le plus compatible.

Framework .NET

Plate-forme de développement logiciel développée par Microsoft pour créer et exécuter des applications Windows. Il fournit une vaste bibliothèque de classes appelée Framework Class Library (FCL) et prend en charge l'interopérabilité des langages entre plusieurs langages de programmation.

Lors de la migration vers Elastic Beanstalk, les applications créées sur le .NET Framework continuent de s'exécuter sur la même version du framework dans l'environnement cloud. Elastic Beanstalk prend en charge plusieurs versions de .NET Framework (4.x) sur ses plateformes Windows Server.

.NET Core

Successeur open source multiplateforme du .NET Framework, conçu pour être plus modulaire et léger. .NET Core (désormais simplement appelé .NET 5 et versions ultérieures) permet aux développeurs de créer des applications qui s'exécutent sous Windows, Linux et macOS.

Lorsque vous migrez des applications basées sur .NET Core vers Elastic Beanstalk, vous pouvez choisir entre des plateformes Windows Server ou des plateformes basées sur Linux, en fonction des exigences et des dépendances de votre application.

Common Language Runtime (CLR)

Composant de machine virtuelle du .NET Framework qui gère l'exécution des programmes .NET. Le CLR fournit des services tels que la gestion de la mémoire, la sécurité des types, la gestion des exceptions, la collecte des déchets et la gestion des threads.

Lors de la migration vers Elastic Beanstalk, la version CLR appropriée est automatiquement disponible sur la plate-forme Windows Server que vous sélectionnez, garantissant ainsi la compatibilité avec les exigences de votre application.

Site

Conteneur logique dans IIS qui représente une application ou un service Web, identifié par une liaison unique d'adresse IP, de port et d'en-tête d'hôte. Chaque site IIS possède son propre pool d'applications, ses propres liaisons et paramètres de configuration, et peut contenir une ou plusieurs applications.

Application

Regroupement de fichiers de contenu et de code au sein d'un site IIS qui gère les demandes relatives à un espace URL spécifique. Les applications dans IIS peuvent avoir leurs propres paramètres de configuration, qui sont généralement stockés dans `web.config` des fichiers.

Lors de la migration vers Elastic Beanstalk, les applications sont préservées avec leur structure de chemin et leurs paramètres de configuration. Le processus de migration garantit que les applications imbriquées conservent leur hiérarchie et leurs chemins d'URL dans l'environnement cloud.

ApplicationPool

Fonctionnalité IIS qui isole les applications Web pour améliorer la sécurité, la fiabilité et la gestion des performances. Les pools d'applications définissent l'environnement d'exécution des applications, notamment la version du .NET Framework, le mode pipeline et les paramètres d'identité.

VirtualDirectory

Un mappage de répertoires dans IIS qui permet de diffuser du contenu à partir d'un emplacement situé en dehors du répertoire racine du site. Les annuaires virtuels vous permettent d'organiser le contenu sur différents emplacements physiques tout en présentant une structure d'URL unifiée aux utilisateurs.

Lors de la migration vers Elastic Beanstalk, les répertoires virtuels sont conservés avec leurs mappages de chemins. La eb migrate commande crée la structure de répertoire et les configurations nécessaires dans l'environnement cloud pour conserver les mêmes chemins d'URL.

ARR

Application Request Routing, une extension IIS qui fournit des fonctionnalités d'équilibrage de charge et de proxy pour les serveurs Web. L'ARR permet le routage basé sur les URL, le transfert de requêtes HTTP et la répartition de la charge sur plusieurs serveurs.

Lors de la migration vers Elastic Beanstalk, les configurations ARR sont préservées grâce à l'installation de fonctionnalités ARR sur les instances EC2 et à la configuration de règles de routage appropriées. Pour les scénarios de routage complexes, le processus de migration peut également tirer parti des règles Application Load Balancer pour implémenter des fonctionnalités similaires.

Réécriture d'URL

Module IIS qui modifie les demandes en URLs fonction de règles définies avant qu'elles n'atteignent l'application Web. La réécriture d'URL permet la manipulation d'URL, la redirection et la diffusion de contenu en fonction de modèles et de conditions.

Lors de la migration vers Elastic Beanstalk, les règles de réécriture d'URL de vos fichiers sont traduites en règles `web.config` de routage ALB dans la mesure du possible, ou conservées dans la configuration IIS sur les instances EC2. Cela garantit que les modèles d'URL et les redirections continuent de fonctionner comme prévu dans l'environnement cloud.

msdeploy.exe

Outil de ligne de commande utilisé pour déployer des applications Web et des sites Web sur des serveurs IIS. Également connu sous le nom de Web Deploy, il permet d'empaqueter, de synchroniser et de déployer des applications Web, des sites Web et des configurations de serveur.

La eb migrate commande utilise Web Deploy (version 3.6 ou ultérieure) pour empaqueter vos applications lors de la migration vers Elastic Beanstalk. Cet outil doit être installé sur votre serveur source pour que le processus de migration fonctionne correctement.

Trajectoire physique

Emplacement réel du système de fichiers où sont stockés les fichiers de contenu d'un site ou d'une application IIS. Les chemins physiques peuvent pointer vers des répertoires locaux, des partages réseau ou d'autres emplacements de stockage accessibles au serveur IIS.

Lors de la migration vers Elastic Beanstalk, les chemins physiques sont mappés vers les emplacements appropriés sur les instances EC2 de votre environnement. Le processus de

migration préserve la structure du contenu tout en garantissant que tous les fichiers sont correctement déployés dans l'environnement cloud.

Hôte de l'application. Config

Le fichier de configuration racine pour IIS qui définit les paramètres à l'échelle du serveur et contient la configuration de tous les sites, applications et répertoires virtuels. Ce fichier se trouve dans le `%windir%\System32\inetsrv\config` répertoire et contrôle le comportement général du serveur IIS.

Lors de la migration vers Elastic Beanstalk, les `applicationHost.config` paramètres pertinents sont extraits et appliqués à la configuration IIS sur les instances EC2 de votre environnement. Cela garantit que les paramètres à l'échelle du serveur sont préservés pendant la migration.

web.config

Fichier de configuration XML utilisé dans les applications ASP.NET pour contrôler les paramètres, la sécurité et le comportement des applications au niveau de l'application ou du répertoire. `web.config`les fichiers peuvent contenir des paramètres d'authentification, d'autorisation, d'état de session, de compilation et de paramètres d'application personnalisés.

Pendant la migration vers Elastic `web.config` Beanstalk, les fichiers sont conservés et déployés avec votre application. Le processus de migration garantit que les configurations spécifiques aux applications continuent de fonctionner comme prévu dans l'environnement cloud.

DefaultDocument

Fonctionnalité IIS qui indique le fichier par défaut à servir lorsqu'un utilisateur demande un répertoire sans spécifier de nom de fichier. Les documents par défaut sont activés par défaut, et IIS 7 définit les fichiers de documents par défaut suivants dans le `applicationHost.config` fichier comme fichiers par défaut à l'échelle du serveur : Default.htm, Default.asp, Index.htm, Index.html, Iisstart.htm.

Lors de la migration vers Elastic Beanstalk, les paramètres de document par défaut sont conservés dans la configuration IIS sur les instances EC2, ce qui garantit que les demandes d'annuaire sont traitées de manière cohérente dans l'environnement cloud.

System.WebServer

Section de configuration figurant dans `web.config` ou `applicationHost.config` contenant des paramètres spécifiques à l'IIS pour les modules, les gestionnaires et les autres

comportements du serveur. Cette section contrôle la manière dont IIS traite les demandes, gère les modules et configure les fonctionnalités du serveur.

Lors de la migration vers Elastic Beanstalk, les configurations System.WebServer sont conservées dans les fichiers de votre `web.config` application et appliquées à l'installation IIS sur les instances EC2 de votre environnement. Cela garantit que les comportements spécifiques à IIS sont maintenus dans l'environnement cloud.

Termes relatifs à Elastic Beanstalk

Plateforme

Combinaison de composants du système d'exploitation, du langage de programmation, du serveur Web, du serveur d'applications et d'Elastic Beanstalk qui définit la pile logicielle pour exécuter des applications.

Pour les migrations vers Windows, Elastic Beanstalk fournit des plateformes basées sur Windows Server 2016, 2019 et 2022 avec IIS et différentes versions de .NET Framework afin de garantir la compatibilité avec votre environnement source.

SolutionStack

Configuration de plate-forme prédéfinie dans Elastic Beanstalk qui spécifie le système d'exploitation, le moteur d'exécution et les autres composants nécessaires pour exécuter une application. Conceptuellement identique à une plate-forme et utilisé de manière interchangeable pour faire fonctionner des environnements.

Au cours de la migration, la eb migrate commande sélectionne une pile de solutions appropriée en fonction de la configuration de votre environnement source, garantissant ainsi la compatibilité avec vos applications IIS.

CreateEnvironment

Action d'API Elastic Beanstalk qui crée un nouvel environnement pour héberger une version d'application. Cette API est utilisée par la eb migrate commande pour fournir les AWS ressources nécessaires à votre application migrée.

Le processus de migration configure les paramètres d'environnement appropriés en fonction de votre environnement IIS source, notamment le type d'instance, les variables d'environnement et les paramètres des options.

CreateApplicationVersion

Action d'API Elastic Beanstalk qui crée une nouvelle version d'application à partir d'un bundle source stocké dans Amazon S3. La eb migrate commande utilise cette API pour enregistrer votre application IIS packagée en tant que version dans Elastic Beanstalk.

Pendant la migration, les fichiers et la configuration de votre application sont empaquetés, téléchargés sur Amazon S3 et enregistrés en tant que version de l'application avant le déploiement.

DescribeEvents

Action d'API Elastic Beanstalk qui extrait une liste d'événements relatifs à un environnement, notamment les déploiements, les modifications de configuration et les problèmes opérationnels. La eb migrate commande utilise cette API pour suivre la progression de votre migration.

Vous pouvez également utiliser la eb events commande après la migration pour consulter l'historique des événements de votre environnement.

DescribeEnvironmentHealth

Action d'API Elastic Beanstalk qui fournit des informations détaillées sur l'état des instances et des autres composants d'un environnement. Cette API est utilisée pour vérifier l'état de votre application migrée après le déploiement.

Après la migration, vous pouvez utiliser la eb health commande pour vérifier l'état de votre environnement et identifier les problèmes nécessitant une attention particulière.

Santé D

Agent de surveillance dans Elastic Beanstalk qui collecte des métriques, surveille les journaux et indique l'état de santé des instances EC2 dans un environnement. HealthD fournit des rapports de santé améliorés pour vos applications migrées.

Après la migration, HealthD surveille les performances de votre application, l'utilisation des ressources et les taux de réussite des demandes, fournissant ainsi une vue complète de l'état de votre environnement.

Journaux du bundle

Fonctionnalité d'Elastic Beanstalk qui compresse et télécharge les journaux des instances EC2 vers Amazon S3 pour un stockage et une analyse centralisés. Cette fonctionnalité vous aide à résoudre les problèmes liés à vos applications migrées.

Après la migration, vous pouvez utiliser la eb logs commande pour récupérer et consulter les journaux de votre environnement.

aws-windows-deployment-manifest.json

Fichier qui décrit le contenu, les dépendances et la configuration d'un progiciel ou d'une application. Ce manifeste est généré pendant le processus de migration pour définir la manière dont vos applications IIS doivent être déployées sur Elastic Beanstalk.

section de manifeste personnalisée

Une section fournit `aws-windows-deployment-manifest.json` un contrôle personnalisé sur le déploiement des applications. Cette section contient des PowerShell scripts et des commandes exécutés au cours du processus de déploiement.

Pendant la migration, des sections de manifeste personnalisées sont générées pour gérer des aspects spécifiques de votre configuration IIS, tels que la configuration du répertoire virtuel, la gestion des autorisations et la configuration du pool d'applications.

INTERFACE DE LIGNE DE COMMANDE EB

Outil de ligne de commande qui fournit des commandes pour créer, configurer et gérer les applications et les environnements Elastic Beanstalk. L'EB CLI inclut la eb migrate commande spécifiquement destinée à la migration des applications IIS vers Elastic Beanstalk.

Après la migration, vous pouvez continuer à utiliser l'interface de ligne de commande EB pour gérer votre environnement, déployer des mises à jour, surveiller l'état de santé et effectuer d'autres tâches administratives.

Réglages des options

Des valeurs de configuration qui définissent la manière dont Elastic Beanstalk approvisionne AWS et configure les ressources de votre environnement. Les paramètres des options sont organisés en espaces de noms qui représentent les différents composants de votre environnement, tels que les équilibreurs de charge, les instances et les processus environnementaux.

Pendant la migration, la eb migrate commande génère les paramètres d'options appropriés en fonction de votre configuration IIS afin de garantir que votre environnement cloud correspond aux capacités de votre environnement source. Pour plus d'informations, consultez la section Options de configuration du manuel Elastic Beanstalk Developer Guide.

aws:elbv2:listener:default

Un espace de noms de configuration Elastic Beanstalk pour l'écouteur par défaut sur un Application Load Balancer. Pendant la migration, cet espace de noms est configuré en fonction de vos liaisons de site IIS afin de garantir le bon routage du trafic.

L'écouteur par défaut gère généralement le trafic HTTP sur le port 80, qui est ensuite transféré aux instances de votre application conformément aux règles de routage.

aws:elbv2:listener:listener_port

Un espace de noms de configuration Elastic Beanstalk pour un port d'écouteur spécifique sur un Application Load Balancer. Cet espace de noms est utilisé pour configurer des écouteurs supplémentaires pour vos applications migrées, tels que le protocole HTTPS sur le port 443.

Pendant la migration, les écouteurs sont créés en fonction des liaisons de ports de vos sites IIS, ce qui garantit que vos applications restent accessibles sur les mêmes ports que dans votre environnement source.

aws:elbv2:listenerrule:rule_name

Un espace de noms de configuration Elastic Beanstalk permettant de définir des règles de routage pour un écouteur Application Load Balancer. Ces règles déterminent la manière dont les demandes entrantes sont acheminées vers les différents groupes cibles en fonction des modèles de chemin ou des en-têtes d'hôte.

Au cours de la migration, des règles d'écoute sont créées pour correspondre à la structure d'URL de vos applications IIS, afin de garantir que les demandes sont acheminées vers les chemins d'application appropriés.

aws:elasticbeanstalk:environnement:process:default

Un espace de noms de configuration Elastic Beanstalk pour le processus par défaut dans un environnement. Cet espace de noms définit la manière dont le processus d'application Web par défaut est configuré, y compris les paramètres de vérification de l'état, les mappages de ports et les paramètres de proxy.

Pendant la migration, le processus par défaut est configuré en fonction des paramètres de votre site IIS principal, afin de garantir une surveillance de l'état et une gestion des demandes appropriées.

aws:elasticbeanstalk:environnement:process:process_name

Un espace de noms de configuration Elastic Beanstalk pour un processus nommé spécifique dans un environnement. Cet espace de noms vous permet de définir plusieurs processus avec différentes configurations, comme si vous disposiez de plusieurs pools d'applications dans IIS.

Au cours de la migration, des processus supplémentaires peuvent être créés pour représenter différentes liaisons de site par rapport à votre environnement source.

> **Note**
>
> Pour plus d'informations sur certains termes décrits dans cette rubrique, consultez les ressources suivantes :
>
> - Actions de l'API Elastic Beanstalk - Référence des API AWS Elastic Beanstalk
> - Plateformes Elastic Beanstalk, y compris les versions prises en charge - Plateformes prises en charge dans le guide des plateformes AWS Elastic Beanstalk
> - Espaces de noms de configuration d'Elastic Beanstalk : dans ce guide Options générales pour tous les environnements
> - L'EB CLI ou les commandes spécifiques de l'EB CLI - Configuration de l'interface de ligne de commande EB (EB CLI) pour gérer Elastic Beanstalk dans ce guide

Termes de Python

pip

Le programme d'installation de packages pour Python, utilisé pour installer et gérer des packages logiciels écrits en Python. La CLI EB est installée et mise à jour à l'aide de pip.

Au cours du processus de migration, pip est utilisé pour installer le package EB CLI et ses dépendances sur votre serveur source, fournissant ainsi les outils nécessaires à la migration.

PyPI

Python Package Index, le référentiel officiel des packages logiciels Python tiers, à partir duquel pip récupère et installe les packages. L'EB CLI et ses dépendances sont hébergées sur PyPI.

Lors de l'installation de l'EB CLI pour la migration, pip se connecte à PyPI pour télécharger et installer les packages nécessaires.

virtualenv

Un outil pour créer des environnements Python isolés, permettant à différents projets d'avoir leurs propres dépendances et packages sans conflits. L'utilisation de virtualenv est recommandée lors de l'installation de la CLI EB afin d'éviter les conflits avec d'autres applications Python.

La création d'un environnement virtuel avant l'installation de l'interface de ligne de commande EB garantit que les outils de migration disposent d'un environnement propre et isolé avec les dépendances appropriées.

pywin33

Ensemble d'extensions Python qui donnent accès à de nombreux systèmes Windows APIs, permettant ainsi une interaction avec le système d'exploitation Windows et ses composants. L'EB CLI utilise pywin32 pour interagir avec les fonctionnalités spécifiques à Windows lors de la migration.

Au cours du processus de migration, pywin32 est utilisé pour accéder à la configuration IIS, aux paramètres du registre Windows et à d'autres informations système nécessaires pour empaqueter et migrer correctement vos applications.

pythonnet

Package qui permet au code Python d'interagir avec les applications .NET Framework et .NET Core. Cette intégration permet à l'EB CLI de fonctionner avec les composants .NET pendant le processus de migration.

Le processus de migration peut utiliser pythonnet pour interagir avec les assemblages et les composants .NET lors de l'analyse et du packaging de vos applications en vue de leur déploiement sur Elastic Beanstalk.

Réalisation de migrations IIS de base

Cette section vous guide tout au long du processus de migration de vos applications IIS vers Elastic eb migrate Beanstalk à l'aide de la commande.

Exploration de votre environnement IIS

Avant d'apporter des modifications, vous devez connaître les ressources présentes sur votre serveur. Commencez par explorer vos sites IIS en exécutanteb migrate explore, comme illustré dans l'exemple suivant :

```
PS C:\migrations_workspace> eb migrate explore
```

Cette commande affiche vos sites IIS. Reportez-vous à la liste suivante :

```
Default Web Site
Intranet
API.Internal
Reports
```

Pour une vue détaillée de la configuration de chaque site, y compris les liaisons, les applications et les répertoires virtuels, ajoutez l'`--verbose`option, comme indiqué dans cet exemple :

```
PS C:\migrations_workspace> eb migrate explore --verbose
```

La liste suivante présente les informations complètes sur votre environnement fournies par la commande :

```
1: Default Web Site:
  - Bindings:
    - *:80:www.example.com
    - *:443:www.example.com
  - Application '/':
    - Application Pool: DefaultAppPool
    - Enabled Protocols: http
    - Virtual Directories:
      - /:
        - Physical Path: C:\inetpub\wwwroot
        - Logon Method: ClearText
  - Application '/api':
    - Application Pool: ApiPool
    - Enabled Protocols: http
    - Virtual Directories:
      - /:
        - Physical Path: C:\websites\api
        - Logon Method: ClearText
2: Intranet:
...
3. API.Internal:
...
4. Reports:
...
```

Comprendre le résultat de la découverte

La sortie détaillée fournit les informations essentielles suivantes pour la planification de la migration :

Sites

Le résultat de découverte répertorie tous les sites IIS de votre serveur. Chaque site est identifié par son nom (par exemple, « Site Web par défaut », « Intranet », « API.Internal ») et numéroté de manière séquentielle. Lorsque plusieurs sites existent sur un serveur, la `eb migrate` commande peut empaqueter et déployer chacun d'eux séparément ou ensemble, en fonction de votre stratégie de migration.

Reliures

Les liaisons de protocole révèlent les protocoles (HTTP/HTTPS) utilisés par vos sites et les ports sur lesquels ils fonctionnent. Les informations de liaison incluent les exigences d'en-tête de l'hôte qui définissent les configurations de routage basées sur le domaine.

Applications

Les chemins d'application indiquent à la fois les structures d'application racine et imbriquées au sein de votre configuration IIS. Les attributions de pool d'applications indiquent comment vos applications sont isolées les unes des autres pour la gestion de la sécurité et des ressources.

Répertoires virtuels

Les mappages de chemins physiques indiquent l'emplacement de votre contenu dans le système de fichiers. Les paramètres d'authentification indiquent des exigences d'accès spéciales qui doivent être maintenues après la migration.

Préparation de la migration

Grâce à une bonne connaissance de votre environnement, assurez-vous que votre serveur répond aux prérequis. Vérifiez d'abord votre version d'IIS à l'aide de la PowerShell commande suivante :

```
PS C:\migrations_workspace> Get-ItemProperty "HKLM:\SOFTWARE\Microsoft\InetStp\" -Name
 MajorVersion
```

Vous avez besoin d'IIS 7.0 ou version ultérieure. L'outil de migration utilise Web Deploy 3.6 pour empaqueter vos applications. Vérifiez son installation à l'aide de la commande suivante :

```
PS C:\migrations_workspace> Get-ItemProperty "HKLM:\SOFTWARE\Microsoft\IIS Extensions
\MSDeploy\3" -Name InstallPath
```

Si Web Deploy n'est pas installé sur votre serveur, vous pouvez le télécharger depuis la page de téléchargement de Microsoft Web Platform Installer.

Votre première migration

Commençons par une migration de base du site Web par défaut. L'exemple suivant montre la commande la plus simple,eb migrate.

```
PS C:\migrations_workspace> eb migrate
```

Cette commande lance une série d'étapes automatisées, illustrées dans l'exemple de sortie suivant :

```
Identifying VPC configuration of this EC2 instance (i-0123456789abcdef0)
  id: vpc-1234567890abcdef0
  publicip: true
  elbscheme: public
  ec2subnets: subnet-123,subnet-456,subnet-789
  securitygroups: sg-123,sg-456
  elbsubnets: subnet-123,subnet-456,subnet-789

Using .\migrations\latest to contain artifacts for this migration run.
Generating source bundle for sites, applications, and virtual directories...
  Default Web Site/ -> .\migrations\latest\upload_target\DefaultWebSite.zip
```

L'outil de migration crée un répertoire structuré contenant vos artefacts de déploiement. La liste suivante montre la structure du répertoire :

```
C:\migration_workspace\
### .\migrations\latest\
    ### upload_target\
        ### DefaultWebSite.zip
        ### aws-windows-deployment-manifest.json
        ### ebmigrateScripts\
            ### site_installer.ps1
            ### permission_handler.ps1
            ### >other helper scripts<
```

Contrôle de la migration

Pour mieux contrôler le processus de migration, vous pouvez spécifier exactement les sites à migrer à l'aide de la commande suivante :

```
PS C:\migrations_workspace> eb migrate --sites "Default Web Site,Intranet"
```

Vous pouvez également personnaliser le nom de l'environnement et le nom de l'application, comme indiqué dans l'exemple de commande suivant :

```
PS C:\migrations_workspace> eb migrate `
    --sites "Default Web Site" `
    --application-name "CorporateApp" `
    --environment-name "Production"
```

Pour une liste complète des options, voirthe section called “eb migrate”.

Surveillance de la progression

Pendant la migration, eb migrate fournit des mises à jour de statut en temps réel. Reportez-vous à l'exemple de sortie suivant :

```
...
Creating application version
Creating environment... This may take a few minutes

2024-03-18 18:12:15    INFO    Environment details for: Production
  Application name: CorporateApp
  Region: us-west-2
  Deployed Version: app-230320_153045
  Environment ID: e-abcdef1234
  Platform: 64bit Windows Server 2019 v2.7.0 running IIS 10.0
  Tier: WebServer-Standard-1.0
  CNAME: production.us-west-2.elasticbeanstalk.com
  Updated: 2024-03-20 15:30:45
2025-03-18 18:12:17    INFO    createEnvironment is starting.
2025-03-18 18:12:19    INFO    Using elasticbeanstalk-us-east-1-180301529717 as Amazon
 S3 storage bucket for environment data.
2025-03-18 18:12:40    INFO    Created security group named: sg-0fdd4d696a26b086a
2025-03-18 18:12:48    INFO    Environment health has transitioned to Pending.
 Initialization in progress (running for 7 seconds). There are no instances.
```

```
...
2025-03-18 18:23:59    INFO    Application available at EBMigratedEnv-arrreal3.us-
east-1.elasticbeanstalk.com.
2025-03-18 18:24:00    INFO    Successfully launched environment: EBMigratedEnv-
arrreal3
```

Vérification de la migration

Une fois l'environnement prêt, Elastic Beanstalk propose plusieurs méthodes pour vérifier votre déploiement.

Accédez à votre application

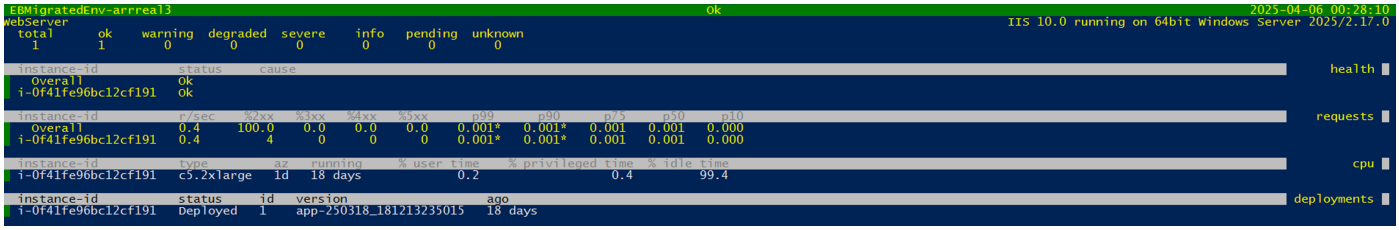
Ouvrez l'URL de votre application (CNAME) dans un navigateur Web pour vérifier qu'elle fonctionne correctement.

Vérifiez la santé de l'environnement

Utilisez la eb health commande pour vérifier l'état de votre environnement.

```
PS C:\migrations_workspace> eb health
```

L'image d'écran suivante montre l'état de santé de l'instance, les mesures de réponse des applications et l'utilisation des ressources système.



Utilisez la eb logs commande pour accéder aux journaux afin de résoudre les problèmes éventuels :

```
PS C:\migrations_workspace> eb logs --zip
```

La eb logs commande télécharge les journaux `.elasticbeanstalk/logs` dans le répertoire. Pour de plus amples informations, veuillez consulter the section called “CloudWatch Journaux”.

Connect aux instances

Si vous avez spécifié une paire de clés lors de la migration, vous pouvez vous connecter à vos instances à l'aide du protocole RDP pour un dépannage direct.

Accédez à la console Elastic Beanstalk

Vous pouvez consulter l'état, les journaux et les propriétés de configuration de l'environnement via la console de gestion de l'environnement correspondant.

Gestion des artefacts de migration

La eb migrate commande crée des artefacts locaux pendant le processus de migration. Ces artefacts contiennent des informations sensibles et peuvent consommer beaucoup d'espace disque au fil du temps. Utilisez la cleanup sous-commande pour gérer ces artefacts, comme indiqué dans l'exemple suivant :

```
PS C:\migrations_workspace> eb migrate cleanup
Are you sure you would like to cleanup older artifacts within ./migrations/? (Y/N):
```

Pour forcer le nettoyage sans confirmation, utilisez l'--forceoption suivante :

```
PS C:\migrations_workspace> eb migrate cleanup --force
```

Le processus de nettoyage préserve la migration réussie la plus récente dans le `./migrations/latest` répertoire et supprime les anciens répertoires de migration

Configuration réseau et paramètres des ports

Cette section décrit les options de configuration réseau pour les migrations IIS, notamment les paramètres VPC, les configurations de port et les déploiements multisites.

Configuration VPC

La eb migrate commande fournit des options de configuration VPC flexibles pour votre environnement Elastic Beanstalk. L'outil peut soit détecter les paramètres VPC à partir d'une instance EC2 source, soit accepter des configurations VPC personnalisées via des paramètres de ligne de commande. Passez Utilisation d'Elastic Beanstalk avec Amazon VPC en revue pour comprendre comment configurer Elastic Beanstalk avec VPC.

Détection automatique des VPC

Lorsqu'il est eb migrate exécuté sur une instance EC2, il découvre et utilise automatiquement la configuration VPC des instances EC2 de l'environnement source. L'exemple de sortie suivant illustre les informations de configuration détectées :

```
PS C:\migrations_workspace > eb migrate
Identifying VPC configuration of this EC2 instance (i-0123456789abcdef0):
  id: vpc-1234567890abcdef0
  publicip: true
  elbscheme: public
  ec2subnets: subnet-123,subnet-456,subnet-789
  securitygroups: sg-123,sg-456
  elbsubnets: subnet-123,subnet-456,subnet-789
...
```

La configuration détectée inclut :

- Identifiant VPC
- Paramètres d'attribution d'adresses IP publiques
- Schéma d'équilibrage de charge (public/privé)
- Attributions de sous-réseaux d'instances EC2
- Associations de groupes de sécurité
- Attributions de sous-réseaux d'équilibreur de charge

Hôtes sur site ou hors AWS cloud

Lorsqu'il est eb migrate exécuté à partir d'un serveur sur site ou d'un hôte non AWS cloud, le service Elastic Beanstalk utilise le VPC par défaut de votre compte. AWS La liste suivante présente un exemple de commande et de sortie :

```
PS C:\migrations_worspace> eb migrate `
      -k windows-test-pem `
      --region us-east-1 `
      -a EBMigratedEnv `
      -e EBMigratedEnv-test2 `
      --copy-firewall-config
Determining EB platform based on host machine properties
Using .\migrations\latest to contain artifacts for this migration run.
```

```
...
```

Passez Utilisation d'Elastic Beanstalk avec Amazon VPC en revue pour comprendre comment Elastic Beanstalk configure le VPC par défaut pour votre environnement.

Configuration VPC personnalisée

Pour tout environnement source (EC2, sur site ou hors AWS cloud) dans lequel vous avez besoin de paramètres VPC spécifiques, fournissez un fichier de configuration VPC tel que celui présenté dans l'exemple suivant :

```
{
    "id": "vpc-12345678",
    "publicip": "true",
    "elbscheme": "public",
    "ec2subnets": ["subnet-a1b2c3d4", "subnet-e5f6g7h8"],
    "securitygroups": "sg-123456,sg-789012",
    "elbsubnets": ["subnet-a1b2c3d4", "subnet-e5f6g7h8"]
}
```

Appliquez cette configuration à l'aide de la commande suivante :

```
PS C:\migrations_workspace> eb migrate --vpc-config vpc-config.json
```

> **Note**
>
> Le fichier de configuration du VPC nécessite le `id` champ qui spécifie l'ID du VPC. Tous les autres champs sont facultatifs, et Elastic Beanstalk utilisera des valeurs par défaut pour tous les champs que vous ne spécifiez pas.

> **Important**
>
> La migration ignorera tous les paramètres VPC existants de l'environnement source lorsque vous spécifiez le *`--vpc-config`* paramètre. Lorsque vous utilisez ce paramètre, la migration utilise uniquement les paramètres VPC spécifiés dans le fichier de configuration que vous transmettez. L'utilisation de ce paramètre remplace le comportement par défaut qui consiste à découvrir la configuration VPC de l'instance source ou à utiliser le VPC par défaut.

Utilisez le `--vpc-config` paramètre dans les scénarios suivants :

- Lorsque vous migrez des environnements non EC2 qui ne possèdent pas de paramètres VPC détectables
- Lorsque vous migrez vers un VPC différent de celui utilisé par l'environnement source
- Lorsque vous devez personnaliser les sélections de sous-réseaux ou les configurations de groupes de sécurité
- Lorsque la découverte automatique n'identifie pas correctement les paramètres VPC souhaités
- Lorsque vous effectuez une migration depuis un environnement local et que vous ne souhaitez pas utiliser le VPC par défaut

Configuration de la sécurité du réseau

Par défaut, eb migrate ouvre le port 80 sur les instances cibles mais ne copie pas les autres règles du pare-feu Windows depuis la machine source. Pour inclure toutes les configurations de pare-feu, utilisez la commande suivante :

```
PS C:\migrations_workspace> eb migrate --copy-firewall-config
```

Cette commande effectue les actions suivantes :

- Identifie les ports utilisés par les liaisons de sites IIS
- Récupère les règles de pare-feu correspondantes
- Génère PowerShell des scripts pour recréer des règles sur les instances cibles
- Préserve toutes les règles DENY pour le port 80 depuis la machine source (sinon le port 80 est autorisé par défaut)

Prenons un cas d'utilisation où votre machine source possède les règles de pare-feu spécifiées dans l'exemple suivant :

```
# Source machine firewall configuration
Get-NetFirewallRule | Where-Object {$_.Enabled -eq 'True'} | Get-NetFirewallPortFilter
 | Where-Object {$_.LocalPort -eq 80 -or $_.LocalPort -eq 443 -or $_.LocalPort -eq
 8081}
# Output shows rules for ports 80, 443, and 8081
```

La migration crée un script (`modify_firewall_config.ps1`) qui contient la configuration suivante :

```
New-NetFirewallRule -DisplayName "Allow Web Traffic" -Direction Inbound -Action Allow -
Protocol TCP -LocalPort 80,443
New-NetFirewallRule -DisplayName "Allow API Traffic" -Direction Inbound -Action Allow -
Protocol TCP -LocalPort 8081
```

L'outil de migration effectue automatiquement les actions suivantes :

- Extrait HTTP/HTTPS les ports de toutes les liaisons de sites IIS
- Utilise l'interface Windows Firewall INetFwPolicy2 pour énumérer les règles de pare-feu
- Filtre les règles pour n'inclure que celles qui font explicitement référence aux ports spécifiés
- Traite uniquement les liaisons de sites HTTP et HTTPS et leurs règles de pare-feu associées
- Préserve les propriétés des règles, notamment le nom d'affichage, l'action, le protocole et l'état activé
- Gère à la fois les ports individuels et les plages de ports dans les règles de pare-feu
- Ajoute le script de configuration du pare-feu au manifeste de déploiement

Configuration de l'équilibreur de charge

Vous pouvez spécifier la configuration du Load Balancer via l'`--vpc-config`argument. Les exemples suivants illustrent les paramètres.

Sélection du schéma

Choisissez entre des schémas d'équilibreur de charge publics et privés :

```
{
    "id": "vpc-12345678",
    "elbscheme": "private",
    "elbsubnets": ["subnet-private1", "subnet-private2"]
}
```

Distribution de sous-réseaux

Pour une haute disponibilité, répartissez les sous-réseaux d'équilibrage de charge entre les zones de disponibilité :

```
{
    "elbsubnets": [
        "subnet-az1", // Availability Zone 1
        "subnet-az2", // Availability Zone 2
        "subnet-az3"  // Availability Zone 3
    ]
}
```

> **Note**
>
> Alors qu'Elastic Beanstalk prend en charge la création d'environnements avec des équilibreurs de charge d'application, des équilibreurs de charge réseau et des équilibreurs de charge classiques, la commande prend uniquement en charge les eb migrate équilibreurs de charge d'application. Pour plus d'informations sur les types d'équilibreurs de charge, consultez la section Équilibreur de charge pour votre environnement Elastic Beanstalk.

Déploiements multisites avec configurations de ports

La eb migrate commande gère les déploiements IIS multisites complexes dans lesquels les applications peuvent partager des dépendances ou utiliser des ports non standard. Prenons l'exemple suivant d'une configuration d'entreprise typique avec plusieurs sites :

```
<!-- IIS Configuration -->
<sites>
    <site name="Default Web Site" id="1">
        <bindings>
            <binding protocol="http" bindingInformation="*:80:www.example.com" />
        </bindings>
    </site>
    <site name="InternalAPI" id="2">
        <bindings>
            <binding protocol="http" bindingInformation="*:8081:api.internal" />
        </bindings>
    </site>
    <site name="ReportingPortal" id="3">
        <bindings>
            <binding protocol="http" bindingInformation="*:8082:reports.internal" />
        </bindings>
    </site>
```

```
</sites>
```

Pour migrer cette configuration, utilisez les exemples de commande et de paramètres suivants :

```
PS C:\migrations_workspace> eb migrate `
    --sites "Default Web Site,InternalAPI,ReportingPortal" `
    --copy-firewall-config `
    --instance-type "c5.large"
```

La eb migrate commande crée un package de déploiement qui préserve l'identité et la configuration de chaque site. La commande génère un `aws-windows-deployment-manifest.json` qui définit la manière dont ces sites doivent être déployés. L'exemple suivant illustre un fichier json généré :

```
{
    "manifestVersion": 1,
    "deployments": {
        "msDeploy": [
            {
                "name": "DefaultWebSite",
                "parameters": {
                    "appBundle": "DefaultWebSite.zip",
                    "iisPath": "/",
                    "iisWebSite": "Default Web Site"
                }
            }
        ],
        "custom": [
            {
                "name": "InternalAPI",
                "scripts": {
                    "install": {
                        "file": "ebmigrateScripts\\install_site_InternalAPI.ps1"
                    },
                    "restart": {
                        "file": "ebmigrateScripts\\restart_site_InternalAPI.ps1"
                    },
                    "uninstall": {
                        "file": "ebmigrateScripts\\uninstall_site_InternalAPI.ps1"
                    }
                }
            },
            {
                "name": "ReportingPortal",
```

```
                "scripts": {
                    "install": {
                        "file": "ebmigrateScripts\\install_site_ReportingPortal.ps1"
                    },
                    "restart": {
                        "file": "ebmigrateScripts\\restart_site_ReportingPortal.ps1"
                    },
                    "uninstall": {
                        "file": "ebmigrateScripts\\uninstall_site_ReportingPortal.ps1"
                    }
                }
            }
        ]
    }
}
```

Le processus de migration crée les règles d'écoute Application Load Balancer suivantes qui conservent votre logique de routage d'origine :

- Le trafic du port 80 est acheminé vers le site Web par défaut
- Le trafic du port 8081 est acheminé vers l'API interne
- Le trafic du port 8082 est acheminé vers ReportingPortal

Configuration et dépendances partagées

Lorsque les sites partagent des configurations ou des dépendances, eb migrate gère ces relations de manière appropriée. Reportez-vous à l'exemple suivant où plusieurs sites partagent une configuration commune :

```
<!-- Shared configuration in applicationHost.config -->
<location path="Default Web Site">
    <system.webServer>
        <asp enableSessionState="true" />
        <caching enabled="true" enableKernelCache="true" />
    </system.webServer>
</location>
```

Le processus de migration exécute les tâches suivantes :

1. Identifie les configurations partagées entre les sites

2. Génère PowerShell des scripts appropriés pour appliquer ces paramètres
3. Maintient la hiérarchie de configuration et l'héritage

Bonnes pratiques

Nous vous recommandons de suivre les meilleures pratiques pour la configuration réseau de votre application migrée. Les groupes suivants fournissent des directives récapitulatives.

Conception en VPC

- Suivez les meilleures AWS pratiques en matière de conception de VPC
- Utiliser des sous-réseaux distincts pour les équilibreurs de charge et les instances EC2
- Implémenter des tables de routage appropriées et NACLs
- Envisagez les points de terminaison VPC pour les services AWS

Haute disponibilité

- Déployer sur plusieurs zones de disponibilité
- Utiliser au moins deux sous-réseaux pour les équilibreurs de charge
- Configurer l'auto-scaling sur AZs
- Mettre en œuvre des bilans de santé appropriés

Sécurité

- Suivez les meilleures pratiques en matière de sécurité
- Utiliser les groupes de sécurité comme contrôle d'accès principal
- Implémenter des listes de contrôle d'accès au réseau (ACLs) pour une sécurité accrue
- Surveiller les journaux de flux VPC

Résolution des problèmes

Les problèmes courants de configuration réseau concernent les domaines suivants. Chaque sujet est suivi d'exemples de commandes permettant d'obtenir plus d'informations sur la configuration du réseau et l'état de santé de votre environnement.

Configuration du sous-réseau

```
# Verify subnet availability
PS C:\migrations_workspace> aws ec2 describe-subnets --subnet-ids subnet-id
```

```
# Check available IP addresses
PS C:\migrations_workspace>aws ec2 describe-subnets --subnet-ids subnet-id --query
 'Subnets[].AvailableIpAddressCount'
```

Accès aux groupes de sécurité

```
# Verify security group rules
PS C:\migrations_workspace> aws ec2 describe-security-groups --group-ids sg-id

# Test network connectivity
PS C:\migrations_workspace> aws ec2 describe-network-interfaces --filters
 Name=group-id,Values=sg-id
```

État de l'équilibreur de charge

```
# Check load balancer health
PS C:\migrations_workspace> aws elbv2 describe-target-health --target-group-arn
 arn:aws:elasticloadbalancing:region:account-id:targetgroup/group-name/group-id
```

Configurations de sécurité et rôles IAM

La eb migrate commande gère les configurations AWS de sécurité via les rôles IAM, les profils d'instance et les rôles de service. La compréhension de ces composants garantit un contrôle d'accès approprié et une conformité en matière de sécurité lors de la migration.

Configuration du profil d'instance

Un profil d'instance sert de conteneur pour un rôle IAM qu'Elastic Beanstalk attache aux instances EC2 de votre environnement. Lors de l'exécutioneb migrate, vous pouvez spécifier un profil d'instance personnalisé :

```
PS C:\migrations_workspace> eb migrate --instance-profile "CustomInstanceProfile"
```

Si vous ne spécifiez aucun profil d'instance, eb migrate crée un profil par défaut avec les autorisations suivantes :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::elasticbeanstalk-*",
                "arn:aws:s3:::elasticbeanstalk-*/*"
            ]
        }
    ]
}
```

Gestion des rôles de service

Un rôle de service permet à Elastic Beanstalk AWS de gérer les ressources en votre nom. Spécifiez un rôle de service personnalisé lors de la migration à l'aide de la commande suivante :

```
PS C:\migrations_workspace> eb migrate --service-role "CustomServiceRole"
```

S'il n'est pas spécifié, eb migrate crée un rôle de service par défaut nommé `aws-elasticbeanstalk-service-role` avec une politique de confiance qui permet à Elastic Beanstalk d'assumer ce rôle. Ce rôle de service est essentiel pour qu'Elastic Beanstalk puisse surveiller l'état de votre environnement et effectuer des mises à jour de plateforme gérées. Le rôle de service nécessite deux politiques gérées :

- `AWSElasticBeanstalkEnhancedHealth`- Permet à Elastic Beanstalk de surveiller l'état de l'instance et de l'environnement à l'aide du système de reporting de santé amélioré
- `AWSElasticBeanstalkManagedUpdates`- Permet à Elastic Beanstalk d'effectuer des mises à jour gérées de la plateforme, y compris la mise à jour des ressources de l'environnement lorsqu'une nouvelle version de plate-forme est disponible

Avec ces politiques, le rôle de service est autorisé à :

- Création et gestion de groupes Auto Scaling
- Création et gestion des équilibreurs de charge des applications
- Importer des journaux sur Amazon CloudWatch
- Gérer les instances EC2

Pour plus d'informations sur les rôles de service, consultez Rôle de service Elastic Beanstalk le manuel Elastic Beanstalk Developer Guide.

Configuration du groupe de sécurité

La eb migrate commande configure automatiquement les groupes de sécurité en fonction de vos liaisons de site IIS. Par exemple, si votre environnement source comporte des sites utilisant les ports 80, 443 et 8081, les résultats de configuration suivants sont les suivants :

```
<site name="Default Web Site">
    <bindings>
        <binding protocol="http" bindingInformation="*:80:" />
        <binding protocol="https" bindingInformation="*:443:" />
    </bindings>
</site>
<site name="InternalAPI">
    <bindings>
        <binding protocol="http" bindingInformation="*:8081:" />
    </bindings>
</site>
```

Le processus de migration exécute les actions suivantes :

- Crée un groupe de sécurité d'équilibrage de charge autorisant le trafic entrant sur les ports 80 et 443 depuis Internet (0.0.0.0/0)
- Crée un groupe de sécurité EC2 autorisant le trafic provenant de l'équilibreur de charge
- Configure des ports supplémentaires (tels que 8081) si cela est spécifié `--copy-firewall-config`

Par défaut, l'Application Load Balancer est configuré avec un accès public depuis Internet. Si vous devez personnaliser ce comportement, par exemple en restreignant l'accès à des plages d'adresses

IP spécifiques ou en utilisant un équilibreur de charge privé, vous pouvez remplacer la configuration par défaut du VPC et du groupe de sécurité à l'aide du paramètre : `--vpc-config`

```
PS C:\migrations_workspace> eb migrate --vpc-config vpc-config.json
```

Par exemple, la `vpc-config.json` configuration suivante crée un équilibreur de charge privé dans un sous-réseau privé :

```
{
    "id": "vpc-12345678",
    "publicip": "false",
    "elbscheme": "internal",
    "ec2subnets": ["subnet-private1", "subnet-private2"],
    "elbsubnets": ["subnet-private1", "subnet-private2"]
}
```

Pour plus d'informations sur les options de configuration VPC, consultez. Configuration VPC

Intégration des certificats SSL

Lorsque vous migrez des sites avec des liaisons HTTPS, intégrez les certificats SSL via AWS Certificate Manager (ACM) :

```
PS C:\migrations_workspace> eb migrate --ssl-certificates
 "arn:aws:acm:region:account:certificate/certificate-id"
```

Cette configuration permet de réaliser les actions suivantes :

- Associe le certificat à l'Application Load Balancer
- Maintient la terminaison HTTPS au niveau de l'équilibreur de charge
- Préserve la communication HTTP interne entre l'équilibreur de charge et les instances EC2

Authentification Windows

Pour les applications utilisant l'authentification Windows, eb migrate préserve les paramètres d'authentification de l'application `web.config` comme suit :

```
<configuration>
```

```
    <system.webServer>
        <security>
            <authentication>
                <windowsAuthentication enabled="true">
                    <providers>
                        <add value="Negotiate" />
                        <add value="NTLM" />
                    </providers>
                </windowsAuthentication>
            </authentication>
        </security>
    </system.webServer>
</configuration>
```

> **Important**
>
> La eb migrate commande ne copie pas les profils ou comptes utilisateur de votre environnement source vers les instances Elastic Beanstalk cibles. Tous les comptes ou groupes d'utilisateurs personnalisés que vous avez créés sur votre serveur source devront être recréés sur l'environnement cible après la migration.

Les comptes `IUSR` et groupes Windows intégrés`IIS_IUSRS`, ainsi que tous les autres comptes et groupes intégrés, sont inclus par défaut dans les instances Windows Server cibles. Pour plus d'informations sur les comptes et groupes IIS intégrés, consultez la section Présentation des comptes d'utilisateurs et de groupes intégrés dans IIS dans la documentation Microsoft.

Si votre application repose sur des comptes utilisateur Windows personnalisés ou sur l'intégration d'Active Directory, vous devrez configurer ces aspects séparément une fois la migration terminée.

Bonnes pratiques et dépannage

Gestion des rôles

Mettez en œuvre les meilleures pratiques AWS IAM lors de la gestion des rôles dans vos environnements Elastic Beanstalk :

Création et gestion des rôles

- Créez des rôles à l'aide de politiques AWS gérées dans la mesure du possible
- Suivez les meilleures pratiques de sécurité IAM

- Utilisez le générateur AWS de politiques pour des politiques personnalisées
- Implémenter des limites d'autorisation pour une sécurité accrue

Surveillance et audit

Activez AWS CloudTrail pour surveiller l'utilisation des rôles :

- Suivez le guide de AWS CloudTrail l'utilisateur
- Configurer l'intégration CloudWatch des journaux pour une surveillance en temps réel
- Configurer des alertes pour les appels d'API non autorisés

Processus de révision régulier

Établissez un cycle de révision trimestriel pour effectuer les tâches suivantes :

- Auditez les autorisations non utilisées à l'aide d'IAM Access Analyzer
- Supprimer les autorisations périmées
- Mettre à jour les rôles en fonction du principe du moindre privilège

Gestion des certificats

Mettez en œuvre les pratiques suivantes pour les SSL/TLS certificats dans vos environnements Elastic Beanstalk :

Cycle de vie des certificats

- Utilisation AWS Certificate Managerpour la gestion des certificats
- Activer le renouvellement automatique des certificats émis par ACM
- Configurer les notifications d'expiration

Normes de sécurité

- Utiliser TLS 1.2 ou version ultérieure
- Respectez les politiques de AWS sécurité pour les écouteurs HTTPS
- Mettre en œuvre le protocole HTTP Strict Transport Security (HSTS) si nécessaire

Gestion des groupes de sécurité

Mettez en œuvre les meilleures pratiques suivantes en matière de groupes de sécurité :

Gestion des règles

- Documentez toutes les exigences relatives aux ports personnalisés
- Utiliser les journaux de flux VPC pour surveiller le trafic
- Utilisez les règles de référence des groupes de sécurité plutôt que les plages d'adresses IP dans la mesure du possible

Audit régulier

Établissez des révisions mensuelles pour effectuer les tâches suivantes :

- Identifier et supprimer les règles non utilisées
- Valider source/destination les exigences
- Vérifiez les règles qui se chevauchent

Journalisation et surveillance

Pour une surveillance efficace de la sécurité, configurez les journaux suivants :

Journaux d'événements Windows sur les instances EC2

```
# Review Security event log
PS C:\migrations_workspace> Get-EventLog -LogName Security -Newest 50

# Check Application event log
PS C:\migrations_workspace> Get-EventLog -LogName Application -Source "IIS*"
```

CloudWatch Intégration des journaux

Configurez l'agent CloudWatch Logs pour qu'il diffuse les journaux d'événements Windows vers CloudWatch des fins de surveillance et d'alerte centralisées.

Pour les problèmes persistants, collectez ces journaux et contactez-nous AWS Support avec les informations suivantes :

- ID de l'environnement
- ID de déploiement (le cas échéant)
- Messages d'erreur pertinents

- Chronologie des modifications de sécurité

Comprendre le mappage de migration entre IIS et Elastic Beanstalk

La migration d'IIS vers Elastic Beanstalk implique de mapper la configuration de votre serveur Windows sur site aux ressources du cloud. AWS La compréhension de ce mappage est essentielle à la réussite des migrations et à la gestion post-migration.

Sites et applications IIS dans Elastic Beanstalk

Dans IIS, un site Web représente un ensemble d'applications Web et de répertoires virtuels, chacun ayant sa propre configuration et son propre contenu. Lors de la migration vers Elastic Beanstalk, ces composants sont transformés comme suit :

Sites Web IIS

Vos sites Web IIS deviennent des applications au sein d'Elastic Beanstalk. La configuration de chaque site Web, y compris ses liaisons, ses pools d'applications et ses paramètres d'authentification, est préservée via le manifeste de déploiement d'Elastic Beanstalk (). `aws-windows-deployment-manifest.json`

Par exemple, si vous avez plusieurs sites tels que Default Web Site et IntranetSite, regroupe eb migrate le contenu et la configuration de chaque site tout en préservant leur isolation.

La commande crée des règles d'écoute Application Load Balancer (ALB) appropriées pour gérer les demandes de routage vers vos applications. Il configure également les groupes de sécurité pour garantir un accès aux ports approprié en fonction de vos liaisons IIS d'origine.

Pools d'applications

Les pools d'applications IIS fournissent des fonctionnalités d'isolation des processus de travail, de gestion du temps d'exécution et de recyclage pour vos applications. Dans Elastic Beanstalk, ils sont mappés aux processus environnementaux définis `aws:elasticbeanstalk:environment:process` via l'espace de noms et configurés via IIS sur les instances. EC2

La migration préserve les paramètres critiques du pool d'applications, notamment les suivants :

- Configurations des modèles de processus : identité (ApplicationPoolIdentityou comptes personnalisés), paramètres de délai d'inactivité et intervalles de recyclage des processus NetworkService

- Paramètres de version .NET CLR - Maintient la version de .NET Framework que vous avez spécifiée (v2.0, v4.0 ou aucun code managé) pour garantir la compatibilité des applications
- Mode pipeline géré : préserve les paramètres du mode pipeline intégré ou classique pour conserver votre architecture de traitement des requêtes HTTP
- Paramètres avancés : longueur de la file d'attente, limites du processeur, seuils de protection en cas de défaillance rapide et limites de temps de démarrage

La eb migrate commande préserve les mappages entre les sites et les pools d'applications pendant la migration vers votre environnement Elastic Beanstalk.

Si vos pools d'applications utilisent des programmes de recyclage personnalisés (durées ou seuils de mémoire spécifiques), ceux-ci sont mis en œuvre par le biais de PowerShell scripts du package de déploiement qui configurent les paramètres IIS appropriés sur les EC2 instances.

Liaisons du site Web

Les liaisons de site Web IIS, qui définissent la manière dont les clients accèdent à vos applications, sont transformées en configurations Application Load Balancer (ALB) suivantes :

- Les liaisons de port sont mappées aux règles d'écoute ALB correspondantes
- Les configurations d'en-tête de l'hôte sont traduites en règles de routage ALB
- Les sites compatibles SSL utilisent AWS Certificate Manager (ACM) pour la gestion des certificats

Gestion des répertoires virtuels et des chemins d'application

Les répertoires virtuels et les applications IIS fournissent un mappage de chemins d'URL vers des répertoires physiques. Elastic Beanstalk maintient ces relations par le biais des structures suivantes :

Répertoires virtuels

Le processus de migration préserve les chemins physiques de vos répertoires virtuels dans le package de déploiement.

Les mappages de chemins sont configurés dans la configuration IIS sur les EC2 instances, ce qui garantit que votre structure d'URL reste intacte après la migration.

3 chemins physiques du lecteur

> **Important**
>
> Par défaut, les environnements Windows Elastic Beanstalk fournissent uniquement le lecteur C : \ (volume racine). Dans la version actuelle, les applications dont le contenu se trouve sur des lecteurs (D : \, E : \, etc.) ne sont pas prises en charge pour la migration.

La eb migrate commande détecte automatiquement les chemins physiques situés sur les lecteurs et vous avertit en cas de problème potentiel, comme dans l'exemple suivant :

```
ERROR: Detected physical paths on drive D:\ which are not supported in the current
 version:
  - D:\websites\intranet
  - D:\shared\images

Migration of content from non-system drives is not supported. Please relocate this
 content to the C:\ drive before migration. Otherwise, select only those sites that
 are on C:\.
```

Si votre application dépend de lecteurs, vous devrez modifier votre application pour stocker tout le contenu sur le lecteur C : \ avant la migration.

Applications imbriquées

Les applications imbriquées sous des sites Web sont déployées avec leurs configurations de chemin correctes et leurs affectations de pool d'applications appropriées. Le processus de migration préserve tous les `web.config` paramètres, garantissant ainsi que les configurations spécifiques aux applications continuent de fonctionner comme prévu dans l'environnement cloud.

Réécriture d'URL et routage des demandes d'applications (ARR)

Si votre déploiement IIS utilise la réécriture d'URL ou le routage des demandes d'application (ARR), eb migrate gère ces configurations selon les règles et la configuration suivantes :

Règles de réécriture d'URL

Les règles de réécriture d'URL de vos `web.config` fichiers sont traduites en règles de routage ALB dans la mesure du possible. Par exemple, l'entrée suivante devient une règle d'écoute ALB dirigeant le trafic en fonction des en-têtes de l'hôte et des modèles de chemin. :

```
<!-- Original IIS URL Rewrite Rule -->
<rule name="Redirect to WWW" stopProcessing="true">
    <match url="(.*)" />
    <conditions>
        <add input="{HTTP_HOST}" pattern="^example.com$" />
    </conditions>
    <action type="Redirect" url="http://www.example.com/{R:1}" />
</rule>
```

Routage des demandes d'application

Les configurations ARR sont préservées grâce à l'installation de fonctionnalités ARR sur EC2 les instances. Le processus de migration exécute les tâches suivantes :

- Configure les paramètres du proxy en fonction de votre environnement source
- Maintient les règles de réécriture d'URL associées à l'ARR

Structure des artefacts de migration

Lorsque vous l'exécutezeb migrate, il crée un répertoire structuré contenant tous les composants de déploiement nécessaires. La liste suivante décrit la structure du répertoire :

```
C:\migration_workspace\
### .\migrations\latest\
    ### upload_target\
        ### [SiteName].zip                # One ZIP per IIS site
        ### aws-windows-deployment-manifest.json
        ### ebmigrateScripts\
            ### site_installer.ps1         # Site installation scripts
            ### arr_configuration.ps1      # ARR configuration scripts
            ### permission_handler.ps1     # Permission management
            ### firewall_config.ps1        # Windows Firewall rules
```

Le `aws-windows-deployment-manifest.json` fichier est le fichier de configuration de base qui indique à Elastic Beanstalk comment déployer vos applications. Reportez-vous à l'exemple de structure suivant :

```
{
    "manifestVersion": 1,
    "deployments": {
        "msDeploy": [
            {
                "name": "Primary Site",
                "parameters": {
                    "appBundle": "DefaultWebSite.zip",
                    "iisPath": "/",
                    "iisWebSite": "Default Web Site"
                }
            }
        ],
        "custom": [
            {
                "name": "ConfigureARR",
                "scripts": {
                    "install": {
                        "file": "ebmigrateScripts\\arr_configuration.ps1"
                    },
                    "uninstall": {
                        "file": "ebmigrateScripts\\noop.ps1"
                    },
                    "restart": {
                        "file": "ebmigrateScripts\\noop.ps1"
                    }
                }
            }
        ]
    }
}
```

Ce manifeste garantit les résultats suivants pour votre migration :

- Les applications sont déployées pour corriger les chemins IIS
- Des configurations personnalisées sont appliquées
- Les paramètres spécifiques au site sont préservés

- L'ordre de déploiement est maintenu

Scénarios de migration avancés

Cette section couvre les scénarios de migration avancés pour les déploiements IIS complexes.

Migrations multisites avec routage des demandes d'application (ARR)

La eb migrate commande détecte et préserve automatiquement les configurations ARR pendant la migration. Lorsqu'il identifie les paramètres ARR dans votre IIS`applicationHost.config`, il génère les PowerShell scripts nécessaires pour réinstaller et configurer ARR sur les EC2 instances cibles.

Détection de configuration ARR

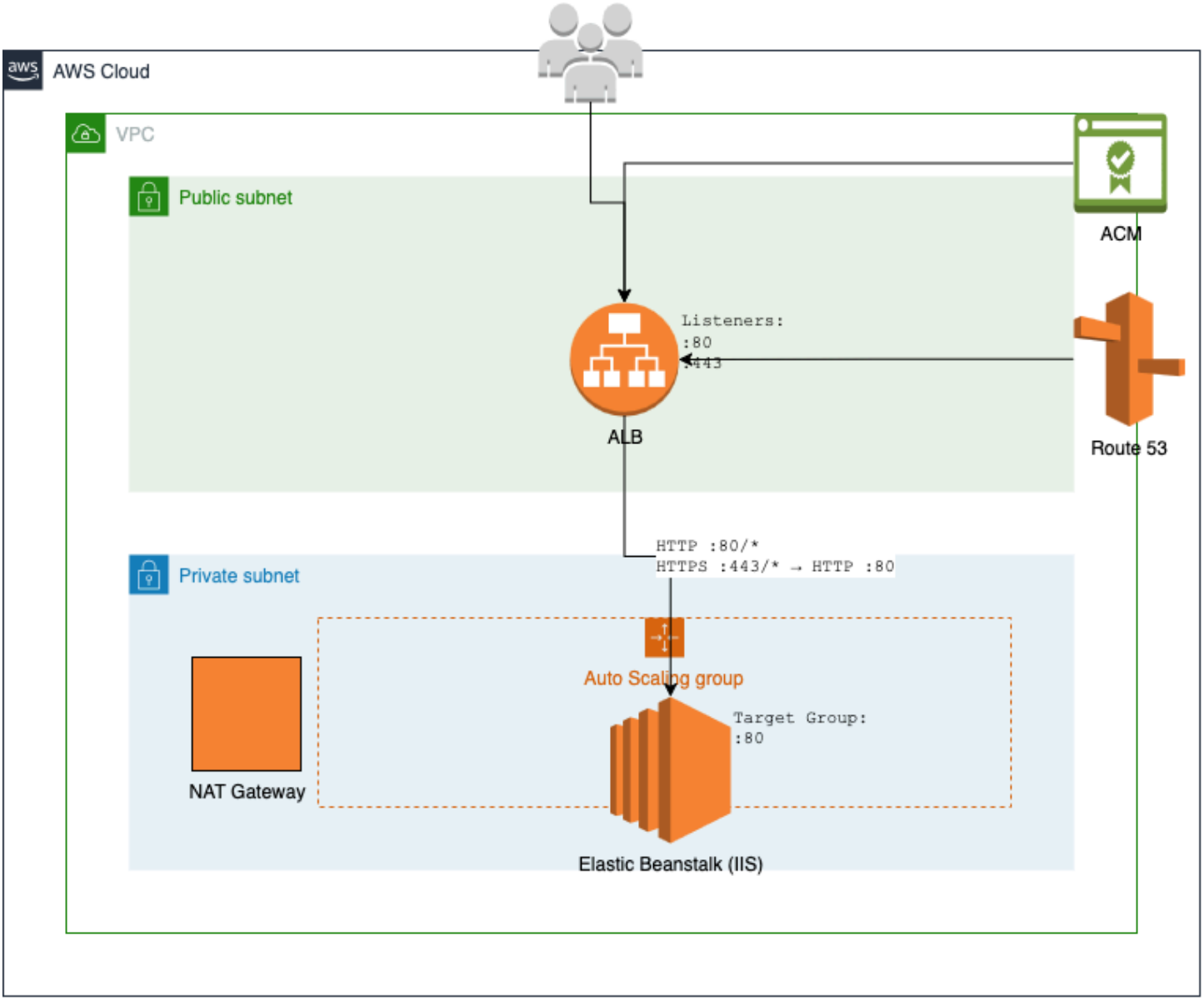
Le processus de migration examine trois sections de configuration clés dans IIS :

- `system.webServer/proxy`: Paramètres du proxy ARR de base
- `system.webServer/rewrite`: règles de réécriture d'URL
- `system.webServer/caching`: Configuration de la mise en cache

Par exemple, considérez une configuration ARR courante dans laquelle une application `RouterSite` exécutée sur le port 80 envoie des requêtes aux ports 8081 `APIService` et 8082 et `AdminPortal` s'exécute respectivement sur ces ports :

```
<!-- Original IIS ARR Configuration -->
<rewrite>
    <rules>
        <rule name="Route to API" stopProcessing="true">
            <match url="^api/(.*)$" />
            <action type="Rewrite" url="http://backend:8081/api/{R:1}" />
        </rule>
        <rule name="Route to Admin" stopProcessing="true">
            <match url="^admin/(.*)$" />
            <action type="Rewrite" url="http://backend:8082/admin/{R:1}" />
        </rule>
    </rules>
</rewrite>
```

Le schéma suivant montre comment ces règles sont masquées derrière le port 80 sur le serveur IIS et ne sont pas exposées via les groupes EC2 de sécurité. Seul le port 80 est accessible à l'Application Load Balancer et tout le trafic provenant de celui-ci est acheminé vers le groupe cible sur le port 80.



La commande suivante peut faire migrer cette configuration :

```
PS C:\migrations_workspace> eb migrate --sites "RouterSite,APIService,AdminPortal" `
    --copy-firewall-config
```

Processus de migration ARR

Le processus de migration préserve votre configuration ARR en plusieurs étapes.

Exportation de configuration

L'outil exporte vos paramètres ARR existants depuis les trois sections de configuration clés dans des fichiers XML distincts stockés dans le `ebmigrateScripts` répertoire :

```
ebmigrateScripts\
### arr_config_proxy.xml
### arr_config_rewrite.xml
### arr_config_caching.xml
```

Scripts d'installation

Deux PowerShell scripts sont générés pour gérer la configuration de l'ARR :

1. `arr_msi_installer.ps1`: télécharge et installe le module ARR
2. `arr_configuration_importer_script.ps1`: Importe votre configuration ARR exportée

Intégration du manifeste de déploiement

Les scripts sont intégrés au processus de déploiement par le biais d'entrées dans `aws-windows-deployment-manifest.json` :

```
{
    "manifestVersion": 1,
    "deployments": {
        "custom": [
            {
                "name": "WindowsProxyFeatureEnabler",
                "scripts": {
                    "install": {
                        "file": "ebmigrateScripts\
\windows_proxy_feature_enabler.ps1"
                    }
                }
            },
            {
                "name": "ArrConfigurationImporterScript",
                "scripts": {
                    "install": {
```

```
                        "file": "ebmigrateScripts\
\arr_configuration_importer_script.ps1"
                    }
                }
            }
        ]
    }
}
```

Intégration de l'équilibreur de charge

Le processus de migration traduit vos règles ARR en règles d'écoute Application Load Balancer (ALB) dans la mesure du possible. Par exemple, la configuration ARR ci-dessus donne lieu à des règles ALB qui acheminent le trafic en fonction des modèles de chemin d'URL tout en maintenant le routage interne sur les EC2 instances.

L'environnement qui en résulte conserve votre logique de routage ARR tout en tirant parti de l'infrastructure élastique AWS de l'ARR. Vos applications continuent de fonctionner comme avant, ARR gérant le routage interne tandis que l'Application Load Balancer gère la distribution du trafic externe.

Migrations multisites sans ARR à l'aide d'un routage basé sur l'hôte

Bien que le routage des demandes d'application (ARR) soit une approche courante pour gérer plusieurs sites dans IIS, vous pouvez également migrer des déploiements multisites directement vers Elastic Beanstalk sans ARR en tirant parti des fonctionnalités de routage basées sur l'hôte de l'Application Load Balancer. Cette approche permet de réduire la complexité et d'améliorer les performances en éliminant une couche de routage supplémentaire.

Vue d'ensemble du routage basé sur l'hôte

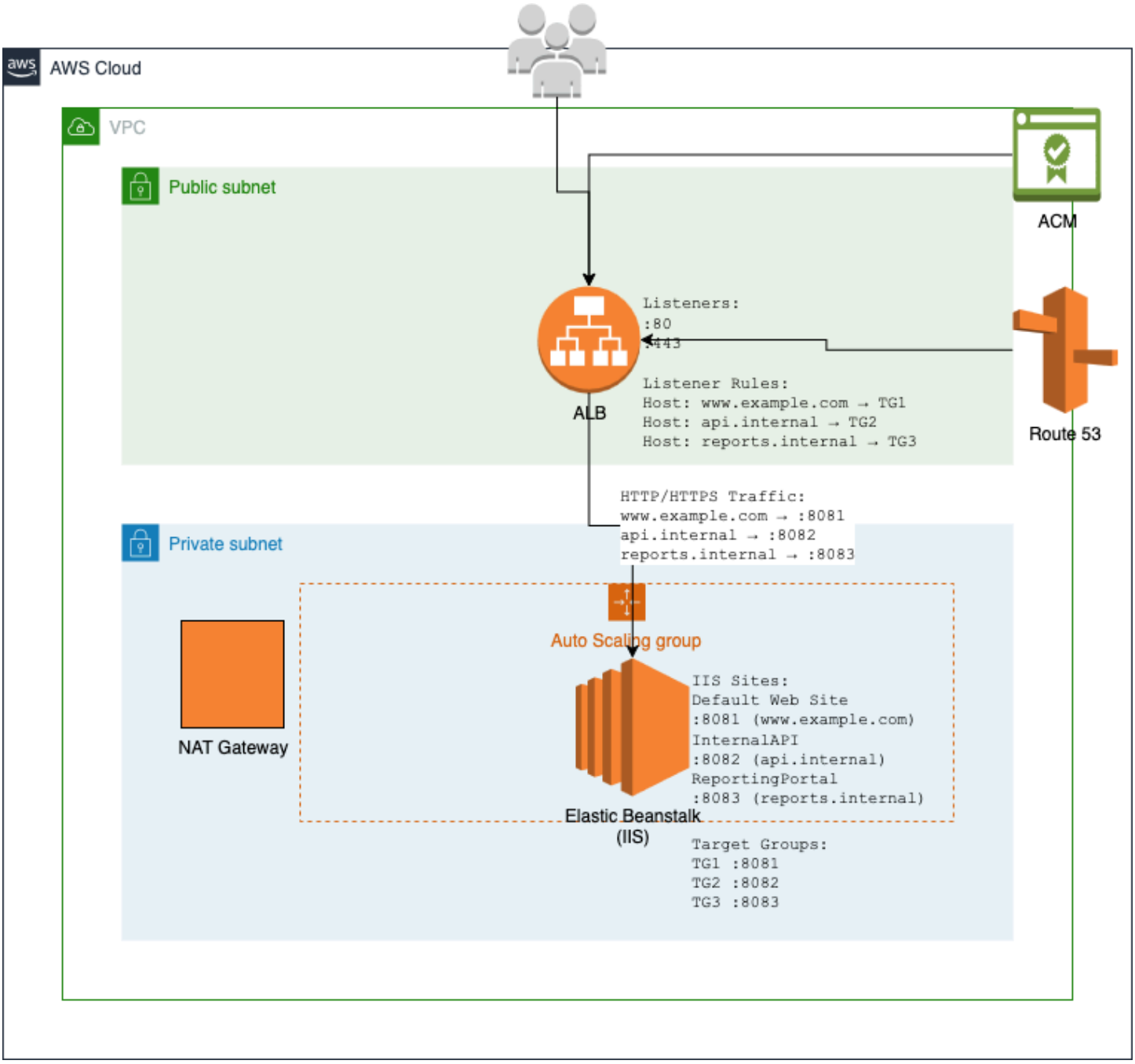
Dans cette approche, chaque site IIS est exposé en dehors de l' EC2 instance, et l'Application Load Balancer achemine le trafic directement vers le port approprié en fonction de l'en-tête de l'hôte. Cela élimine le besoin d'ARR tout en maintenant la séparation entre vos applications.

Prenons l'exemple d'une configuration IIS multisite avec trois sites, chacun ayant sa propre liaison de nom d'hôte :

```
<sites>
```

```
    <site name="Default Web Site" id="1">
        <bindings>
            <binding protocol="http" bindingInformation="*:8081:www.example.com" />
        </bindings>
    </site>
    <site name="InternalAPI" id="2">
        <bindings>
            <binding protocol="http" bindingInformation="*:8082:api.internal" />
        </bindings>
    </site>
    <site name="ReportingPortal" id="3">
        <bindings>
            <binding protocol="http" bindingInformation="*:8083:reports.internal" />
        </bindings>
    </site>
</sites>
```

Ces sites sont exposés sur les ports 8081, 8082 et 8083 via les EC2 groupes de sécurité. L'Application Load Balancer les achemine vers eux en fonction de la configuration des règles de l'écouteur Load Balancer.



Processus de migration

Pour migrer cette configuration vers Elastic Beanstalk sans eb migrate utiliser ARR, utilisez la commande de l'exemple suivant :

```
PS C:\migrations_workspace> eb migrate --sites "Default Web
 Site,InternalAPI,ReportingPortal"
```

Le processus de migration configure automatiquement l'Application Load Balancer avec des règles de routage basées sur l'hôte qui dirigent le trafic vers le groupe cible approprié en fonction de l'en-tête de l'hôte. Chaque groupe cible transmet le trafic vers le port correspondant sur vos EC2 instances :

1. En-tête de l'hôte www.example.com → Groupe cible sur le port 8081
2. En-tête d'hôte api.internal → Groupe cible sur le port 8082
3. En-tête de l'hôte reports.internal → Groupe cible sur le port 8083

Configuration SSL/TLS

Pour sécuriser vos applications, SSL/TLS procédez comme suit :

1. Demandez des certificats pour vos domaines via AWS Certificate Manager(ACM).
2. Configurez les écouteurs HTTPS sur votre Application Load Balancer à l'aide de ces certificats.
3. Mettez à jour la configuration de votre environnement pour inclure les écouteurs HTTPS avec les paramètres d'options de configuration suivants.

```
option_settings:
  aws:elb:listener:443:
    ListenerProtocol: HTTPS
    SSLCertificateId: arn:aws:acm:region:account-id:certificate/certificate-id
    InstancePort: 80
    InstanceProtocol: HTTP
```

Avec cette configuration, la terminaison SSL se produit au niveau de l'équilibreur de charge et le trafic est transféré vers vos instances via HTTP. Cela simplifie la gestion des certificats tout en maintenant des connexions sécurisées avec les clients.

Bonnes pratiques

Groupes de sécurité

> Configurez les groupes de sécurité pour autoriser le trafic entrant uniquement sur les ports utilisés par vos sites IIS (8081, 8082, 8083 dans cet exemple) à partir du groupe de sécurité Application Load Balancer.

Surveillance de l'état

Configurez les contrôles de santé pour chaque groupe cible afin de garantir que le trafic est uniquement acheminé vers des instances saines. Créez des points de terminaison de contrôle de santé pour chaque application s'ils n'existent pas déjà.

Contrôle

Configurez des CloudWatch alarmes pour surveiller la santé et les performances de chaque groupe cible séparément. Cela vous permet d'identifier les problèmes spécifiques aux applications individuelles.

Mise à l’échelle

Tenez compte des besoins en ressources de toutes les applications lors de la configuration des politiques d'autodimensionnement. Si les besoins en ressources d'une application sont très différents, envisagez de la migrer vers un environnement distinct.

Gestion des annuaires virtuels

La eb migrate commande préserve les structures de répertoires virtuels lors de la migration de vos applications IIS vers Elastic Beanstalk.

Configuration des autorisations par défaut

Lors de la migration de répertoires virtuels, eb migrate établit un ensemble d'autorisations de base en accordant l' ReadAndExecute accès à :

- IIS_IUSRS
- IUSR
- Utilisateurs authentifiés

Par exemple, considérez une structure de répertoire virtuel typique :

```
<site name="CorporatePortal">
    <application path="/" applicationPool="CorporatePortalPool">
        <virtualDirectory path="/" physicalPath="C:\sites\portal" />
        <virtualDirectory path="/shared" physicalPath="C:\shared\content" />
        <virtualDirectory path="/reports" physicalPath="D:\reports" />
    </application>
```

```
</site>
```

Répertoires virtuels protégés par mot de passe

Lorsqu'il eb migrate rencontre des répertoires virtuels protégés par mot de passe, il émet des avertissements et nécessite une intervention manuelle.

L'exemple de configuration suivant provoquera la réponse d'avertissement qui suit l'exemple.

```
<virtualDirectory path="/secure"
                  physicalPath="C:\secure\content"
                  userName="DOMAIN\User"
                  password="[encrypted]" />
```

```
[WARNING] CorporatePortal/secure is hosted at C:\secure\content which is password-
protected and won't be copied.
```

Pour garantir la protection par mot de passe, créez un script de déploiement personnalisé tel que le suivant :

```
# PS C:\migrations_workspace> cat secure_vdir_config.ps1

$vdirPath = "C:\secure\content"
$siteName = "CorporatePortal"
$vdirName = "secure"
$username = "DOMAIN\User"
$password = "SecurePassword"

# Ensure directory exists
if (-not (Test-Path $vdirPath)) {
    Write-Host "Creating directory: $vdirPath"
    New-Item -Path $vdirPath -ItemType Directory -Force
}

# Configure virtual directory with credentials
Write-Host "Configuring protected virtual directory: $vdirName"
New-WebVirtualDirectory -Site $siteName -Name $vdirName `
    -PhysicalPath $vdirPath -UserName $username -Password $password

# Set additional permissions as needed
$acl = Get-Acl $vdirPath
```

```
$rule = New-Object System.Security.AccessControl.FileSystemAccessRule(
    $username, "ReadAndExecute", "ContainerInherit,ObjectInherit", "None", "Allow"
)
$acl.AddAccessRule($rule)
Set-Acl $vdirPath $acl
```

Ajoutez ce script à votre déploiement en l'incluant dans le manifeste :

```
{
    "manifestVersion": 1,
    "deployments": {
        "custom": [
            {
                "name": "SecureVirtualDirectory",
                "scripts": {
                    "install": {
                        "file": "secure_vdir_config.ps1"
                    }
                }
            }
        ]
    }
}
```

Gestion personnalisée des autorisations

La eb migrate commande fournit un cadre pour les scripts d'autorisation personnalisés afin de prendre en charge les applications qui nécessitent des autorisations autres que celles par défaut.

```
$paths = @(
    "C:\sites\portal\uploads",
    "C:\shared\content"
)

foreach ($path in $paths) {
    if (-not (Test-Path $path)) {
        Write-Host "Creating directory: $path"
        New-Item -Path $path -ItemType Directory -Force
    }

    $acl = Get-Acl $path
```

```
    # Add custom permissions
    $customRules = @(
        # Application Pool Identity - Full Control
        [System.Security.AccessControl.FileSystemAccessRule]::new(
            "IIS AppPool\CorporatePortalPool",
            "FullControl",
            "ContainerInherit,ObjectInherit",
            "None",
            "Allow"
        ),
        # Custom Service Account
        [System.Security.AccessControl.FileSystemAccessRule]::new(
            "NT SERVICE\CustomService",
            "Modify",
            "ContainerInherit,ObjectInherit",
            "None",
            "Allow"
        )
    )

    foreach ($rule in $customRules) {
        $acl.AddAccessRule($rule)
    }

    Set-Acl $path $acl
    Write-Host "Custom permissions applied to: $path"
}
```

Bonnes pratiques

Suivez ces bonnes pratiques pour planifier, exécuter, surveiller et vérifier votre migration.

Planification préalable à la migration

Documentez les autorisations existantes et les exigences d'authentification avant la migration. Testez des scripts d'autorisation personnalisés dans un environnement de développement avant de les déployer en production.

Gestion de contenu partagé

Pour les répertoires de contenu partagé, assurez-vous que toutes les autorisations nécessaires au système de fichiers sont correctement configurées par le biais de scripts personnalisés. Envisagez d'utiliser Amazon FSx pour Windows File Server pour les besoins de stockage partagé.

Surveillance et vérification

Surveillez les journaux des applications après la migration pour vérifier l'accès approprié aux répertoires virtuels. Portez une attention particulière aux domaines suivants :

- Accès aux identités du pool d'applications
- Autorisations de compte de service personnalisées
- Connectivité de partage réseau
- Authentication failures (Échecs d’authentification)

Paramètres personnalisés du pool d'applications

La eb migrate commande ne copie pas les paramètres personnalisés du pool d'applications par défaut. Pour conserver les configurations personnalisées du pool d'applications, suivez cette procédure pour créer et appliquer une section de manifeste personnalisée.

1. Créez une archive de vos artefacts de migration.

```
PS C:\migrations_workspace> eb migrate --archive
```

2. Créez un PowerShell script personnalisé pour configurer les pools d'applications.

```
# PS C:\migrations_workspace> cat .\migrations\latest\upload_target
\customize_application_pool_config.ps1

$configPath = "$env:windir\System32\inetsrv\config\applicationHost.config"

[xml]$config = Get-Content -Path $configPath

$newPoolXml = @"
<!-- Original IIS Configuration -->
<applicationPools>
    <add name="CustomPool"
         managedRuntimeVersion="v4.0"
         managedPipelineMode="Integrated">
        <processModel identityType="SpecificUser"
                      userName="AppPoolUser"
                      password="[encrypted]" />
        <recycling>
            <periodicRestart time="00:00:00">
                <schedule>
```

```
                    <add value="02:00:00" />
                    <add value="14:00:00" />
                </schedule>
            </periodicRestart>
        </recycling>
    </add>
</applicationPools>
"@
$newPoolXmlNode = [xml]$newPoolXml

# Find the applicationPools section
$applicationPools = $config.SelectSingleNode("//configuration/
system.applicationHost/applicationPools")

# Import the new node into the document
$importedNode = $config.ImportNode($newPoolXmlNode.DocumentElement, $true)
$applicationPools.AppendChild($importedNode)

# Save the changes
$config.Save($configPath)

Write-Host "ApplicationHost.config has been updated successfully."
```

3. Mettez à jour le `aws-windows-deployment-manifest.json` fichier pour inclure votre script personnalisé.

```
{
    "manifestVersion": 1,
    "deployments": {
        ...
        "custom": [
            ...,
            {
                "name": "ModifyApplicationPoolConfig",
                "description": "Modify application pool configuration from source
 machine to remove",
                "scripts": {
                    "install": {
                        "file": "customize_application_pool_config.ps1"
                    },
                    "restart": {
                        "file": "ebmigrateScripts\\noop.ps1"
                    },
```

```
                    "uninstall": {
                        "file": "ebmigrateScripts\\noop.ps1"
                    }
                }
            }
        ]
    }
}
```

4. Créez un environnement avec le répertoire d'archives mis à jour.

```
PS C:\migrations_workspace> eb migrate `
    --archive-dir '.\migrations\latest\upload_target\'
```

L'`--archive-dir`argument indique eb migrate d'utiliser le code source qu'il a créé précédemment, en évitant la création de nouvelles archives.

Déploiement des versions précédentes

eb migrateconserve un historique de vos migrations via des annuaires horodatés et des versions d'applications dans Elastic Beanstalk. Chaque migration crée un fichier zip unique qui peut être déployé si nécessaire.

```
PS C:\migrations_workspace> ls .\migrations\

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d----l        3/18/2025  10:34 PM                latest
d-----        3/16/2025   5:47 AM                migration_1742104049.479849
d-----        3/17/2025   9:18 PM                migration_1742246303.18056
d-----        3/17/2025   9:22 PM                migration_1742246546.565739
...
d-----        3/18/2025  10:34 PM                migration_1742337258.30742
```

Le lien `latest` symbolique pointe toujours vers le répertoire des artefacts de migration le plus récemment créé. Outre les journaux d'applications et d'erreurs pertinents, chaque répertoire d'artefacts de migration contient également un `upload_target.zip` fichier que vous pouvez déployer sur Elastic Beanstalk.

```
PS C:\migrations_workspace> ls .\migrations\latest\
```

```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        3/18/2025  10:34 PM                upload_target
-a----        3/18/2025  10:34 PM          13137 application.log
-a----        3/18/2025  10:34 PM              0 error.log
-a----        3/18/2025  10:34 PM        1650642 upload_target.zip
```

Vous pouvez déployer le `upload_target.zip` fichier en utilisant eb migrate :

```
PS C:\migrations_workspace> eb migrate --zip .\migrations\latest\upload_target.zip
```

Dépannage et diagnostic

> **Essayez Amazon Q Developer CLI pour un dépannage assisté par l'IA**
>
> La CLI Amazon Q Developer peut vous aider à résoudre rapidement les problèmes d'environnement. La Q CLI fournit des solutions en vérifiant l'état de l'environnement, en examinant les événements, en analysant les journaux et en posant des questions de clarification. Pour plus d'informations et des instructions détaillées, consultez la section Résolution des problèmes liés aux environnements Elastic Beanstalk avec Amazon Q Developer CLI dans les blogs. AWS

Cette section fournit des conseils pour résoudre les problèmes courants susceptibles de survenir lors de la migration des applications IIS vers Elastic Beanstalk.

Associer une EC2 paire de clés à votre environnement

Vous pouvez vous connecter en toute sécurité aux instances Amazon Elastic Compute Cloud (Amazon EC2) mises en service pour votre application Elastic Beanstalk à l'aide d'une paire de clés Amazon. EC2 Pour obtenir des instructions sur la création d'une paire de clés, consultez la section Création d'une paire de clés à l'aide d'Amazon EC2 dans le guide de EC2 l'utilisateur Amazon.

La spécification d'un nom de clé eb migrate a pour effet d'associer votre environnement Elastic Beanstalk à la paire de clés. Pour des raisons de sécurité, cela n'ouvrira pas le port 3389 sur le groupe de sécurité de votre EC2 instance. Vous pouvez associer des groupes EC2 de sécurité supplémentaires autorisant le trafic du port 3389 à passer eb config après la migration initiale.

```
PS C:\migrations_workspace> eb migrate  `
    --keyname "my-keypair"  `
    --verbose
```

Lorsque vous créez une paire de clés, Amazon EC2 stocke une copie de votre clé publique. Si vous n'avez plus besoin de l'utiliser pour vous connecter à des instances d'environnement, vous pouvez le supprimer d'Amazon EC2. Pour plus de détails, consultez Supprimer votre paire de clés dans le guide de EC2 l'utilisateur Amazon.

Pour plus d'informations sur la connexion aux EC2 instances Windows Amazon, consultez Connexion à une instance Windows.

Accès aux journaux

L'EB CLI fournit une eb logs fonctionnalité que vous pouvez utiliser pour récupérer les journaux d'un environnement Elastic Beanstalk sans vous connecter à ses instances. EC2 Après l'exécution deeb migrate, vous pouvez lancer la eb logs --zip commande qui téléchargera et enregistrera les journaux dans le `.elasticbeanstalk\logs` répertoire.

Vous pouvez également consulter les journaux via la console AWS Elastic Beanstalk. Pour de plus amples informations, veuillez consulter Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk.

Accès aux artefacts côté client

La eb migrate commande stocke les journaux d'applications et d'erreurs générés par les répertoires d'artefacts des migrations msdeploy internes.

```
./migrations/
### latest -> migration_20240308_123456/
### migration_20240308_123456/
    ### application.log
    ### error.log
    ### upload_target\
```

Surveillance de la santé de l'environnement

Elastic Beanstalk vous aide à surveiller l'état de santé à l'aide des fonctionnalités améliorées de surveillance de l'état de santé. Il s'agit d'un système de surveillance automatique de l'état de santé

qui suit en permanence l'état opérationnel des instances d'application, en tirant parti de mesures intégrées telles que l'utilisation du processeur, la latence, le nombre de demandes et les codes de réponse.

Le système de surveillance de l'état utilise une approche basée sur des agents pour collecter des données au niveau de l'instance et intègre la journalisation et les alertes en temps réel. Elastic Load Balancing (ELB) et Auto Scaling répondent de manière dynamique aux changements d'état de santé, garantissant ainsi une disponibilité et une tolérance aux pannes élevées. Les modes de surveillance avancés, notamment les rapports de santé améliorés, fournissent une visibilité précise du comportement des applications, permettant un dépannage proactif et des mécanismes de restauration automatique.

Exécutez la eb health commande EB CLI pour afficher l'état de santé de l'environnement. Les informations suivantes s’affichent :

- État d'une instance
- Métriques de réponse des applications
- Utilisation des ressources du système
- Événements de déploiement récents

EC2 optimisation des performances

Par défaut, eb migrate sélectionne le type d'instance c5.2xlarge pour offrir une première expérience optimale avec Elastic Beanstalk. Vous pouvez modifier ce comportement avec l'--instance-typeargument suivant :

```
PS C:\migrations_workspace> eb migrate `
    --instance-type "t3.large"
```

Pour les environnements de production, tenez compte des facteurs suivants lors de la sélection d'un type d'instance :

- Besoins en mémoire de vos applications
- Exigences relatives au processeur pour le traitement des charges de travail
- Besoins de performance du réseau
- Objectifs d'optimisation des coûts

Configuration du volume EBS

Par défaut, Elastic Beanstalk crée uniquement un volume root `C:\` block-device () pour votre environnement. Vous pouvez transmettre des volumes de snapshots Amazon Elastic Block Store supplémentaires avec l'--ebs-snapshotsoption suivante :

```
PS C:\migrations_workspace> eb migrate `
    --ebs-snapshots "snap-123456789abc"
```

Pour des exemples illustrant la manière dont vous pouvez configurer des mappages par blocs avec Elastic Beanstalk, consultez l'article de blog Customize Ephemeral et EBS volumes in Elastic Beanstalk Environments.

Pour les applications nécessitant beaucoup de stockage, envisagez les options suivantes :

- Utilisation de volumes EBS pour les données persistantes
- Implémentation d'Amazon S3 pour le contenu statique
- Utilisation du serveur de fichiers Amazon FSx pour Windows pour les systèmes de fichiers partagés

Problèmes courants et solutions correspondantes

Événement : installation manquante de Web Deploy

Si vous rencontrez des erreurs liées à l'impossibilité de trouver Web Deploy, installez Web Deploy 3.6 ou version ultérieure à partir du programme d'installation de Microsoft Web Platform. L'exemple suivant affiche un message d'erreur possible.

```
Couldn't find msdeploy.exe. Follow instructions here: https://learn.microsoft.com/en-
us/iis/install/installing-publishing-technologies/installing-and-configuring-web-deploy
```

Événement : Problèmes d'autorisation lors de la migration

Si vous rencontrez des erreurs liées aux autorisations, assurez-vous d'exécuter l'EB CLI avec des privilèges administratifs. L'exemple suivant affiche un message d'erreur possible.

```
[ERROR] Access to the path 'C:\inetpub\wwwroot\web.config' is denied.
```

Événement : Problèmes d'identité du pool d'applications

Si votre application ne démarre pas en raison de problèmes d'identité du pool d'applications, créez un script personnalisé pour configurer les identités du pool d'applications, comme indiqué dansthe section called “Paramètres personnalisés du pool d'applications”.

Evénement : erreurs de configuration du certificat SSL

Si les liaisons HTTPS ne fonctionnent pas, assurez-vous d'avoir spécifié un ARN de certificat ACM valide à l'aide du paramètre eb mibrate option`--ssl-certificates`.

Événement : expiration du délai de création de l'environnement

Si le délai de création de l'environnement est expiré, vérifiez les CloudFormation événements dans la console AWS de gestion pour détecter les échecs de création de ressources spécifiques. Les causes courantes incluent les problèmes de configuration des VPC ou les limites de service.

Obtention de support

Si vous rencontrez des problèmes que vous ne parvenez pas à résoudre, AWS Support collectez les informations suivantes avant de nous contacter :

- ID d'environnement (`eb status`)
- Journaux des applications (`eb logs --zip`)
- Artefacts de migration depuis `.\migrations\latest\`
- Configuration IIS source (sortie de`eb migrate explore --verbose`)
- Messages d'erreur détaillés

Pour plus d'informations sur la résolution des problèmes liés à Elastic Beanstalk, consultez. Résolution des problèmes liés à votre environnement Elastic Beanstalk

Comparaison des options de migration : EB CLI et AWS Application Migration Service

AWS propose plusieurs chemins pour la migration des applications Windows vers le cloud. Cette section compare deux options principales : la eb migrate commande dans l'EB CLI et AWS

Application Migration Service (MGN). Comprendre les différences entre ces approches vous aidera à choisir la stratégie de migration la mieux adaptée à vos besoins spécifiques.

Comparaison des options de migration

Fonctionnalité	CLI WEB (eb migrate)	AWS Application Migration Service (MGN)
Objectif principal	Migration au niveau de l'application de sites Web et d'applications IIS	Réhébergement au niveau du serveur de machines entières (serveurs physiques, virtuels ou cloud)
Idéal pour	Applications IIS que vous souhaitez migrer directement vers Elastic Beanstalk avec une reconfiguration minimale	Migrations à grande échelle impliquant de nombreux serveurs ou une infrastructure complexe
Approche de découverte	Découverte des sites, applications et configurations IIS au niveau de l'application	Réplication au niveau du serveur de machines entières, y compris le système d'exploitation et les applications
Environnement cible	Crée et configure directement des environnements Elastic Beanstalk optimisés pour les applications Windows	Crée EC2 des instances qui nécessitent une configuration supplémentaire pour fonctionner avec Elastic Beanstalk
Préservation de la configuration	Préserve automatiquement les configurations spécifiques à l'IIS (sites, pools d'applications, liaisons)	Préserve la configuration complète du serveur, qui peut inclure des composants inutiles
Modèle de déploiement	Crée un environnement Elastic Beanstalk propre avec vos applications déployées selon les meilleures pratiques d'Elastic Beanstalk	Crée une réplique de votre serveur source qui peut nécessiter une optimisation pour les opérations dans le cloud
Ampleur de la migration	Idéal pour les migrations ciblées d'applications spécifiques	Conçu pour les migrations à grande échelle de nombreux serveurs

Fonctionnalité	CLI WEB (eb migrate)	AWS Application Migration Service (MGN)
Étapes postérieures à la migration	Minimal ; l'environnement est prêt à être utilisé avec les outils de gestion Elastic Beanstalk	Nécessite des étapes supplémentaires pour s'intégrer à Elastic Beanstalk, telles que l'exécution d'actions SSM après le lancement

Quand utiliser chaque option de migration

Choisissez eb migrate lorsque vous répondez aux exigences suivantes :

- Vous souhaitez migrer des applications IIS spécifiques plutôt que des serveurs entiers
- Votre objectif est d'adopter Elastic Beanstalk comme plateforme de gestion des applications
- Vous souhaitez tirer parti des fonctionnalités de la plateforme gérée d'Elastic Beanstalk, telles que la facilité de mise à l'échelle, de déploiement et de surveillance
- Vous préférez un déploiement propre qui suit les AWS meilleures pratiques pour les opérations cloud natives
- Vous souhaitez minimiser le travail de configuration après la migration

Choisissez AWS Application Migration Service lorsque vous répondez aux exigences suivantes :

- Vous devez migrer un grand nombre de serveurs
- Vous avez des configurations de serveur complexes qui doivent être préservées exactement
- Vos applications présentent des problèmes de compatibilité qui nécessitent de maintenir l'environnement de serveur exact
- Vous souhaitez « soulever et déplacer » en apportant un minimum de modifications à vos applications
- Vous envisagez de refactoriser ou d'optimiser vos applications après la migration

Comparaison des flux de travail de migration

Flux de travail EB CLI (eb migrate) :

1. Installez l'interface de ligne de commande EB sur votre serveur IIS source ou sur un hôte bastion.
2. Exécutez eb migrate pour découvrir les applications IIS.

3. La commande regroupe vos applications et configurations.
4. Un environnement Elastic Beanstalk est créé avec les ressources appropriées.
5. Vos applications sont déployées dans le nouvel environnement.
6. Vous pouvez immédiatement gérer vos applications à l'aide des outils Elastic Beanstalk.

AWS Application Migration Service flux de travail :

1. Installez l'agent AWS de réplication sur les serveurs sources.
2. Configurez et testez la réplication des données.
3. Lancez des instances de test pour vérifier le fonctionnement.
4. Planifiez le passage à. AWS
5. Lancez des instances de production.
6. Exécutez des actions après le lancement afin d'optimiser votre solution pour le cloud.
7. Si Elastic Beanstalk est la plate-forme cible, une configuration supplémentaire est requise pour l'intégrer à Elastic Beanstalk.

Conclusion

Elastic Beanstalk est la destination privilégiée pour les AWS applications de la plateforme Windows, car il propose un environnement géré qui simplifie le déploiement, le dimensionnement et la gestion. La eb migrate commande fournit un chemin direct vers les applications Elastic Beanstalk pour IIS, avec une découverte et une configuration automatiques qui préservent les paramètres de votre application.

Bien qu'elle AWS Application Migration Service offre de puissantes fonctionnalités pour les migrations de serveurs à grande échelle, elle nécessite des étapes supplémentaires pour s'intégrer à Elastic Beanstalk. Pour la plupart des migrations d'applications IIS pour lesquelles Elastic Beanstalk est la eb migrate plate-forme cible, cette solution propose une approche plus rationalisée qui s'aligne sur le modèle de service géré d'Elastic Beanstalk.

Choisissez l'approche de migration la mieux adaptée à vos besoins spécifiques, en tenant compte de facteurs tels que l'échelle, la complexité et l'architecture finale que vous souhaitez utiliser. AWS

Pour plus d'informations AWS Application Migration Service, voir Qu'est-ce que c'est AWS Application Migration Service ? dans le guide de AWS Application Migration Service l'utilisateur.

Résolution des problèmes liés à votre environnement Elastic Beanstalk

Ce chapitre fournit des conseils pour résoudre les problèmes liés à votre environnement Elastic Beanstalk. Elle comprend les éléments suivants :

- Présentation de AWS Systems Manager, y compris une procédure permettant d'exécuter des runbooks Elastic Beanstalk prédéfinis qui fournissent des étapes de dépannage et des recommandations.
- Conseils généraux concernant les actions et les ressources à utiliser lorsque l'état de votre environnement se dégrade.
- Conseils de dépannage spécifiques organisés par catégorie.

Si l'état de votre environnement passe au rouge, nous vous recommandons d'utiliser d'abord l' AWS Systems Manager outil qui inclut des runbooks prédéfinis pour résoudre les problèmes liés à Elastic Beanstalk. Pour en savoir plus, consultez le Utilisation de l'outil Systems Manager.

> Essayez Amazon Q Developer CLI pour un dépannage assisté par l'IA
>
> La CLI Amazon Q Developer peut vous aider à résoudre rapidement les problèmes d'environnement. La Q CLI fournit des solutions en vérifiant l'état de l'environnement, en examinant les événements, en analysant les journaux et en posant des questions de clarification. Pour plus d'informations et des instructions détaillées, consultez la section Résolution des problèmes liés aux environnements Elastic Beanstalk avec Amazon Q Developer CLI dans les blogs. AWS

Rubriques

- Utilisation de l'analyse de l'environnement basée sur l'IA
- Utilisation des AWS Systems Manager runbooks Elastic Beanstalk
- Conseils généraux pour résoudre les problèmes liés à votre environnement Elastic Beanstalk
- Environnements qui accèdent aux secrets et aux paramètres à l'aide de variables d'environnement
- Création de l'environnement et lancements d'instance

- Déploiements
- Santé
- Environnements dotés d'équilibreurs de charge configurés à double pile
- Configuration
- Résolution des problèmes de conteneurs Docker
- FAQ
- Erreurs courantes
- Erreurs de déploiement

Utilisation de l'analyse de l'environnement basée sur l'IA

Elastic Beanstalk fournit des analyses basées sur l'IA pour aider à identifier les causes profondes et à recommander des solutions aux problèmes de santé liés à l'environnement. Vous pouvez accéder à cette fonctionnalité via le bouton AI Analysis de la console, en utilisant l'API Elastic Beanstalk AWS CLI via ou en eb logs --analyze exécutant l'EB CLI. L'analyse examine les journaux, les événements et l'état de santé des instances de votre environnement afin d'identifier les problèmes critiques et de fournir des recommandations de step-by-step résolution des problèmes.

Pour de plus amples informations, veuillez consulter Analyse de l'environnement basée sur l'IA.

Utilisation des AWS Systems Manager runbooks Elastic Beanstalk

Vous pouvez utiliser Systems Manager pour résoudre les problèmes de vos environnements Elastic Beanstalk. Pour vous aider à démarrer rapidement, Systems Manager fournit des runbooks d'automatisation prédéfinis pour Elastic Beanstalk. Un runbook d'automatisation est un type de document Systems Manager qui définit les actions à effectuer sur les instances de votre environnement et d'autres ressources AWS .

Le document `AWSSupport-TroubleshootElasticBeanstalk` est un runbook d'automatisation conçu pour aider à identifier un certain nombre de problèmes courants susceptibles de dégrader votre environnement Elastic Beanstalk. Pour ce faire, il vérifie les composants de votre environnement, notamment les suivants : les instances EC2, le VPC, la pile, les CloudFormation équilibreurs de charge, les groupes Auto Scaling et la configuration réseau associée aux règles des groupes de sécurité, aux tables de routage et. ACLs

Il fournit également la possibilité de charger des fichiers journaux groupés depuis votre environnement vers AWS Support.

Pour plus d'informations, consultez `AWSSupport-TroubleshootElasticBeanstalk` dans la référence AWS Systems Manager Automation runbook.

Utilisation de Systems Manager pour exécuter le runbook **`AWSSupport-TroubleshootElasticBeanstalk`**

> **Note**
>
> Exécutez cette procédure Région AWS là où se trouve votre environnement Elastic Beanstalk.

1. Ouvrez la console AWS Systems Manager.
2. Dans le panneau de navigation, dans la section Gestion des modifications, choisissez Automatisation.
3. Choisissez Execute automation (Exécuter l'automatisation).
4. Dans l'onglet Appartenant à Amazon, dans le champ de recherche de Document d'automatisation, saisissez `AWSSupport-TroubleshootElasticBeanstalk`.
5. Sélectionnez la AWSSupport-TroubleshootElasticBeanstalkcarte, puis choisissez Next.
6. Sélectionnez Exécuter.
7. Dans la section Paramètres d'entrée :
   a. Dans le menu AutomationAssumeRoledéroulant, sélectionnez l'ARN du rôle qui permet à Systems Manager d'effectuer des actions en votre nom.
   b. Pour ApplicationName, entrez le nom de l'application Elastic Beanstalk.
   c. Dans Nom de l'environnement, saisissez l'environnement Elastic Beanstalk.
   d. (Facultatif) Pour S3 UploaderLink, entrez un lien si un ingénieur AWS Support vous a fourni un lien S3 pour la collecte des journaux.
8. Sélectionnez Execute (Exécuter).

   Si l'une des étapes échoue, sélectionnez le lien situé sous la colonne ID d'étape correspondant à l'étape qui a échoué. Cela affiche la page Détails de l'exécution pour l'étape. La VerificationErrorMessagesection affichera un résumé des étapes qui nécessitent une attention

particulière. Par exemple, `IAMPermissionCheck` peut afficher un message d'avertissement. Dans ce cas, vous pouvez vérifier que le rôle sélectionné dans le AutomationAssumeRolemenu déroulant dispose des autorisations nécessaires.

Une fois que toutes les étapes ont été effectuées avec succès, le résultat fournit des étapes de dépannage et des recommandations pour rétablir l'état de votre environnement.

Conseils généraux pour résoudre les problèmes liés à votre environnement Elastic Beanstalk

Des messages d'erreur peuvent s'afficher sur la page Événements de la console, dans les journaux ou sur la page État. Vous pouvez également prendre des mesures pour récupérer un environnement dégradé causé par une modification récente. Si l'intégrité de votre environnement passe au Rouge, essayez les éléments suivants :

- Demandez une analyse de votre environnement basée sur l'IA. Pour de plus amples informations, veuillez consulter Analyse de l'environnement basée sur l'IA.
- Si une opération sur votre environnement renvoie une erreur contenant le texte`The stack stack_id associated with environment environment-ID is in stack-status state`, consultez l'aide à la résolution Restaurer votre environnement Elastic Beanstalk à partir d'un état non valide des problèmes.
- Si une opération sur votre environnement renvoie une erreur contenant le texte`Environment environment-name associated CloudFormation stack stack_arn does not exist`, fermez votre environnement et créez-en un autre.
- Consultez les événements récents de l'environnement. Les messages d'Elastic Beanstalk sur les problèmes de déploiement, de charge et de configuration s'affichent souvent ici.
- Examinez l'historique des modifications d'environnement récents. L'historique des modifications répertorie toutes les modifications apportées à la configuration de vos environnements et inclut d'autres informations, telles que l'utilisateur IAM ayant apporté les modifications et les paramètres de configuration définis.
- Extrayez des journaux afin d'afficher de récentes entrées du fichier journal. Les journaux de serveur web contiennent des informations sur les erreurs et les demandes entrantes.
- Connectez-vous à une instance et vérifiez les ressources du système.
- Restaurez sur une version de travail précédente de l'application.

- Annulez les récentes modifications de configuration ou restaurez une configuration sauvegardée.
- Déployez un nouvel environnement. Si l'environnement s'avère sain, effectuez un échange CNAME pour acheminer le trafic vers le nouvel environnement et continuer à déboguer l'ancien.

Environnements qui accèdent aux secrets et aux paramètres à l'aide de variables d'environnement

Événement : le déploiement de l'instance n'a pas réussi à obtenir un ou plusieurs secrets

Ce message indique qu'Elastic Beanstalk n'a pas pu récupérer un ou plusieurs des secrets spécifiés lors du déploiement de votre application.

- Vérifiez que les ressources spécifiées par les valeurs ARN dans la configuration de vos variables d'environnement existent.
- Vérifiez que votre rôle de profil d'instance Elastic Beanstalk EC2 dispose des autorisations IAM requises pour accéder aux ressources.
- Si cet événement a été déclenché par l'`RestartAppServer`opération, une fois le problème résolu, relancez l'`RestartAppServer`appel pour le résoudre.
- Si l'événement a été déclenché par un `UpdateEnvironment` appel, recommencez l'`UpdateEnvironment`opération.

Pour des exemples de ces commandes, consultez les AWS CLI exemples d'Elastic Beanstalk. Pour plus d'informations sur les actions d'API associées à ces opérations, consultez la référence des AWS Elastic Beanstalk API.

Événement : le déploiement de l'instance a détecté une ou plusieurs valeurs d'environnement multilignes, qui ne sont pas prises en charge pour cette plate-forme

Les variables multilignes ne sont pas prises en charge sur les plateformes Amazon Linux 2, à l'exception des plateformes Docker et Docker gérées par ECS. Pour connaître les options disponibles pour continuer, consultezValeurs multilignes.

Événement : CreateEnvironment échoue lorsqu'un secret est spécifié

En cas d'`CreateEnvironment`échec et que vous avez des secrets en tant que variables d'environnement, vous devez résoudre le problème sous-jacent, puis l'utiliser `UpdateEnvironment` pour terminer la configuration de l'environnement. Ne pas utiliser`RestartAppServer`, car cela ne

suffira pas à améliorer l'environnement dans cette situation. Pour des exemples de ces commandes, consultez les AWS CLI exemples d'Elastic Beanstalk. Pour plus d'informations sur les actions d'API associées à ces opérations, consultez la référence des AWS Elastic Beanstalk API.

Création de l'environnement et lancements d'instance

Événement : Échec du lancement de l'environnement

Cet événement se produit lorsque Elastic Beanstalk tente de lancer un environnement et rencontre des défaillances ce faisant. Les événements précédents sur la page Événements vous alertent de la cause racine de ce problème.

Événement : L'opération de création de l'environnement est terminée, mais avec des délais de commande. Essayez d'augmenter le délai d'expiration.

Le déploiement de votre application peut prendre beaucoup de temps si vous utilisez des fichiers de configuration qui exécutent des commandes sur l'instance, téléchargent des fichiers volumineux ou installent des packages. Augmentez le délai de commande afin que votre application dispose de plus de temps pour commencer son exécution pendant les déploiements.

Événement : La ou les ressources suivantes n'ont pas pu être créées : [AWSEBInstanceLaunchWaitCondition]

Ce message indique que les instances Amazon EC2 de votre environnement n'ont pas communiqué à Elastic Beanstalk qu'elles ont été lancées avec succès. Cette erreur peut se produire si ces instances n'ont pas de connectivité Internet. Si vous avez configuré votre environnement pour lancer des instances sur un sous-réseau VPC privé, vérifiez que ce sous-réseau possède un NAT pour permettre aux instances de se connecter à Elastic Beanstalk.

Événement : Une fonction du service est requise dans cette région. Veuillez ajouter une option Fonction du service à l'environnement.

Elastic Beanstalk utilise un rôle de service pour surveiller les ressources de votre environnement et prendre en charge les mises à jour de la plateforme gérée. Pour plus d'informations, consultez Gestion des rôles de service Elastic Beanstalk.

Déploiements

Problème : L'application devient indisponible pendant les déploiements

Un temps d'arrêt de quelques secondes est possible, car Elastic Beanstalk utilise un processus de mise à niveau par remplacement direct. Utilisez des propagations de déploiements pour minimiser l'impact des déploiements sur vos environnements de production.

Événement : Impossible de créer la version de l'application AWS Elastic Beanstalk

L'ensemble de sources de votre application est peut-être trop volumineux ou vous avez peut-être atteint le quota des versions d'application.

Événement : L'opération de mise à jour de l'environnement est terminée, mais avec des délais de commande. Essayez d'augmenter le délai d'expiration.

Le déploiement de votre application peut prendre beaucoup de temps si vous utilisez des fichiers de configuration qui exécutent des commandes sur l'instance, téléchargent des fichiers volumineux ou installent des packages. Augmentez le délai de commande afin que votre application dispose de plus de temps pour commencer son exécution pendant les déploiements.

Santé

Événement : L'utilisation de l'UC dépasse 95 %

Essayez d'exécuter davantage d'instances ou choisissez un type d'instance différent.

Événement : Elastic Load Balancer awseb-*myapp* Il n'y a aucune instance saine, les processus ELB ne sont pas sains en cas de panne d'instances. *X Y* , les processus ELB ne sont pas sains sur toutes les instances.

Ces messages indiquent que les vérifications de l'état de votre équilibreur de charge échouent sur une ou plusieurs instances. Si votre application semble fonctionner, assurez-vous que l'URL de vérification de l'état de votre application est correctement configurée. Si ce n'est pas le cas, essayez ce qui suit :

- Vérifiez les journaux de vos applications (par exemple,`/var/log/web.stdout.log`) et vérifiez s'il `/var/log/eb-engine.log` y a des erreurs, des échecs de démarrage ou des problèmes de déploiement.
- Vérifiez les journaux du proxy pour détecter les erreurs de connexion dans `/var/log/nginx/` (nginx) `/var/log/httpd/` ou (Apache).
- Vérifiez que votre application répond par HTTP 200 au chemin de vérification de santé configuré et qu'elle écoute sur le port attendu.

- Vérifiez que les groupes de sécurité autorisent le trafic entrant depuis l'équilibreur de charge vers les instances situées sur le port de contrôle de santé. Dans la console EC2, vérifiez les règles entrantes du groupe de sécurité de l'instance pour confirmer que le port de vérification de l'état est ouvert au groupe de sécurité de l'équilibreur de charge.
- Vérifiez l'utilisation du processeur et de la mémoire dans CloudWatch les métriques (`CPUUtilization`,`EnvironmentHealth`). L'épuisement des ressources peut entraîner des délais d'attente pour les bilans de santé.

Pour plus de détails, récupérez les journaux complets du bundle ou activez le streaming des CloudWatch journaux. Vous pouvez également exécuter le manuel `AWSSupport-TroubleshootElasticBeanstalk`d'automatisation pour aider à diagnostiquer le problème.

Événement : Elastic Load Balancer awseb- introuvable *`myapp`*

L'équilibreur de charge de votre environnement a peut-être été supprimé. out-of-band Ne modifiez les ressources de votre environnement qu'avec les options de configuration et l'extensibilité fournies par Elastic Beanstalk. Recréez votre environnement ou lancez-en un nouveau.

Événement : Échec du lancement de l'instance EC2. Lancement d'une nouvelle instance EC2 en attente…

La disponibilité du type d'instance de votre environnement peut être faible ou vous avez peut-être atteint le quota d'instances de votre compte. Vérifiez le tableau de bord de l'état des services pour vous assurer que le service Elastic Compute Cloud (Amazon EC2) est vert ou demandez une augmentation de quota.

Événement : aucune des instances n'envoie de données.

Elastic Beanstalk ne reçoit de données sur l'état de santé d'aucune de vos instances. Cela peut être dû à des problèmes de connectivité réseau, à des problèmes d'agent de santé ou à des défaillances au niveau du système.

- Vérifiez la connectivité réseau. Vérifiez que les groupes de sécurité autorisent le protocole HTTPS sortant (port 443). Pour les instances situées dans des sous-réseaux privés, vérifiez qu'une passerelle NAT ou des points de terminaison VPC sont configurés. Vérifiez que les routes sortantes sont correctes dans les tables de routage et assurez-vous que la résolution DNS fonctionne correctement dans votre VPC.
- Vérifiez l'agent de surveillance de l'état de santé. Recherchez `/var/log/healthd/daemon.log` les erreurs liées aux agents de santé et `/var/log/messages` les erreurs au niveau du système.

Vérifiez que le service healthd est en cours d'exécution en vous connectant à l'instance via SSH et en l'exécutant. `systemctl status healthd`

- Vérifiez les problèmes au niveau du système tels que l'épuisement des ressources (mémoire, processeur, espace disque), la synchronisation des NTP/system horloges et la connectivité des services de métadonnées d'instance.

Pour plus de détails, récupérez les journaux complets du bundle ou activez le streaming des CloudWatch journaux. Vous pouvez également exécuter le manuel `AWSSupport-TroubleshootElasticBeanstalk`d'automatisation pour aider à diagnostiquer le problème.

Environnements dotés d'équilibreurs de charge configurés à double pile

Événement : aucun bloc IPv6 CIDR *`vpc_id`* n'est configuré pour le VPC. IPv6 Les blocs CIDR sont nécessaires pour l'équilibreur de charge à double pile. Veuillez associer un bloc IPv6 CIDR à votre VPC avant d'utiliser le mode dualstack.

Des blocs IPv6 CIDR doivent être associés à votre VPC et à tous les sous-réseaux. Il s'agit de l'une des conditions préalables au VPC que vous devez remplir avant de configurer votre équilibreur de charge pour la prise en charge du double stack. Pour plus d'informations sur l'exécution de cette tâche, voir Conditions préalables à Amazon VPC plus haut dans cette rubrique.

Événement : aucun bloc CIDR n' IPv6 est configuré pour un ou plusieurs sous-réseaux *`vpc_id`* VPC. IPv6 Les blocs CIDR sont requis pour les sous-réseaux utilisés avec l'équilibreur de charge à double pile. Associez les blocs IPv6 CIDR à tous les sous-réseaux requis avant d'utiliser le mode dualstack.

Des blocs CIDR doivent être IPv6 associés à tous les sous-réseaux de votre VPC. Il s'agit de l'une des conditions préalables au VPC que vous devez remplir avant de configurer votre équilibreur de charge pour la prise en charge du double stack. Pour plus d'informations sur l'exécution de cette tâche, voir Conditions préalables à Amazon VPC plus haut dans cette rubrique.

Erreur : l'*`IpAddressType`*option ne peut être appliquée que sur les environnements Elastic Beanstalk configurés avec un Application Load Balancer ou un Network Load Balancer.

Ce message indique que votre environnement Elastic Beanstalk est peut-être un environnement à instance unique ou qu'il utilise un Classic Load Balancer. Seuls les environnements configurés avec un Application Load Balancer ou un Network Load Balancer peuvent être configurés. `IpAddressType`

Configuration

Événement : La pile *`stack_id`* associée à l'environnement *`environment-ID`* est en *`stack-status`* état

La CloudFormation pile sous-jacente de votre environnement peut avoir le statut *_FAILED. Ce statut doit être corrigé afin de poursuivre les opérations d'Elastic Beanstalk dans votre environnement. Pour de plus amples informations, veuillez consulter Restaurer votre environnement Elastic Beanstalk à partir d'un état non valide.

Événement : Vous ne pouvez pas configurer un environnement Elastic Beanstalk avec des valeurs pour les options Elastic Load Balancing Target (Cible Elastic Load Balancing) et Application Healthcheck URL (URL de vérification de l'état de l'application)

L'option `Target` dans l'espace de noms `aws:elb:healthcheck` est obsolète. Supprimez l'option `Target` de votre environnement et essayez à nouveau de mettre à jour.

Événement : l'ELB ne peut pas être attaché à plusieurs sous-réseaux dans la même zone de disponibilité

Ce message s'affiche si vous tentez de déplacer un équilibreur de charge entre des sous-réseaux d'une même zone de disponibilité. La modification de sous-réseaux sur l'équilibreur de charge nécessite de le faire sortir en dehors des zones de disponibilité, puis de le réintégrer dans la zone de disponibilité d'origine avec les sous-réseaux souhaités. Au cours du processus, toutes vos instances seront migrées entre elles AZs, ce qui entraînera des temps d'arrêt importants. Pensez plutôt à créer un nouvel environnement et effectuez un échange CNAME.

Résolution des problèmes de conteneurs Docker

Événement : Échec de l'extraction de l'image Docker : la plus récente : Nom de référentiel () non valide, seulement [a-z0-9-_.] sont autorisés. Consultez les journaux pour plus d'informations.

Vérifiez la syntaxe du fichier `dockerrun.aws.json` à l'aide d'un validateur JSON. Vérifiez également le contenu dockerfile par rapport aux conditions décrites dans Préparation de votre image Docker pour le déploiement sur Elastic Beanstalk

Événement : Aucune directive EXPOSE trouvée dans le Dockerfile, abandon du déploiement

Le `Dockerfile` ou le fichier `dockerrun.aws.json` ne déclare pas le port de conteneur. Utilisez l'instruction `EXPOSE` (`Dockerfile`) ou le bloc `Ports` (fichier `dockerrun.aws.json`) pour exposer un port pour le trafic entrant.

Événement : Impossible de télécharger les informations d'authentification *`repository`* depuis *`bucket name`*

`dockerrun.aws.json`Fournit un compartiment and/or S3 à paire de clés EC2 non valide pour le `.dockercfg` fichier. Ou bien, le profil d'instance n'est pas GetObject autorisé pour le compartiment S3. Vérifiez que le fichier `.dockercfg` contient un compartiment S3 et une paire de clés EC2 valides. Accordez des autorisations pour l'action `s3:GetObject` au rôle IAM dans le profil d'instance. Pour obtenir des détails, consultez Gestion des profils d'instance Elastic Beanstalk

Événement : L'exécution de l'activité a échoué, car : AVERTISSEMENT : Fichier de configuration d'authentification non valide

Le format de votre fichier d'authentification (`config.json`) n'est pas correct. Consultez Authentification avec des référentiels d'images

FAQ

Question : Comment remplacer l'URL de mon application myapp.us-west-2.elasticbeanstalk.com par www.myapp.com ?

Dans un serveur DNS, enregistrez un enregistrement CNAME tel que **`www.mydomain.com CNAME mydomain.elasticbeanstalk.com`**.

Question : Comment spécifier une zone de disponibilité spécifique pour mon application Elastic Beanstalk ?

Vous pouvez sélectionner une zone de disponibilité spécifique à l'aide de la CLI APIs, du plugin Eclipse ou du plugin Visual Studio. Pour de plus amples informations sur l'utilisation de la console Elastic Beanstalk pour spécifier une zone de disponibilité, veuillez consulter Auto Scaling de vos instances d'environnement Elastic Beanstalk.

Question : Comment modifier le type d'instance de mon environnement ?

Pour modifier le type d'instance de votre environnement, accédez à la page de configuration de l'environnement et sélectionnez Edit (Modifier) dans la catégorie de configuration Instances. Ensuite, sélectionnez un nouveau type d'instance, puis choisissez Apply (Appliquer) pour mettre à jour votre

environnement. Après cela, Elastic Beanstalk résilie toutes les instances en cours d'exécution et les remplace par de nouvelles.

Question : Comment déterminer si quelqu'un a apporté des modifications de configuration à un environnement ?

Pour afficher ces informations, dans le volet de navigation de la console Elastic Beanstalk, sélectionnez Change history (Historique des modifications) pour afficher la liste des modifications de configuration de tous les environnements. Cette liste inclut la date et l'heure de la modification, le paramètre de configuration modifié et sa nouvelle valeur, ainsi que l'utilisateur IAM qui a effectué la modification. Pour plus d'informations, consultez Historique des modifications.

Question : Puis-je éviter que les volumes Amazon EBS soient supprimés lorsque les instances sont mises hors service ?

Les instances de votre environnement utilisent Amazon EBS pour le stockage. Toutefois, le volume racine est supprimé lorsqu'une instance est résiliée par Auto Scaling. Nous vous recommandons de ne pas stocker les données d'état ou d'autres informations dans vos instances. Si nécessaire, vous pouvez empêcher la suppression de volumes à l'aide du AWS CLI : `$ aws ec2 modify-instance-attribute -b '/dev/sdc=<vol-id>:false` comme décrit dans la AWS CLI référence.

Question : Comment supprimer des informations personnelles de mon application Elastic Beanstalk ?

AWS les ressources utilisées par votre application Elastic Beanstalk peuvent stocker des informations personnelles. Lorsque vous arrêtez un environnement, Elastic Beanstalk arrête les ressources qu'il a créées. Les ressources que vous avez ajoutées à l'aide de fichiers de configuration sont également résiliées. Toutefois, si vous avez créé AWS des ressources en dehors de votre environnement Elastic Beanstalk et que vous les avez associées à votre application, vous devrez peut-être vérifier manuellement que les informations personnelles que votre application a pu stocker ne sont pas conservées. Tout au long de ce manuel du développeur, chaque fois que nous abordons la création de ressources supplémentaires, nous mentionnons également le moment auquel vous devez envisager de les supprimer.

Erreurs courantes

Cette rubrique répertorie les messages d'erreur courants rencontrés lors de l'utilisation de l'interface de ligne de commande EB et les solutions possibles. Si vous rencontrez un message d'erreur qui n'est pas présenté ici, utilisez les liens Feedback pour nous en faire part.

ERREUR : une erreur s'est produite lors de la gestion de la commande git. Code d'erreur : 128 Erreur : fatale : nom d'objet non valide HEAD

Cause : Ce message d'erreur s'affiche lorsque vous avez initialisé un dépôt Git mais que vous ne l'avez pas encore validé. L'interface de ligne de commande EB cherche la révision HEAD lorsque votre dossier de projet contient un référentiel Git.

Solution : ajoutez les fichiers dans votre dossier de projet à la zone intermédiaire et valides :

```
~/my-app$ git add .
~/my-app$ git commit -m "First commit"
```

ERREUR : Cette branche ne possède pas d'environnement par défaut. Vous devez soit spécifier un environnement en saisissant « eb status my-env-name », soit définir un environnement par défaut en tapant « eb use my-env-name ».

Cause : lorsque vous créez une branche dans git, elle n'est pas attachée à un environnement Elastic Beanstalk par défaut.

Solution : exécutez eb list pour afficher une liste des environnements disponibles. Exécutez ensuite eb use ***env-name*** pour utiliser l'un des environnements disponibles.

ERREUR : Les plateformes 2.0+ nécessitent un rôle de service. Vous pouvez en fournir un avec l'option --service-role

Cause : si vous spécifiez un nom d'environnement avec eb create (par exemple, eb create my-env), l'interface de ligne de commande EB n'essaie pas de créer un rôle de service pour vous. Si vous n'avez pas le rôle de service par défaut, l'erreur ci-dessus s'affiche.

Solution : exécutez eb create sans un nom d'environnement et suivez les instructions pour créer le rôle du service par défaut.

Erreurs de déploiement

Votre déploiement d'Elastic Beanstalk peut répondre par une réponse 404 (si votre application ne démarre pas) ou 500 (si votre application échoue pendant l'exécution). Pour résoudre de nombreux problèmes courants, vous pouvez utiliser l'interface de ligne de commande EB pour vérifier l'état de votre déploiement, consulter ses journaux, accéder à votre instance EC2 via SSH ou ouvrir la page de console de AWS gestion de votre environnement d'application.

Pour utiliser l'interface de ligne de commande EB pour aider à dépanner votre déploiement

1. Exécutez eb status pour afficher l'état de votre déploiement actuel et l'état de vos hôtes EC2. Par exemple :

```
$ eb status --verbose

Environment details for: python_eb_app
  Application name: python_eb_app
  Region: us-west-2
  Deployed Version: app-150206_035343
  Environment ID: e-wa8u6rrmqy
  Platform: 64bit Amazon Linux 2014.09 v1.1.0 running Python 2.7
  Tier: WebServer-Standard-
  CNAME: python_eb_app.elasticbeanstalk.com
  Updated: 2015-02-06 12:00:08.557000+00:00
  Status: Ready
  Health: Green
  Running instances: 1
      i-8000528c: InService
```

> **Note**
>
> L'utilisation du commutateur `--verbose` fournit des informations sur l'état de vos instances en cours d'exécution. Sans cela, eb status imprimera uniquement des informations générales sur votre environnement.

2. Exécutez eb health afin d'afficher des informations d'intégrité sur votre environnement :

```
$ eb health --refresh
 elasticBeanstalkExa-env                                  Degraded
 2016-03-28 23:13:20
WebServer
    Ruby 2.1 (Puma)
  total      ok    warning  degraded  severe    info   pending  unknown
    5        2        0        2        1        0        0        0

  instance-id   status     cause
    Overall     Degraded  Incorrect application version found on 3 out of 5
 instances. Expected version "Sample Application" (deployment 1).
```

```
 i-d581497d    Degraded  Incorrect application version "v2" (deployment 2).
Expected version "Sample Application" (deployment 1).
 i-d481497c    Degraded  Incorrect application version "v2" (deployment 2).
Expected version "Sample Application" (deployment 1).
 i-136e00c0    Severe    Instance ELB health has not been available for 5 minutes.
 i-126e00c1    Ok
 i-8b2cf575    Ok

 instance-id   r/sec    %2xx   %3xx   %4xx   %5xx      p99      p90      p75
p50     p10
   Overall     646.7   100.0    0.0    0.0    0.0    0.003    0.002    0.001
0.001   0.000
 i-dac3f859    167.5    1675      0      0      0    0.003    0.002    0.001
0.001   0.000
 i-05013a81    161.2    1612      0      0      0    0.003    0.002    0.001
0.001   0.000
 i-04013a80    0.0         -      -      -      -        -        -       -
-       -
 i-3ab524a1    155.9    1559      0      0      0    0.003    0.002    0.001
0.001   0.000
 i-bf300d3c    162.1    1621      0      0      0    0.003    0.002    0.001
0.001   0.000

 instance-id   type       az   running     load 1  load 5      user%  nice%
system%  idle%   iowait%
 i-d581497d    t2.micro   1a   25 mins       0.16     0.1        7.0    0.0
1.7   91.0       0.1
 i-d481497c    t2.micro   1a   25 mins       0.14     0.1        7.2    0.0
1.6   91.1       0.0
 i-136e00c0    t2.micro   1b   25 mins        0.0    0.01        0.0    0.0
0.0   99.9       0.1
 i-126e00c1    t2.micro   1b   25 mins       0.03    0.08        6.9    0.0
2.1   90.7       0.1
 i-8b2cf575    t2.micro   1c   1 hour        0.05    0.41        6.9    0.0
2.0   90.9       0.0

 instance-id   status     id   version             ago
     deployments
 i-d581497d    Deployed   2    v2                  9 mins
 i-d481497c    Deployed   2    v2                  7 mins
 i-136e00c0    Failed     2    v2                  5 mins
 i-126e00c1    Deployed   1    Sample Application  25 mins
 i-8b2cf575    Deployed   1    Sample Application  1 hour
```

L'exemple ci-dessus illustre un environnement avec cinq cas où le déploiement de la version « v2 » a échoué sur la troisième instance. Après un échec de déploiement, la version attendue revient sur la dernière version qui a abouti, ce qui dans ce cas correspond à « Exemple d'Application » depuis le premier déploiement. Pour plus d'informations, consultez Utilisation de l'interface de ligne de commande EB pour surveiller l'intégrité de l'environnement.

3. Exécutez eb logs pour télécharger et afficher les journaux associés au déploiement de votre application.

```
$ eb logs
```

4. Exécutez eb ssh pour vous connecter à l'instance EC2 qui exécute votre application et examinez-la directement. Sur l'instance, votre application déployée est disponible dans le répertoire `/opt/python/current/app` et votre environnement Python se trouve dans `/opt/python/run/venv/`.
5. Exécutez eb console pour afficher l'environnement de votre application sur la console de gestion AWS. Vous pouvez utiliser l'interface web afin d'examiner facilement les différents aspects de votre déploiement, y compris les journaux, les événements, l'état et la configuration de votre application. Vous pouvez également télécharger les versions d'application actuelles ou passées que vous avez déployées sur le serveur.

Ressources Elastic Beanstalk

Les ressources connexes suivantes peuvent s'avérer utiles lors de l'utilisation de ce service.

- Référence de l'API Elastic Beanstalk Description complète de tous les protocoles SOAP et Query. APIs Cette référence contient également une liste de tous les types de données SOAP.
- elastic-beanstalk-samples on GitHub — Un GitHub référentiel contenant des exemples de fichiers de configuration d'Elastic Beanstalk (.ebextensions). Le `README.md` fichier du référentiel contient des liens vers des GitHub référentiels supplémentaires contenant des exemples d'applications.
- FAQ technique Elastic Beanstalk – Les principales questions posées par les développeurs à propos de ce produit.
- AWS Elastic Beanstalk Notes de mise à jour : détails sur les nouvelles fonctionnalités, les mises à jour et les correctifs des versions du service, de la plateforme, de la console et de l'EB CLI d'Elastic Beanstalk.

- Cours et ateliers — Liens vers des cours spécialisés et basés sur des rôles, ainsi que des ateliers à votre rythme pour vous aider à perfectionner vos AWS compétences et à acquérir une expérience pratique.
- AWS Centre pour développeurs : découvrez les didacticiels, téléchargez des outils et découvrez les événements AWS destinés aux développeurs.
- AWS Outils de développement : liens vers des outils de développement SDKs, des boîtes à outils IDE et des outils de ligne de commande pour le développement et la gestion d' AWS applications.
- Centre de ressources pour la mise en route : découvrez comment configurer votre application Compte AWS, rejoindre la AWS communauté et lancer votre première application.
- Tutoriels pratiques — Suivez les step-by-step didacticiels pour lancer votre première application sur AWS.
- AWS Livres blancs : liens vers une liste complète de livres AWS blancs techniques, traitant de sujets tels que l'architecture, la sécurité et l'économie, rédigés par des architectes de AWS solutions ou d'autres experts techniques.
- AWS Support Centre — Le centre de création et de gestion de vos AWS Support dossiers. Comprend également des liens vers d'autres ressources utiles, telles que des forums, des informations techniques FAQs, l'état de santé des services et AWS Trusted Advisor.
- Support— La principale page Web contenant des informations sur Support un one-on-one canal d'assistance à réponse rapide pour vous aider à créer et à exécuter des applications dans le cloud.

- Contactez-nous : point de contact central pour toute question relative à la facturation AWS , à votre compte, aux événements, à des abus ou à d'autres problèmes.
- AWS Conditions du site — Informations détaillées sur nos droits d'auteur et notre marque commerciale ; votre compte, votre licence et l'accès au site ; et d'autres sujets.

Exemples d’applications

Vous trouverez ci-dessous des liens permettant de télécharger les exemples d'application déployés dans le cadre de Découvrez comment démarrer avec Elastic Beanstalk.

> **Note**
>
> Certains exemples utilisent des fonctions qui ont peut-être été publiées après le lancement de la plateforme que vous utilisez. Si l'exécution de l'exemple échoue, tentez de mettre à jour votre plateforme vers une version actuelle, comme décrit dans the section called “Plateformes prises en charge”.

- Docker – docker.zip
- Docker multiconteneur — 2.zip docker-multicontainer-v
- Docker préconfiguré (Glassfish) — 1.zip docker-glassfish-v
- Go – go.zip
- Corretto – corretto.zip
- Tomcat – tomcat.zip
- .NET Core sous Linux — dotnet-core-linux.zip
- .NET Core — dotnet-asp-windows.zip
- Node.js – nodejs.zip
- PHP – php.zip
- Python – python.zip
- Ruby – ruby.zip

AWS SDK pour Java

AWS SDK pour Java fournit une API Java que vous pouvez utiliser pour créer des applications utilisant des services AWS d'infrastructure. Avec le AWS SDK pour Java, vous pouvez démarrer en quelques minutes avec un seul package téléchargeable qui inclut la bibliothèque AWS Java, des exemples de code et de la documentation.

AWS SDK pour Java Nécessite le kit de développement J2SE. Vous pouvez télécharger la dernière version du logiciel Java sur la page http://developers.sun.com/downloads/. Le SDK nécessite également des packages tiers Apache Commons (Codec et Logging) et Saxon-HE, qui sont inclus dans le répertoire tiers du SDK. HTTPClient

Pour plus d'informations, consultez Kit SDK AWS pour Java. Pour plus d'informations sur le SDK, des exemples de code, de la documentation, des outils et des ressources supplémentaires, consultez le Java on AWS Developer Center.

AWS SDK pour .NET

Avec le AWS SDK pour .NET, vous pouvez démarrer rapidement avec un seul package téléchargeable qui inclut la bibliothèque AWS .NET, des exemples de code et de la documentation.

Pour plus d'informations, consultez Kit SDK AWS pour .NET. Pour connaître les versions de .NET Framework et de Visual Studio prises en charge, consultez le Guide du développeur AWS SDK pour .NET.

Pour plus d'informations sur le SDK, des exemples de code, de la documentation, des outils et des ressources supplémentaires, consultez le .NET on AWS Developer Center.

AWS Toolkit for Visual Studio

Le AWS Toolkit for Visual Studio plug-in vous permet de déployer une application .NET existante sur Elastic Beanstalk. Vous pouvez également créer des projets à l'aide AWS des modèles préconfigurés avec le AWS SDK pour .NET.

Pour plus d'informations sur les prérequis et l'installation, consultez AWS Toolkit for Visual Studio. Pour commencer à créer votre application Elastic Beanstalk à l'aide de Visual Studio, veuillez consulter Déploiement d'applications .NET Windows avec Elastic Beanstalk.

AWS SDK pour JavaScript Node.js

Avec le AWS SDK pour JavaScript Node.js, vous pouvez démarrer rapidement avec un seul package téléchargeable qui inclut la bibliothèque AWS Node.js, des exemples de code et de la documentation.

Pour plus d'informations, consultez le AWS SDK du fichier Node.js. JavaScript

AWS SDK pour PHP

Avec le AWS SDK pour PHP, vous pouvez démarrer rapidement avec un seul package téléchargeable qui inclut la bibliothèque AWS PHP, des exemples de code et de la documentation.

Pour plus d'informations, consultez Kit SDK AWS pour PHP. Pour plus d'informations sur le SDK, des exemples de code, de la documentation, des outils et des ressources supplémentaires, consultez le PHP on AWS Developer Center.

AWS SDK pour Python (Boto)

Avec le AWS SDK pour Python (Boto), vous pouvez démarrer rapidement avec un seul package téléchargeable qui inclut la bibliothèque AWS Python, des exemples de code et de la documentation. Vous pouvez également créer des applications Python en APIs simplifiant le codage directement par rapport aux interfaces de service Web.

La all-in-one bibliothèque fournit des outils Python adaptés aux développeurs et APIs masquent de nombreuses tâches de niveau inférieur associées à la programmation pour le AWS cloud, notamment l'authentification, les nouvelles tentatives de demande et la gestion des erreurs. Le kit SDK fournit des exemples pratiques en Python sur la façon d'utiliser les bibliothèques pour créer des applications.

Pour plus d'informations sur Boto, des exemples de code, de la documentation, des outils et des ressources supplémentaires, consultez le Python on AWS Developer Center.

AWS SDK pour Ruby

Grâce à la bibliothèque AWS Ruby, aux exemples de code et à la documentation, vous pouvez créer des APIs applications Ruby en plus de simplifier le codage directement sur les interfaces de services Web.

La all-in-one bibliothèque fournit Ruby, convivial pour les développeurs, APIs qui masque de nombreuses tâches de niveau inférieur associées à la programmation pour le AWS cloud, notamment

l'authentification, les nouvelles tentatives de demande et la gestion des erreurs. Le kit SDK fournit des exemples pratiques en Ruby sur la façon d'utiliser les bibliothèques pour créer des applications.

Pour plus d'informations sur le SDK, des exemples de code, de la documentation, des outils et des ressources supplémentaires, consultez le Ruby on AWS Developer Center.

Contenu archivé pour le guide du AWS Elastic Beanstalk développeur

À des fins de référence, ce chapitre fournit des informations de configuration relatives à certaines fonctionnalités héritées d'Elastic Beanstalk.

Rubriques

- Plateformes personnalisées Elastic Beanstalk (retirées)

Plateformes personnalisées Elastic Beanstalk (retirées)

> **Note**
>
> Le 18 juillet 2022, Elastic Beanstalk a défini le statut de toutes les branches de la plateforme sur la base de l'AMI AL1 Amazon Linux () comme étant supprimées. Cela inclut les plateformes personnalisées. Elastic Beanstalk ne prend pas en charge les plateformes personnalisées. Pour en savoir plus sur le processus de retrait d'Amazon Linux AMI par Elastic Beanstalk, consultez FAQ sur le retrait de la plateforme.

Cette rubrique reste dans ce document comme référence pour tout client ayant utilisé la fonctionnalité de plateforme personnalisée d'Elastic Beanstalk avant son retrait. Dans le passé, les plateformes personnalisées Elastic Beanstalk permettaient de créer une AMI à partir de l'AMI Amazon Linux, RHEL 7, RHEL 6 ou Ubuntu 16.04 base. AMIs Ces systèmes d'exploitation ne sont plus pris en charge par Elastic Beanstalk. Pour en savoir plus sur la fonctionnalité des plateformes personnalisées, qui n'est plus prise en charge, consultez la rubrique suivante.

Plateformes personnalisées

Une plateforme personnalisée est à plusieurs titres une personnalisation plus avancée qu'une image personnalisée. Une plateforme personnalisée vous permet de développer une plateforme entièrement nouvelle avec un système d'exploitation personnalisé, des logiciels supplémentaires et des scripts exécutés par Elastic Beanstalk sur les instances de la plateforme. Cette souplesse vous permet de construire une plateforme pour une application qui utilise un langage ou d'autres logiciels d'infrastructure pour lesquels Elastic Beanstalk ne fournit pas de plateforme gérée. Cela peut

se comparer aux images personnalisées où une Amazon Machine Image (AMI) est modifiée pour être utilisée avec une plateforme Elastic Beanstalk existante, et Elastic Beanstalk fournit toujours les scripts de plateforme et contrôle la pile logicielle de la plateforme. Par ailleurs, avec les plateformes personnalisées, vous créez et tenez à jour vos personnalisations de façon automatisée et scriptée, alors qu'avec les images personnalisées, vous apportez les modifications manuellement sur une instance en cours d'exécution.

Pour créer une plateforme personnalisée, vous devez générer une AMI à partir de l'un des systèmes d'exploitation pris en charge, Ubuntu, RHEL ou Amazon Linux (consultez l'entrée `flavor` dans Format de fichier platform.yaml pour connaître les numéros de version exacts), puis ajouter d'autres personnalisations. Vous créez votre propre plateforme Elastic Beanstalk à l'aide de Packer, un outil open source permettant de créer des images de machine pour de nombreuses plateformes, AMIs notamment pour Amazon Elastic Compute Cloud (Amazon). EC2 Une plateforme Elastic Beanstalk comprend une AMI configurée pour exécuter un ensemble de logiciels prenant en charge une application, et des métadonnées qui peuvent inclure des options de configuration personnalisées et des paramètres d'option de configuration par défaut.

Elastic Beanstalk gère Packer comme une plateforme intégrée distincte. Autrement dit, vous n'avez pas besoin de vous soucier de la configuration et des versions de Packer.

Vous créez une plateforme en fournissant un modèle Packer à Elastic Beanstalk, ainsi que les scripts et les fichiers que le modèle appelle pour créer une AMI. Ces composants sont fournis avec un fichier de définition de plateforme, qui spécifie le modèle et les métadonnées, dans une archive ZIP appelée archive de définition de plateforme.

Lorsque vous créez une plateforme personnalisée, vous lancez un environnement d'instance unique sans aucune adresse IP élastique pour exécuter Packer. Packer lance alors une autre instance pour générer une image. Vous pouvez réutiliser cet environnement pour plusieurs plateformes et plusieurs versions de chaque plateforme.

> **Note**
>
> Les plateformes personnalisées sont spécifiques à chaque AWS région. Si vous utilisez Elastic Beanstalk dans plusieurs régions, vous devez créer vos plateformes séparément dans chaque région.
>
> Dans certaines circonstances, les instances lancées par Packer ne sont pas nettoyées et doivent être mises hors service manuellement. Pour savoir comment nettoyer manuellement ces instances, consultez Nettoyage des instances Packer.

Les utilisateurs de votre compte peuvent utiliser vos plateformes personnalisées en spécifiant un ARN de plateforme lors de la création de l'environnement. Ils ARNs sont renvoyés par la eb platform create commande que vous avez utilisée pour créer la plateforme personnalisée.

Chaque fois que vous créez votre plateforme personnalisée, Elastic Beanstalk crée une nouvelle version de la plateforme. Les utilisateurs peuvent spécifier une plateforme par son nom pour obtenir uniquement la dernière version de la plateforme, ou inclure un numéro de version pour obtenir une version spécifique.

Par exemple, pour déployer la dernière version de la plateforme personnalisée ayant l'ARN **`MyCustomPlatformARN`**, qui peut être la version 3.0, la ligne de commande de votre interface de ligne de commande EB se présente comme suit :

```
eb create -p MyCustomPlatformARN
```

Pour déployer la version 2.1, la ligne de commande de votre interface de ligne de commande EB se présente comme suit :

```
eb create -p MyCustomPlatformARN --version 2.1
```

Vous pouvez appliquer des balises à une version de plateforme personnalisée lorsque vous la créez, et modifier des balises de versions de plateforme personnalisée existantes. Pour en savoir plus, consultez Étiquette des versions de plateforme personnalisée.

Création d'une plateforme personnalisée

Pour créer une plateforme personnalisée, la racine de votre application doit inclure un fichier de définition de plateforme `platform.yaml`, qui définit le type de générateur utilisé pour la création de la plateforme personnalisée. Le format de ce fichier est décrit dans Format de fichier platform.yaml. Vous pouvez créer une plateforme personnalisée entièrement nouvelle ou reposant sur l'un des exemples de plateforme personnalisée.

Utilisation d'un exemple de plateforme personnalisée

Une alternative à la création de votre propre plateforme personnalisée consiste à utiliser l'un des exemples d'archive de définition de plateforme pour amorcer votre plateforme personnalisée. Une AMI source et une région constituent les seuls éléments que vous devez configurer dans les exemples avant de pouvoir les utiliser.

> **Note**
>
> N'utilisez pas d'exemple de plateforme personnalisée non modifié en production. L'objectif des exemples vise à illustrer certaines des fonctionnalités disponibles pour une plateforme personnalisée, mais qui n'ont pas été renforcées à des fins de production.

NodePlatform_Ubuntu.zip

Cette plateforme personnalisée est basée sur Ubuntu 16.04 et prend en charge Node.js 4.4.4. Nous allons utiliser cette plateforme personnalisée pour les exemples de cette section.

NodePlatform_RHEL.zip

Cette plateforme personnalisée est basée sur RHEL 7.2 et prend en charge Node.js 4.4.4.

NodePlatform_ AmazonLinux .zip

Cette plateforme personnalisée est basée sur Amazon Linux 2016.09.1 et prend en charge Node.js 4.4.4.

TomcatPlatform_Ubuntu.zip

Cette plateforme personnalisée est basée sur Ubuntu 16.04 et prend en charge Tomcat 7/Java 8.

CustomPlatform_ NodeSampleApp .zip

Exemple Node.js qui utilise express et ejs pour afficher une page web statique.

CustomPlatform_ TomcatSampleApp .zip

Exemple Tomcat qui affiche une page web statique lors de son déploiement.

Téléchargez l'exemple d'archive de définition de plateforme : `NodePlatform_Ubuntu.zip`. Ce fichier contient un fichier de définition de plateforme, un modèle Packer, des scripts exécutés par Packer lors de la création de l'image, et des scripts et des fichiers de configuration que Packer copie sur l'instance de générateur lors de la création de la plateforme.

Example NodePlatform_Ubuntu.zip

```
|-- builder                Contains files used by Packer to create the custom platform
|-- custom_platform.json   Packer template
|-- platform.yaml          Platform definition file
|-- ReadMe.txt             Briefly describes the sample
```

Le fichier de définition de plateforme, `platform.yaml`, indique à Elastic Beanstalk le nom du modèle Packer, `custom_platform.json`.

```
version: "1.0"

provisioner:
  type: packer
  template: custom_platform.json
  flavor: ubuntu1604
```

Le modèle Packer indique à Packer comment créer le AMIs pour la plate-forme, en utilisant une AMI Ubuntu comme base pour l'image de plate-forme pour les types d'instances HVM. La section `provisioners` demande à Packer de copier tous les fichiers dans le dossier `builder` au sein de l'archive vers l'instance et d'exécute le script `builder.sh` sur l'instance. Lorsque les scripts sont terminés, Packer crée une image à partir de l'instance modifiée.

Elastic Beanstalk crée trois variables d'environnement qui peuvent être utilisées pour baliser dans Packer : AMIs

AWS_EB_PLATFORM_ARN

L'ARN de la plateforme personnalisée.

AWS_EB_NOM_PLATEFORME

Le nom de la plateforme personnalisée.

AWS_EB_VERSION_DE PLATEFORME

La version de la plateforme personnalisée.

L'exemple de fichier `custom_platform.json` utilise ces variables pour définir les valeurs suivantes qu'il utilise dans les scripts :

- `platform_name`, qui est définie par `platform.yaml`
- `platform_version`, qui est définie par `platform.yaml`
- `platform_arn`, qui est définie par le script de génération principal, `builder.sh`, qui s'affiche à la fin de l'exemple de fichier `custom_platform.json`.

Le fichier `custom_platform.json` contient deux propriétés pour lesquelles vous devez préciser une valeur : `source_ami` et `region`. Pour plus de détails sur le choix des bonnes valeurs d'AMI et

de région, consultez la section Mise à jour du modèle Packer dans le eb-custom-platforms-samples GitHub référentiel.

Example custom_platform.json

```
{
  "variables": {
    "platform_name": "{{env `AWS_EB_PLATFORM_NAME`}}",
    "platform_version": "{{env `AWS_EB_PLATFORM_VERSION`}}",
    "platform_arn": "{{env `AWS_EB_PLATFORM_ARN`}}"
  },
  "builders": [
    {
      ...
      "region": "",
      "source_ami": "",
      ...
    }
  ],
  "provisioners": [
    {...},
    {
      "type": "shell",
      "execute_command": "chmod +x {{ .Path }}; {{ .Vars }} sudo {{ .Path }}",
      "scripts": [
        "builder/builder.sh"
      ]
    }
  ]
}
```

Les scripts et les autres fichiers que vous incluez dans votre archive de définition de plateforme varient considérablement selon les modifications que vous souhaitez apporter à l'instance. L'exemple de plateforme inclut les scripts suivants :

- `00-sync-apt.sh` – Mis en commentaire : `apt -y update`. Nous avons mis la commande en commentaire parce qu'elle demande à l'utilisateur une entrée, ce qui rompt la mise à jour automatique du package. Cela peut être un problème Ubuntu. Cependant, l'exécution d'`apt -y update` est toujours recommandée comme bonne pratique. Pour cette raison, nous avons laissé la commande dans l'exemple de script à titre de référence.
- `01-install-nginx.sh` – Installe nginx.

- `02-setup-platform.sh` – Installe `wget`, `tree` et `git`. Copie les hooks et les configurations de journalisation sur l'instance, et crée les répertoires suivants :
  - `/etc/SampleNodePlatform` – Emplacement de chargement du fichier de configuration de conteneur pendant le déploiement.
  - `/opt/elasticbeanstalk/deploy/appsource/` – Emplacement où le script `00-unzip.sh` charge le code source de l'application pendant le déploiement (pour plus d'informations sur ce script, consultez la section Outils de script de plateforme pour vos environnements Elastic Beanstalk).
  - `/var/app/staging/` – Emplacement de traitement du code source de l'application pendant le déploiement.
  - `/var/app/current/` – Emplacement d'exécution du code source de l'application après le traitement.
  - `/var/log/nginx/healthd/` – Emplacement où l'agent amélioré de vérification de l'état écrit les journaux.
  - `/var/nodejs` – Emplacement de chargement des fichiers Node.js pendant le déploiement.

Utilisez l'interface de ligne de commande EB pour créer votre première plateforme personnalisée avec l'exemple d'archive de définition de plateforme.

Pour créer une plateforme personnalisée

1. Installez l'interface de ligne de commande EB.
2. Créez un répertoire dans lequel l'exemple de plateforme personnalisée sera extrait.

   ```
   ~$ mkdir ~/custom-platform
   ```

3. Extrayez `NodePlatform_Ubuntu.zip` dans le répertoire, puis modifiez le répertoire extrait.

   ```
   ~$ cd ~/custom-platform
   ~/custom-platform$ unzip ~/NodePlatform_Ubuntu.zip
   ~/custom-platform$ cd NodePlatform_Ubuntu
   ```

4. Modifiez le fichier `custom_platform.json` et fournissez les valeurs des propriétés `source_ami` et `region`. Pour plus d'informations, consultez Updating Packer template (Mise à jour du modèle Packer).
5. Exécutez eb platform init et suivez les instructions pour initialiser un référentiel de plateforme.

Vous pouvez raccourcir eb platform en ebp.

> **Note**
>
> Windows PowerShell l'utilise ebp comme alias de commande. Si vous exécutez l'interface de ligne de commande EB sous Windows PowerShell, utilisez la forme longue de cette commande :eb platform.

```
~/custom-platform$ eb platform init
```

Cette commande crée également le répertoire `.elasticbeanstalk` dans le répertoire actuel et ajoute le fichier de configuration `config.yml` au répertoire. Ne modifiez ou ne supprimez pas ce fichier, car Elastic Beanstalk l'utilise pour créer la plateforme personnalisée.

Par défaut, eb platform init utilise le nom du dossier actuel comme nom de plateforme personnalisée, `custom-platform` dans cet exemple.

6. Exécutez la commande eb platform create pour lancer un environnement Packer et obtenir l'ARN de la plateforme personnalisée. Vous aurez besoin de cette valeur plus tard lorsque vous créerez un environnement en utilisant la plateforme personnalisée.

```
~/custom-platform$ eb platform create
...
```

Par défaut, Elastic Beanstalk crée le profil d'instance `aws-elasticbeanstalk-custom-platform-ec2-role` pour les plateformes personnalisées. Si vous préférez utiliser un profil d'instance existant, ajoutez l'option `-ip` *`INSTANCE_PROFILE`* à la commande eb platform create.

> **Note**
>
> Packer ne parvient pas à créer de plateforme personnalisée si vous utilisez le profil d'instance Elastic Beanstalk par défaut, `aws-elasticbeanstalk-ec2-role`.

L'interface de ligne de commande EB affiche l'événement généré par l'environnement Packer jusqu'à ce que la génération soit terminée. Vous pouvez quitter la vue des événements en appuyant sur Ctrl+C.

7. Vous pouvez consulter les erreurs d'utilisation de la commande eb platform logs dans les journaux.

```
~/custom-platform$ eb platform logs
...
```

8. Vous pouvez vérifier le processus ultérieurement avec eb platform events.

```
~/custom-platform$ eb platform events
...
```

9. Vérifiez l'état de votre plateforme avec eb platform status.

```
~/custom-platform$ eb platform status
...
```

Lorsque l'opération est terminée, vous disposez d'une plateforme que vous pouvez utiliser pour lancer un environnement Elastic Beanstalk.

Vous pouvez utiliser la plateforme personnalisée lors de la création d'un nouvel environnement à partir de la console. Consultez Assistant de création d'un environnement.

Pour lancer un environnement sur votre plateforme personnalisée

1. Créez un nouveau répertoire pour votre application.

```
~$ mkdir custom-platform-app
~$ cd ~/custom-platform-app
```

2. Initialisez un référentiel d'application.

```
~/custom-platform-app$ eb init
...
```

3. Téléchargez l'exemple d'application NodeSampleApp.zip.

4. Procédez à l'extraction de l'exemple d'application.

```
~/custom-platform-app$ unzip ~/NodeSampleApp.zip
```

5. Exécutezeb create -p *CUSTOM-PLATFORM-ARN*, où *CUSTOM-PLATFORM-ARN* est l'ARN renvoyé par une eb platform create commande, pour lancer un environnement exécutant votre plateforme personnalisée.

```
~/custom-platform-app$ eb create -p CUSTOM-PLATFORM-ARN
...
```

Contenu de l'archive de définition de plateforme

Une archive de définition de plateforme est l'équivalent plateforme d'un bundle de fichiers source d'application. L'archive de définition de plateforme est un fichier ZIP qui contient un fichier de définition de plateforme, un modèle Packer, ainsi que les scripts et les fichiers utilisés par le modèle Packer pour créer votre plateforme.

> **Note**
>
> Lorsque vous utilisez l'interface de ligne de commande EB pour créer une plateforme personnalisée, une archive de définition de plateforme est créée à partir des fichiers et des dossiers de votre référentiel de plateforme. Vous n'avez donc pas besoin de créer l'archive manuellement.

Le fichier de définition de plateforme est un fichier au format YAML qui doit être nommé `platform.yaml` et se trouver à la racine de votre archive de définition de plateforme. Pour obtenir la liste des clés obligatoires et facultatives prises en charge dans un fichier de définition de plateforme, consultez Création d'une plateforme personnalisée.

Vous n'avez pas besoin d'attribuer un nom spécifique au modèle Packer, mais le nom du fichier doit correspondre au modèle de fournisseur spécifié dans le fichier de définition de plateforme. Consultez la documentation Packer officielle pour plus d'informations sur la création de modèles Packer.

Les autres fichiers que contient votre archive de définition de plateforme sont des scripts et des fichiers utilisés par le modèle pour personnaliser une instance avant de créer une AMI.

Hooks de plateforme personnalisée

Elastic Beanstalk utilise une structure de répertoires normalisée pour les hooks sur les plateformes personnalisées. Il s'agit de scripts exécutés au cours d'événements du cycle de vie et en réponse à des opérations de gestion : lorsque des instances dans votre environnement sont lancées, ou lorsqu'un utilisateur initie un déploiement ou utilise la fonctionnalité de redémarrage de serveur d'application.

Placez les scripts que vous souhaitez voir déclenchés par les hooks dans l'un des sous-dossiers du dossier `/opt/elasticbeanstalk/hooks/`.

> **Warning**
>
> L'utilisation de hooks de plateforme personnalisée sur des plateformes gérées n'est pas prise en charge. Les hooks de plateforme personnalisée sont conçus pour les plateformes personnalisées. Sur les plateformes Elastic Beanstalk gérées, ils peuvent fonctionner différemment ou présenter quelques problèmes, et le comportement peut varier d'une plateforme à l'autre. Sur les plateformes de l'AMI Amazon Linux (anciennement Amazon Linux 2), ils peuvent toujours s'avérer utiles dans certains cas ; utilisez-les avec prudence. Les hooks de plateforme personnalisés sont une fonctionnalité héritée qui existe sur les plateformes de l'AMI Amazon Linux. Sur les plateformes Amazon Linux 2, les hooks de plateforme personnalisés qui figurent dans le dossier `/opt/elasticbeanstalk/hooks/` sont complètement obsolètes. Elastic Beanstalk ne les lit ou ne les exécute pas. Les plateformes Amazon Linux 2 prennent en charge un nouveau type de hook de plateforme, spécialement conçu pour étendre les plateformes Elastic Beanstalk gérées. Vous pouvez ajouter des scripts et des programmes personnalisés directement dans un répertoire hooks de votre bundle de fichiers source d'application. Elastic Beanstalk les exécute au cours des différentes étapes de provisionnement d'instance. Pour plus d'informations, développez la section Platform Hooks dans the section called “Extension des plateformes Linux”.

Les hooks sont organisés dans les dossiers suivants :

- `appdeploy` &endash; Scripts exécutés lors d'un déploiement d'application. Elastic Beanstalk effectue un déploiement d'application lorsque de nouvelles instances sont lancées et quand un client initie un déploiement de nouvelle version.

- `configdeploy` — Scripts exécutés lorsqu'un client effectue une mise à jour de configuration qui affecte la configuration logicielle sur l'instance, par exemple, en définissant des propriétés d'environnement ou en activant la rotation des journaux sur Amazon S3.
- `restartappserver` &endash; Scripts exécutés lorsqu'un client effectue une opération de redémarrage de serveur d'application.
- `preinit` &endash; Scripts exécutés lors de l'amorçage d'une instance.
- `postinit` &endash; Scripts exécutés après l'amorçage d'une instance.

Les dossiers `appdeploy`, `configdeploy` et `restartappserver` contiennent les sous-dossiers `pre`, `enact` et `post`. Lors de chaque phase d'une opération, tous les scripts du dossier `pre` sont exécutés par ordre alphabétique, puis viennent ceux du dossier `enact`, puis ceux du dossier `post`.

Lorsqu'une instance est démarrée, Elastic Beanstalk exécute `preinit`, `appdeploy` et `postinit`. Sur les déploiements suivants sur des instances en cours d'exécution, Elastic Beanstalk exécute les hooks `appdeploy`. Les hooks `configdeploy` sont exécutés lorsqu'un utilisateur met à jour des paramètres de configuration de logiciel d'instance. Les hooks `restartappserver` sont exécutés uniquement lorsque l'utilisateur lance un redémarrage de serveur d'application.

Lorsque vos scripts détectent des erreurs, ils peuvent quitter avec un état différent de zéro et écrire sur `stderr` pour faire échouer l'opération. Le message que vous écrivez dans `stderr` apparaîtra dans l'événement qui est généré lorsque l'opération échoue. Elastic Beanstalk capture également ces informations dans le fichier journal `/var/log/eb-activity.log`. Si vous ne voulez pas que l'opération échoue, renvoyez 0 (zéro). Les messages que vous écrivez dans `stderr` ou `stdout` apparaissent dans les journaux de déploiement, mais pas dans le flux d'événements, sauf si l'opération échoue.

Nettoyage des instances Packer

Dans certaines conditions, par exemple la suppression du processus du générateur Packer avant sa fin, les instances lancées par Packer ne sont pas nettoyées. Ces instances ne font pas partie de l'environnement Elastic Beanstalk et ne peuvent être consultées et résiliées qu'en utilisant le service Amazon. EC2

Pour nettoyer manuellement ces instances

1. Ouvrez la EC2 console Amazon.
2. Assurez-vous que vous vous trouvez dans la même AWS région que celle dans laquelle vous avez créé l'instance avec Packer.

3. Sous Ressources, choisissez *N* Running Instances, où *N* indique le nombre d'instances en cours d'exécution.
4. Dans la zone de texte de la requête.
5. Sélectionnez la balise Name.
6. Saisissez packer.

   La requête doit se présenter comme suit : tag:Name: packer
7. Sélectionnez les instances qui correspondent à la requête.
8. Si État de l'instance est en cours d'exécution, sélectionnez Actions, État de l'instance, Arrêter, puis Actions, État de l'instance, Résilier.

Format de fichier platform.yaml

Le fichier `platform.yaml` a le format suivant :

```
version: "version-number"

provisioner:
   type: provisioner-type
   template: provisioner-template
   flavor: provisioner-flavor

metadata:
   maintainer: metadata-maintainer
   description: metadata-description
   operating_system_name: metadata-operating_system_name
   operating_system_version: metadata-operating_system_version
   programming_language_name: metadata-programming_language_name
   programming_language_version: metadata-programming_language_version
   framework_name: metadata-framework_name
   framework_version: metadata-framework_version

option_definitions:
   - namespace: option-def-namespace
     option_name: option-def-option_name
     description: option-def-description
     default_value: option-def-default_value

option_settings:
   - namespace: "option-setting-namespace"
```

```
    option_name: "option-setting-option_name"
    value: "option-setting-value"
```

Remplacez les espaces réservés par les valeurs suivantes :

version-number

Obligatoire. La version de la définition YAML. Doit indiquer **1.0**.

provisioner-type

Obligatoire. Le type de générateur utilisé pour créer la plateforme personnalisée. Doit indiquer **packer**.

provisioner-template

Obligatoire. Le fichier JSON contenant les paramètres pour*provisioner-type*.

provisioner-flavor

Facultatif. Le système d'exploitation de base utilisé pour l'AMI. L'un des éléments suivants :

amazon (valeur par défaut)

Amazon Linux. Si la version n'est pas spécifiée, il s'agit de la dernière version d'Amazon Linux disponible au moment de la création de la plateforme.

Amazon Linux 2 n'est pas une version de système d'exploitation prise en charge.

ubuntu1604

Ubuntu 16.04 LTS

rhel7

RHEL 7

rhel6

RHEL 6

metadata-maintainer

Facultatif. Informations de contact pour la personne qui possède la plateforme (100 caractères).

metadata-description

Facultatif. Description de la plateforme (2 000 caractères).

metadata-operating_system_name

Facultatif. Nom du système d'exploitation de la plateforme (50 caractères). Cette valeur est disponible lors du filtrage de la sortie pour l'ListPlatformVersionsAPI.

metadata-operating_system_version

Facultatif. Version du système d'exploitation de la plateforme (20 caractères).

metadata-programming_language_name

Facultatif. Langage de programmation pris en charge par la plateforme (50 caractères).

metadata-programming_language_version

Facultatif. Langue de la version de la plateforme (20 caractères).

metadata-framework_name

Facultatif. Nom de l'infrastructure web utilisée par la plateforme (50 caractères).

metadata-framework_version

Facultatif. Version de l'infrastructure web de la plateforme (20 caractères).

option-def-namespace

Facultatif. Un espace de noms sous `aws:elasticbeanstalk:container:custom` (100 caractères)

option-def-option_name

Facultatif. Nom de l'option (100 caractères). Vous pouvez définir jusqu'à 50 options de configuration personnalisées que la plateforme fournit aux utilisateurs.

option-def-description

Facultatif. Description de l'option (1 024 caractères).

option-def-default_value

Facultatif. Valeur par défaut utilisée lorsque l'utilisateur n'en spécifie pas.

L'exemple suivant crée l'option **`NPM_START`**.

```
options_definitions:
 -  namespace: "aws:elasticbeanstalk:container:custom:application"
    option_name: "NPM_START"
    description: "Default application startup command"
```

```
    default_value: "node application.js"
```

option-setting-namespace

Facultatif. Espace de noms de l'option.

option-setting-option_name

Facultatif. Nom de l'option. Vous pouvez spécifier jusqu'à 50 options fournies par Elastic Beanstalk.

option-setting-value

Facultatif. Valeur utilisée lorsque l'utilisateur n'en spécifie pas.

L'exemple suivant crée l'option **TEST**.

```
option_settings:
 - namespace: "aws:elasticbeanstalk:application:environment"
   option_name: "TEST"
   value: "This is a test"
```

Étiquette des versions de plateforme personnalisée

Vous pouvez appliquer des balises aux versions AWS Elastic Beanstalk personnalisées de votre plateforme. Les balises sont des paires clé-valeur associées à des AWS ressources. Pour plus d'informations sur l'étiquette des ressources Elastic Beanstalk, les cas d'utilisation, les contraintes de clé et de valeur d'identification, et les types de ressources pris en charge, consultez Étiquette des ressources d'application Elastic Beanstalk.

Vous pouvez spécifier des identifications lorsque vous créez une version de plateforme personnalisée. Dans une version de plateforme personnalisée, vous pouvez ajouter ou supprimer des identifications, ainsi que mettre à jour les valeurs des identifications existantes. Vous pouvez ajouter jusqu'à 50 identifications à chaque version de plateforme personnalisée.

Ajout d'identifications lors de la création de versions de plateforme personnalisée

Si vous utilisez l'interface de ligne de commande EB pour créer votre version de plateforme personnalisée, utilisez l'option `--tags` avec eb platform create pour ajouter des identifications.

```
~/workspace/my-app$ eb platform create --tags mytag1=value1,mytag2=value2
```

Avec le AWS CLI ou d'autres clients basés sur une API, ajoutez des balises en utilisant le `--tags` paramètre de la commande. create-platform-version

```
$ aws elasticbeanstalk create-platform-version \
      --tags Key=mytag1,Value=value1 Key=mytag2,Value=value2 \
      --platform-name my-platform --platform-version 1.0.0 --platform-definition-bundle
 S3Bucket=amzn-s3-demo-bucket,S3Key=sample.zip
```

Gestion des identifications d'une version de plateforme personnalisée existante

Vous pouvez ajouter, mettre à jour et supprimer des identifications dans une version existante de la plateforme personnalisée Elastic Beanstalk.

Si vous utilisez l'interface de ligne de commande EB pour mettre à jour votre version de plateforme personnalisée, utilisez eb tags pour ajouter, mettre à jour, supprimer ou répertorier des identifications.

Par exemple, la commande suivante répertorie les identifications dans une version de plateforme personnalisée.

```
~/workspace/my-app$ eb tags --list --resource "arn:aws:elasticbeanstalk:us-east-2:my-
account-id:platform/my-platform/1.0.0"
```

La commande suivante met à jour l'identification `mytag1` et supprime l'identification `mytag2`.

```
~/workspace/my-app$ eb tags --update mytag1=newvalue --delete mytag2 \
      --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:platform/my-
platform/1.0.0"
```

Pour obtenir une liste complète des options et d'autres exemples, consultez `eb tags`.

Avec le AWS CLI ou d'autres clients basés sur une API, utilisez la list-tags-for-resource commande pour répertorier les balises d'une version de plate-forme personnalisée.

```
$ aws elasticbeanstalk list-tags-for-resource --resource-arn
 "arn:aws:elasticbeanstalk:us-east-2:my-account-id:platform/my-platform/1.0.0"
```

Utilisez la commande update-tags-for-resource pour ajouter, mettre à jour ou supprimer des identifications dans une version de plateforme personnalisée.

```
$ aws elasticbeanstalk update-tags-for-resource \
```

```
      --tags-to-add Key=mytag1,Value=newvalue --tags-to-remove mytag2 \
      --resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:platform/my-
platform/1.0.0"
```

Spécifiez les identifications à ajouter et les identifications à mettre à jour dans le paramètre `--tags-to-add` de update-tags-for-resource. Une identification inexistante est ajoutée et la valeur d'une identification existante est mise à jour.

> **Note**
>
> Pour utiliser certaines AWS CLI commandes et certaines de l'interface de ligne de commande EB avec une version de plateforme personnalisée d'Elastic Beanstalk, vous avez besoin de l'ARN de cette version personnalisée. Vous pouvez extraire l'ARN à l'aide de la commande suivante.
>
> ```
> $ aws elasticbeanstalk list-platform-versions
> ```
>
> Utilisez l'option `--filters` pour filtrer la sortie vers le nom de votre plateforme personnalisée.

Historique du document

Le tableau suivant décrit les modifications importantes apportées au Guide du AWS Elastic Beanstalk développeur depuis avril 2024.

Modification	Description	Date
AWS mises à jour de politiques gérées pour `AWSElasticBeanstalkWorkerTier`	Permissions mises à jour dans la politique AWS gérée pour l'analyse de l'environnement basée sur l'IA.	11 mars 2026
AWS mises à jour de politiques gérées pour `AWSElasticBeanstalkWebTier`	Permissions mises à jour dans la politique AWS gérée pour l'analyse de l'environnement basée sur l'IA.	11 mars 2026
AWS mises à jour de politiques gérées pour `AWSElasticBeanstalkMulticontainerDocker`	Permissions mises à jour dans la politique AWS gérée pour l'analyse de l'environnement basée sur l'IA.	11 mars 2026
Nouveau sujet : Analyse de l'environnement basée sur l'IA	Elastic Beanstalk ajoute une analyse basée sur l'IA pour aider à identifier les causes profondes et à fournir des solutions aux problèmes de santé environnementale à l'aide d'Amazon Bedrock.	1er décembre 2025
AWS mises à jour de politiques gérées pour `AdministratorAccess-AWSElasticBeanstalk`	Autorisations mises à jour dans la politique AWS gérée.	12 novembre 2025
Nouveau sujet : Configuration des équilibreurs de charge Elastic Beanstalk à double pile	Elastic Beanstalk prend en charge votre environnement afin qu'il IPv4 serve à la	9 septembre 2025

	IPv6 fois les protocoles et les protocoles en proposant l'option de configuration à double pile pour les équilibre urs de charge des applications et du réseau.	
Sujet révisé : Qu'est-ce que c'est AWS Elastic Beanstalk ?	Nouveau schéma d'archite cture et introduction condensée pour plus de clarté.	5 juin 2025
Rubrique révisée : Configura tion de l'EB CLI	Instructions d'installation condensées et ciblées.	5 juin 2025
Rubrique révisée : PHP Quickstart	Ajout d'une étape pour mettre à jour et redéployer l'application et amélioration de certaines instructions.	5 juin 2025
Rubrique révisée : Tutoriel de mise en route	Simplification du didacticiel de mise en route pour plus de clarté et une intégration plus rapide.	5 juin 2025
Contenu réorganisé	Structure de contenu réorganis ée pour mettre en évidence les informations essentielles et améliorer la découvrabilité.	5 juin 2025
Nouveau sujet : Gestion des groupes de sécurité EC2	Elastic Beanstalk ajoute la possibilité de désactiver l'environnement du groupe de sécurité EC2 par défaut.	30 avril 2025

Nouveau sujet : Migration des applications IIS vers Elastic Beanstalk	Elastic Beanstalk ajoute une fonctionnalité permettant de migrer vos applications Windows IIS vers. AWS Elastic Beanstalk	18 avril 2025
Nouveau sujet : Utilisation d'Elastic AWS Secrets ManagerAWS Systems Manager Beanstalk avec et Parameter Store	Elastic Beanstalk prend en charge AWS Secrets Manager les AWS Systems Manager secrets de référence et les paramètres du Parameter Store avec des variables d'environnement.	31 mars 2025
AWS mises à jour des politiques gérées pour`AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy` .	Autorisations mises à jour dans la politique AWS gérée.	27 février 2025
Configurations de lancement déconseillées au profit des modèles de lancement	Nous avons révisé cette rubrique pour expliquer comment les environnements Elastic Beanstalk sont affectés par la dépréciation des configurations de lancement par Amazon EC2 Auto Scaling au profit des modèles de lancement. Nous avons également révisé d'autres contenus relatifs aux configurations de lancement et aux modèles de lancement.	23 janvier 2025

Nouveau sujet : Utilisation d'EC2 Fast Launch avec des branches de plate-forme Windows	Les versions de la plateform e Windows Elastic Beanstalk incluent des AMI de base avec EC2 Fast Launch activé.	22 janvier 2025
Nouveau sujet : Stratégie d'allocation des instances Spot	Elastic Beanstalk prend en charge la configuration de la stratégie d'allocation ponctuelle lors de la création de l'environnement.	15 janvier 2025
`AdministratorAcces s-AWSElasticBeanst alk` AWS stratégie gérée	Autorisations mises à jour dans la politique AWS gérée.	11 décembre 2024
Recommandations pour les environnements de première vague de Graviton arm64 au chapitre sur le contenu archivé	Les recommandations d'EB pour les environnements de première vague de Graviton arm64 ont été déplacées vers le chapitre sur le contenu archivé	5 octobre 2024
Modèles de lancement	Nouveau sujet : Modèles de lancement. Cette rubrique explique comment Amazon EC2 Auto Scaling abandonne progressivement les configura tions de lancement au profit des modèles de lancement.	1er octobre 2024
QuickStartfor Python	Nouveau sujet : QuickStart pour Python	24 septembre 2024
Autorisations de compartim ent Amazon S3 requises pour les politiques de point de terminaison VPC restrictives	Nouveau sujet : Autorisations de compartiment Amazon S3 requises pour les politiques de point de terminaison VPC restrictives	18 septembre 2024

QuickStart pour Java sur Tomcat	Nouveau sujet : QuickStart pour l'application Java JSP exécutée sur Tomcat	12 septembre 2024
QuickStart pour Java	Nouveau sujet : QuickStart pour Java	12 septembre 2024
Utilisation d'Amazon EC2 Systems Manager pour les référentiels privés Docker	Rubrique rétablie : Utilisation d'Amazon EC2 Systems Manager pour les référentiels privés Docker. Remplace le sujet Utilisation de AWS Secrets Manager pour les référentiels privés Docker jusqu'à ce que l'intégration Elastic Beanstalk/SSM soit prise en charge.	8 septembre 2024
EB CLI 2.6 (retiré) vers le chapitre sur le contenu archivé	EB CLI 2.6 (retiré) a été déplacé vers le chapitre sur le contenu archivé	15 août 2024
API EB (retirée) vers le chapitre sur le contenu archivé	API EB déplacée (retirée) vers le chapitre sur le contenu archivé	15 août 2024
Déploiement d'applications Elastic Beanstalk à partir de conteneurs Docker	Mise à jour et réorganisation : déploiement d'applications Elastic Beanstalk à partir de conteneurs Docker	15 août 2024
Contenu archivé	Nouveau chapitre : Contenu archivé	15 août 2024
QuickStart pour Windows ASP.NET	Nouveau sujet : QuickStart pour Windows ASP.NET	5 juillet 2024

QuickStart pour .NET Core sous Windows	Nouveau sujet : QuickStart pour .NET Core sous Windows	28 juin 2024
QuickStart pour Docker	Nouveau sujet : QuickStart pour Docker	19 juin 2024
Empêcher l'accès aux compartiments Amazon S3 entre environnements	Nouveau sujet : Empêcher l'accès aux compartiments Amazon S3 entre environnements	12 juin 2024
QuickStart pour .NET Core sous Linux	Nouveau sujet : QuickStart pour .NET Core sous Windows	28 mai 2024
QuickStart pour PHP	Nouveau sujet : QuickStart pour PHP	10 mai 2024
QuickStart pour Node.js	Nouveau sujet : QuickStart pour Node.js	5 mai 2024
QuickStart pour Go	Nouveau sujet : QuickStart for Go	5 mai 2024
Calendrier de sortie de la plateforme Elastic Beanstalk	Ajout d'un nouveau sujet qui inclut un calendrier deProchaines versions des branches de la plateforme. Déplacé Calendrier de mise hors service des branches de plateforme et Historique des branches de plateforme retirées vers ce sujet.	1 mai 2024
`AWSElasticBeanstalkRoleCore` AWS stratégie gérée	Autorisations mises à jour dans la politique AWS gérée.	30 avril 2024

AWSElasticBeanstalkManagedUpdatesServiceRolePolicy AWS stratégie gérée	Autorisations mises à jour dans la politique AWS gérée.	30 avril 2024
AWSElasticBeanstalkManagedUpdatesInternalServiceRolePolicy AWS stratégie gérée	Autorisations mises à jour dans la politique AWS gérée.	30 avril 2024
AWSElasticBeanstalkMaintenance AWS stratégie gérée	Autorisations mises à jour dans la politique AWS gérée.	30 avril 2024
AWSElasticBeanstalkInternalMaintenanceRolePolicy AWS stratégie gérée	Autorisations mises à jour dans la politique AWS gérée.	30 avril 2024

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.