ユーザーガイド

AWS DevOps エージェント

AWS DevOps エージェント: ユーザーガイド

Table of Contents

AWS DevOps エージェントについて

AWS DevOps エージェントは、インシデントを解決してプロアクティブに防止し、信頼性とパフォーマンスを継続的に向上させるフロンティアエージェントです。

AWS DevOps エージェントはインシデントを調査し、経験豊富な DevOps エンジニアとして運用上の改善を特定します。

エージェントは次の方法で動作します。

- リソースとその関係について学習します。
- オブザーバビリティツール、スキル、コードリポジトリ、CI/CD パイプラインの操作。
- テレメトリ、コード、デプロイデータを関連付けて、アプリケーションリソース間の関係を理解します。
- マルチクラウドおよびハイブリッド環境でのアプリケーションのサポート。

主な特徴

AWS DevOps Agent は、以下の機能を通じて包括的なインシデント対応と防止機能を提供します。

常時オンの自律型インシデント対応

AWS DevOps Agent は、問題が発生した瞬間に、問題を自動的に調査します。

- 自動インシデント調査 – アラートまたはサポートチケットが届いたらすぐに調査を開始します
- AWS DevOps エージェントチャット - DevOps エージェントスペースウェブアプリ全体で、インフラストラクチャのクエリ、システムの状態の分析、自然言語を使用した調査のガイドを行います。チャットは、トポロジのリソースについて質問するか、調査を運営するか、防止のレコメンデーションをフィルタリングするかにかかわらず、表示しているページに基づいてコンテキスト対応のレスポンスを提供します。
- 詳細な緩和計画 – インシデントを解決し、成功を検証し、必要に応じて変更を元に戻すための特定のアクションを提供します。
- インシデントの自動調整 – Slack や ServiceNow などの希望するコミュニケーションチャネルを通じて、観察結果、検出結果、緩和ステップをルーティングします。
- AWS サポート統合 – AWS サポートエキスパートに提供される即時コンテキストを使用して、調査から直接 AWS サポートケースを作成する

将来のインシデントを防ぐ

AWS DevOps Agent は、過去のインシデントのパターンを分析して、事後対応的な発砲から積極的な運用改善への移行を支援します。

- ターゲットを絞った推奨事項 – オブザーバビリティ (モニタリング、アラート、ログ記録) 、インフラストラクチャの最適化 (自動スケーリング、容量調整) 、デプロイパイプラインの強化 (テスト、検証) の 4 つの主要領域を強化する、具体的で実用的な改善を提供します。
- 継続的な学習 – チームのフィードバックに基づいてレコメンデーションを絞り込みます

DevOps ツールを最大限に活用する

AWS DevOps Agent は、ワークフローを変更せずに既存のツールと統合します。

- アプリケーションリソースマッピング – アプリケーションリソースとその関係のトポロジグラフを構築します。
- 組み込みの統合 – 一般的なオブザーバビリティツール (Amazon CloudWatch、Dynatrace、Datadog、New Relic、Splunk)、コードリポジトリ、CI/CD パイプライン (GitHub Actions and repositories、GitLab ワークフローとリポジトリ) と連携します。
- カスタムツール統合 – 追加のツール用に独自のモデルコンテキストプロトコル (MCP) サーバーに接続して機能を拡張します。
- 対話型インフラストラクチャクエリ – 自然言語を使用して、複数のコンソールをナビゲートすることなく、 AWS リソース、システムメトリクス、アラームステータスをクエリします。チャットはコンテキストを理解し、フォローアップの質問の会話履歴を維持します。

AWS DevOps エージェントの仕組み

AWS DevOps Agent は、デュアルコンソールアーキテクチャで動作します。管理者は AWS マネジメントコンソールを使用して、エージェントスペースの作成と管理、統合の設定、アクセスコントロールの設定を行います。運用チームは、DevOps AWS DevOps Agent ウェブアプリを使用して day-to-dayインシデント対応と調査活動を行います。ウェブアプリケーションは、オペレーターがエージェント調査を操作し、クロスアカウントアプリケーショントポロジを参照し、オブザーバビリティ、コード、パイプライン、インフラストラクチャアーキテクチャの予防的改善について学習できる場所です。詳細についてはthe section called “プロアクティブインシデント防止”を参照してください。

このサービスは、 AWS DevOps エージェントがアクセスして調査できる内容を定義する論理コンテナである エージェントスペースを中心に編成されています。各エージェントスペースには、 AWS アカウント設定、サードパーティーのツール統合、アクセス許可が含まれます。詳細についてはthe section called “DevOps エージェントスペースとは”を参照してください。

AWS DevOps エージェントは、リソースとその関係をマッピングするアプリケーショントポロジを自動的に構築します。このトポロジは、調査中にサービスがアプリケーションアーキテクチャを理解するのに役立ちます。詳細についてはthe section called “DevOps エージェントトポロジとは”を参照してください。

利点

- 平均解決時間 (MTTR) の短縮 – 自動調査がすぐに開始され、インシデント解決が数時間から数分に短縮されます
- 定期的なインシデントの防止 – ターゲットを絞った推奨事項は根本原因に対処し、システムの耐障害性を強化します。
- 運用効率の向上 – チームが反復的な調査タスクから解放され、イノベーションに集中できるようにします。
- 既存のワークフロー内での作業 – 中断することなく既存のツールやプロセスと統合します。

DevOps エージェントウェブアプリとは

AWS DevOps エージェントは、管理機能をday-to-dayから分離するデュアルコンソールアーキテクチャを使用します。この設計により、管理者はサービスを設定し、運用チームはインシデント対応と防止に集中できます。

コンソール

AWS DevOps エージェントには 2 つの異なるインターフェイスがあります。

- AWS マネジメントコンソール – 管理者は AWS マネジメントコンソールを使用して AWS DevOps エージェントをセットアップおよび管理します。このコンソールでは、 AWS サービスとサードパーティーツールthe section called “エージェントスペースの作成”を接続し、組織のアクセス許可を管理できます。
- DevOps エージェントウェブアプリ - 運用チームは、DevOps エージェントスペースウェブアプリを毎日のインシデント対応アクティビティに使用します。このスタンドアロンアプリケーションは、オンコールエンジニアが調査を開始したり、自然言語チャットを通じてエージェントとやり取

りしたり、アプリケーショントポロジを表示したり、インシデント防止の推奨事項を確認したりできるインターフェイスを提供します。

ウェブアプリの機能

DevOps Agent ウェブアプリには、次の主要な機能があります。

- インシデント対応 – このページでは、インシデント調査を作成および追跡し、インシデントを解決するための緩和計画を生成します。
- Incident Prevention – Prevention ページにあります。ここでは、オブザーバビリティ体制、配信プロセス、インフラストラクチャアーキテクチャを改善して、将来のインシデントを防ぐための推奨事項を示します。
- トポロジ – トポロジページには、接続されたアカウントのすべてのリソースにおけるアカウントリソースとその関係がインタラクティブに視覚的に表示されます。システムビュー、コンテナビュー、リソースビューを切り替えるには、「表示」ドロップダウンを使用して、さまざまな詳細レベルでトポロジを表示できます。
- スキル – DevOps AWS DevOps エージェントを特殊な機能で拡張するモジュラー命令セット。スキルには、ドメインの知識、調査方法、インフラストラクチャに合わせたツール設定が含まれます。各スキルは特定のツールを有効にし、調査に関連する場合にのみ指示を段階的に開示します。
- 自然言語チャットインターフェイス – ウェブアプリ全体で利用できる Chat は、AI を活用した会話アシスタントで、インフラストラクチャのクエリ、システムの状態の分析、自然言語を使用した調査の操作を行うことができます。チャットは、表示しているページに基づいてコンテキスト対応のレスポンスを提供します。

認証

AWS DevOps Agent は、さまざまな組織要件に対応するための柔軟な認証方法をサポートしています。

- IAM Identity Center 統合 (ユーザーアクセス) – 組織は AWS Identity Center (IAM Identity Center) を使用して、DevOps エージェントスペースウェブアプリケーションへのユーザーアクセスを一元管理できます。IAM Identity Center は、Okta、Ping Identity、Microsoft Entra ID などのプロバイダーを含む、標準の OIDC および SAML プロトコルを通じて外部 ID プロバイダーとフェデレーションできます。このメソッドは、ID プロバイダーからの多要素認証をサポートします。
- 外部 ID プロバイダー (IdP) 認証 – 組織は、IAM アイデンティティセンターを必要とせずに、Okta や Microsoft Entra ID などの OIDC 互換 ID プロバイダーをエージェントスペースウェブアプリに

直接接続できます。ユーザーは IdP を通じて会社の認証情報でサインインします。セットアップ手順については、「」を参照してくださいthe section called “外部 ID プロバイダー (IdP) 認証の設定”。

- IAM 認証リンク (管理者アクセス) – 別の方法では、既存のコンソールセッションを使用して AWS マネジメントコンソールからウェブアプリに直接アクセスできます。このオプションは、完全な Identity Center 統合を実装する前に便利ですが、セッションは 10 分に制限されています。

DevOps エージェントスペースとは

DevOps エージェントスペースは、 AWS DevOps エージェントがアクセスできるツールとインフラストラクチャを定義する論理コンテナです。各エージェントスペースは、独自の AWS アカウントアクセス、サードパーティー統合、およびユーザーアクセス許可を使用して独立して動作します。

エージェントスペースは、インシデント対応中に AWS DevOps エージェントがアクセスして調査できる範囲を表します。エージェントスペースを作成するときは、エージェントがアクセスできる AWS アカウント、接続できる外部ツール、組織内のユーザーがエージェントとやり取りできるアカウントを定義します。

各エージェントスペースは、 AWS DevOps エージェントの独立したデプロイとして機能します。エージェントスペースは AWS マネジメントコンソールを使用して設定しますが、運用チームはエージェントスペースのウェブアプリを使用して調査を行い、そのスペース内のレコメンデーションを確認します。

エージェントスペースの分離方法

エージェントスペースは分離を維持し、セキュリティを確保し、さまざまな環境やチーム間で意図しないアクセスを防止します。

- AWS アカウントの分離 – 各エージェントスペースは、特定の AWS アカウントとリソースにのみアクセスを許可する専用の IAM ロールを使用します。エージェントは、エージェントスペースに明示的に設定されたリソース以外の AWS リソースにアクセスすることはできません。
- ユーザーアクセスの分離 – 各エージェントスペースにアクセスできるユーザーまたはグループを制御します。これにより、アクセス許可を組織構造に合わせて調整できるため、チームは指定されたエージェントスペースとのみやり取りできます。
- データ分離 – 調査データ、インシデント履歴、推奨事項は、各エージェントスペース内で個別に維持されます。あるエージェントスペースからの情報は、別のエージェントスペースからは表示またはアクセスできません。

- チャットデータの分離 - チャット会話履歴も各エージェントスペース内で分離されます。あるエージェントスペースの会話とクエリは、別のエージェントスペースからは表示またはアクセスできません。

エージェントスペースウェブアプリ

各エージェントスペースには、 AWS マネジメントコンソールの外部からアクセスできる専用のウェブアプリがあります。ウェブアプリの詳細については、the section called “DevOps エージェントウェブアプリとは”「」を参照してください。

複数のエージェントスペースを使用するタイミング

さまざまな組織のニーズをサポートするために、複数のエージェントスペースを作成することを検討してください。

- チーム分離 – さまざまなアプリケーションチームまたはビジネスユニット専用のエージェントスペースを作成し、エージェントスペースに明確な所有権の境界を維持します。
- 環境の分離 – 本番環境と非本番環境を異なるエージェントスペースに分割して、誤って環境間にアクセスするのを防ぎます。
- サービス境界 – エージェントスペースを特定のサービスまたはアプリケーションの境界に合わせ、調査の焦点と関連性を維持します。
- コンプライアンス要件 – 規制要件を満たすために、異なるアクセスコントロールまたはデータレジデンシー設定で個別のエージェントスペースを設定します。

> **Note**
>
> 複数のエージェントスペースを作成するときは、専用 AWS アカウントをエージェントスペースのプライマリアカウントとして使用し、個別のアプリケーションアカウントをセカンダリアカウントとして接続できます。このアプローチにより、ロールの自動作成を使用しても、各エージェントスペースが意図した範囲に固有のリソースにのみアクセスできるようにしながら、きめ細かなアクセスコントロールを維持できます。

DevOps エージェントトポロジとは

AWS DevOps Agent の は、アプリケーション内のリソースと関係を自動的に検出して視覚化し、結果のトポロジーを使用して、インシデントの調査中や予防的推奨事項の作成時にインフラストラクチャを理解します。

トポロジグラフの作成方法

AWS DevOps Agent は、いくつかの自動化されたプロセスを通じてトポロジグラフを構築します。

- リソース検出 – エージェントは AWS アカウントを自動的にスキャンして、コンピューティングインスタンス、ストレージサービス、ネットワークコンポーネント、アプリケーションの一部であるデータベースなどのリソースを識別します。
- 関係検出 – エージェントは、設定データ、CloudFormation スタック、リソースタグを分析して、リソースが相互にどのように関連しているかを判断します。
- コードとデプロイのマッピング – CI/CD パイプラインに接続すると、エージェントはインフラストラクチャリソースをデプロイプロセスにリンクし、アプリケーションとインフラストラクチャコードを変更します。
- オブザーバビリティ動作マッピング – Amazon CloudWatch Application Signals や Dynatrace などのオブザーバビリティシステムからのデータは、リソース間の関係を示す観測された動作を識別するために使用されます。

主な機能

リソースマッピングは、インシデントの調査と防止を強化するいくつかの機能を提供します。

- インタラクティブビジュアライゼーション – Operator Web App のインタラクティブグラフを使用してアプリケーショントポロジを調べます。トポロジを拡大およびナビゲートして、リソース間の複雑な関係を理解できます。Chat を使用して、「Show me all Lambda functions connected to this DynamoDB table」や「What resources are affected by this alarm?」などの自然言語を使用してトポロジ情報をクエリすることもできます。
- コンテキスト調査 – インシデント調査中、 AWS DevOps Agent はリソーストポロジーの支援を受けて、影響を受けるコンポーネントを特定し、爆発半径を理解し、システム全体の影響パスを追跡します。
- 根本原因分析 – リソース関係の詳細な理解は、多くの相互依存関係を持つ複雑な分散システムであっても、問題の発生場所を特定するのに役立ちます。

- 影響評価 – インシデントを分析する際、エージェントはトポロジ内の依存関係チェーンを特定することで、どのダウンストリームサービスが影響を受けるかをより的確に判断できます。
- 予防的レコメンデーション – エージェントはトポロジに関するインサイトを使用して、回復力向上のための的を絞ったレコメンデーションを作成し、システムの安定性に最も大きな影響を与える変更を提案します。

トポロジビュー

Operator Web App のトポロジページのトポロジの視覚化には、複数の詳細レベルがあります。

- 学習済み – エージェントスペース理解スキルから生成されたデフォルトビュー。論理サービスとリクエストパス別に整理されたインフラストラクチャの構造化された概要を表示します。
- システム – 高レベルのアカウントとリージョンの境界を表示します。
- コンテナ – 関連するリソースを含む CloudFormation スタックなどのデプロイスタックを表示します。
- コンポーネント – コンテナ内の個々のコンポーネントとその関係を表示します。
- すべてのリソース – 検出されたすべてのリソースとその関係を含む完全なビューを表示します。

リソース検出

リソースは 2 つの方法で検出されます。

- CloudFormation スタック – エージェントは、プライマリ AWS アカウントと接続されたセカンダリアカウントのすべての CloudFormation スタックとそのリソースを一覧表示します。これは、AWS Cloud Development Kit (AWS CDK) など、デプロイに CloudFormation を使用するすべての infrastructure-as-code ツールでサポートされています。
- Resource Explorer – CloudFormation からデプロイされていないリソースの場合、タグ付けされたリソースは AWS Resource Explorer から検出されます。ターゲット AWS アカウントでは、Resource Explorer が有効になっている必要があります。これは、 AWS マネジメントコンソール、 AWS サービス APIs、またはその他の infrastructure-as-code フレームワークを通じてデプロイされたリソースのアプリケーション境界を特定するのに役立ちます。

トポロジーを超えた調査範囲

アプリケーショントポロジは調査中に重要なコンテキストを提供しますが、 AWS DevOps Agent はトポロジに表示されるリソースのみを調査することに限定されません。エージェントは、 AWS サービス APIs や接続されたオブザーバビリティツールなどの追加のデータソースを使用して、アプリケーショントポロジにないリソースを調査できます。

エージェントがアクセスできるリソースを制限するには、エージェントに割り当てられたロールのポリシーを制限して、クロスアカウントリソースにアクセスします。詳細については、「the section called “AWS アカウントでのエージェントアクセスの制限”」を参照してください。

トポロジとエージェントスペースの理解スキル

トポロジグラフは、調査中に使用するインフラストラクチャの構造化された概要をエンコードする Agent Space Understanding learned スキルにフィードされます。新しいエージェントスペースのトポロジ検出が完了すると、システムはエージェントスペース理解スキルを自動的に生成します。学習したスキルの詳細については、「」を参照してくださいthe section called “学習したスキル”。

DevOps エージェントスキル

AWS DevOps エージェントスキルは、インフラストラクチャと運用ワークフローに合わせた専門的なドメインの知識と調査方法論により、エージェントの機能を拡張するモジュラー命令セットです。

スキルとは

スキルは、 AWS DevOps Agent に特殊な機能を提供する Markdown 指示を含む自己完結型ディレクトリです。 AWS DevOps Agent は、エージェントの指示とリソースをパッケージ化するオープンスタンダードである Agent Skills 仕様のサブセットをサポートし、Markdown 指示書、PDFs、画像、データファイルなどの実行不可能なドキュメントのみをサポートします。

すべてのスキルには、 AWS DevOps エージェントに提供する手順を含む SKILL.md ファイルが必要です。必要な SKILL.md ファイルに加えて、スキルには次のようなものがあります。

- 特定のシナリオまたはインフラストラクチャタイプの調査ワークフロー。
- アーキテクチャパターンや運用手順を含む参考資料。
- エージェントタイプのターゲティング – スキルは、特定のエージェントタイプ (汎用、オンデマンド、インシデントトリアージ、インシデント RCA、インシデント緩和、評価) を対象にして、コンテキストの消費を減らし、エージェントのフォーカスを向上させることができます。

スキルを使用する理由

スキルは AWS DevOps エージェントを汎用アシスタントからインフラストラクチャと運用ワークフローのスペシャリストに変換します。チャットメッセージで提供される 1 回限りの手順とは異なり、スキルは、 AWS DevOps Agent によって実行されるタスクに関連する場合に自動的にロードされる再利用可能な機能です。

主な利点:

- エージェントを専門化する – 調査手順、ベストプラクティス、インフラストラクチャと運用パターンに固有の組織知識を使用して、 AWS DevOps エージェントをカスタマイズします。
- 繰り返しを減らす – 調査ワークフローを一度作成すると、 AWS DevOps Agent は関連するすべての調査でそれらを自動的に使用するため、同じガイダンスを繰り返し提供する必要がなくなります。
- 機能を構成する – 複数のスキルを組み合わせてend-to-endの調査ワークフローを構築します。AWS DevOps エージェントは、カスタム CI/CD パイプラインからデプロイを取得するスキルやコードリポジトリを検索するスキルなど、実行中に複数のスキルを読み取ります。
- Amplify カスタムツール – AWS DevOps Agent がカスタム MCP サーバーツールを効果的に使用するのに役立つスキルを作成します。スキルは、特定のツールを呼び出すタイミング、さまざまなシナリオに使用するパラメータ、結果を解釈してインフラストラクチャ固有のワークフローを実行する方法を文書化することができます。

スキルの仕組み

AWS DevOps エージェントが関連するタスクに遭遇すると、適切なスキルをロードし、調査の指針となる指示に従います。例えば、「データベースパフォーマンス調査」スキルには、RDS スロットリングの問題を分析するstep-by-stepの手順が含まれている場合があります。これにより、エージェントはアラームステータスを体系的にチェックし、接続メトリクスを分析し、スロークエリを特定できます。

スキル構造

スキルは、以下を含むディレクトリとして編成されます。

```
my-skill/
### SKILL.md              # Main skill instructions
### references/          # Optional: additional reference documentation
```

```
### assets/               # Optional: images, diagrams, data files
```

SKILL.md

は唯一の必須ファイルSKILL.mdです。これには、Markdown 形式で記述されたコア命令が含まれています。このファイルは次の条件を満たす必要があります。

- スキルを使用するタイミングと方法について説明します。
- step-by-stepの調査手順を提供します。
- さまざまなシナリオのディシジョンツリーを含めます。
- 期待される出力と成功基準を文書化します。

フロントマター

Frontmatter は、SKILL.mdファイルの上部にあるメタデータブロックで、---区切り文字で囲まれています。これには、 AWS DevOps Agent が調査またはタスク中にスキルを有効にするタイミングを判断するために使用する nameおよび descriptionフィールドが含まれます。

```
---
name: rds-performance-investigation
description: Investigation procedures for RDS performance issues including
  connection exhaustion, slow queries, replication lag, and storage capacity.
  Use this skill when investigating database latency, connection errors, or
  read/write performance degradation.
---
```

name – スキルの一意の識別子。小文字、数字、ハイフンのみを使用してください (最大 64 文字) 。先頭または末尾にハイフンを使用しないでください。

説明 – AWS DevOps エージェントがこのスキルを使用するタイミングと理由の詳細な説明。 AWS DevOps エージェントはこのフィールドを評価して、スキルが現在のタスクに関連しているかどうかを判断します。説明があいまいまたは欠落している場合、指示が適切に記述されていても、エージェントはスキルを完全にスキップする可能性があります。

重要 – エージェントの観点から説明を記述します。スキルをトリガーする特定のシナリオ、サービス、エラータイプ、または症状を含めます。たとえば、「Amazon RDS インスタンスのデータベースレイテンシー、接続エラー、またはクエリタイムアウトを調査するときにこのスキルを使用する」は、「RDS スキル」よりも効果的です。

UI でスキルを作成すると、指定した名前と説明からフロントマターが自動的に生成されます。zip ファイルとしてアップロードされるスキルには、SKILL.mdファイルに frontmatter が含まれている必要があります。

例: スキルを完了する

次の例は、RDS パフォーマンスの問題を調査するための完全で適切な形式のスキルを示しています。ディレクトリ構造、SKILL.md frontmatter、実用的な調査手順、および補足参照ファイルを示します。

ディレクトリ構造:

```
rds-performance-investigation/
### SKILL.md
### references/
#   ### rds-metrics-reference.md
### assets/
    ### rds-investigation-flowchart.png
```

SKILL.md:

```
---
name: rds-performance-investigation
description: Investigation procedures for RDS performance issues including
  connection exhaustion, slow queries, replication lag, and storage capacity.
  Use this skill when investigating database latency, connection errors, or
  read/write performance degradation.
---

# RDS Performance Investigation

Use this skill when customers report database latency, connection errors,
query timeouts, or read/write performance degradation.


## Step 1: Check alarm status

Query CloudWatch for active alarms on the affected RDS instance. Look for:
- `DatabaseConnections` exceeding 80% of max_connections
- `ReadLatency` or `WriteLatency` above 20ms
- `FreeStorageSpace` below 20% of total storage
- `ReplicaLag` above 30 seconds (read replicas only)
```

```
## Step 2: Analyze connection metrics

Retrieve `DatabaseConnections` over the past hour. If connections are near
the max_connections limit, check for connection pool misconfiguration or
long-running idle connections.


## Step 3: Identify slow queries

Use Performance Insights (`pi:GetResourceMetrics`) to retrieve the top SQL
statements by average active sessions. Focus on queries with high `db.load`
contribution or frequent I/O waits.


## Step 4: Summarize findings

Provide a summary with:
1. Current performance status (healthy / degraded / critical)
2. Root cause hypothesis with supporting metrics
3. Recommended remediation steps ranked by priority
```

リファレンス/rds-metrics-reference.md:

```
# RDS CloudWatch Metrics Reference


| Metric | Normal Range | Investigation Threshold |
|---|---|---|
| DatabaseConnections | < 70% max_connections | > 80% max_connections |
| ReadLatency | < 5ms | > 20ms |
| WriteLatency | < 5ms | > 20ms |
| FreeStorageSpace | > 30% total storage | < 20% total storage |
| ReplicaLag | < 5 seconds | > 30 seconds |
| CPUUtilization | < 70% | > 85% |
```

スキルの作成

スキルを作成する前に、エージェントスペースが必要です。詳細については、「the section called "エージェントスペースの作成"」を参照してください。

ワークフローの設定とスキルの複雑さに応じて、2 つの方法でスキルを作成できます。

UI でのスキルの作成

AWS DevOps エージェントオペレーターウェブアプリで作成されたスキルには、単一の SKILL.md ファイルの名前、説明、および手順が含まれています。

UI でスキルを作成するには:

- エージェントスペースオペレーターウェブアプリのスキルページに移動します。
- 「スキルの追加」をクリックします。
- モーダルから「スキルの作成」を選択します。
- スキルフォームに入力します。
  - 名前 – 小文字、数字、ハイフンのみ (最大 64 文字) 。先頭または末尾にハイフンを使用しないでください。例: `rds-throttling-investigation`
  - 説明 – このスキルを使用するタイミングの簡単な説明 (推奨文字数は 100 文字以上、最大 1,024 文字) 。これは、エージェントがスキルを有効にするタイミングを決定するのに役立ちます。
  - ステータス – アクティブ (デフォルト) または非アクティブに設定します。非アクティブなスキルはエージェントでは使用されません。
  - エージェントタイプ – このスキルを使用できるエージェントタイプを 1 つ以上選択します。汎用 はデフォルトで選択され、スキルがすべてのエージェントタイプで使用できるようにします。特定のエージェントをターゲットにするには、汎用 の選択を解除し、オンデマンド、インシデントトリアージ、インシデント RCA、インシデント緩和、評価のいずれかを選択します。
  - 手順 – Markdown Step-by-stepの手順。具体的で実用的であること。
- スキルを保存するには、「作成」をクリックします。

システムは、適切なフロントマター構造を持つ SKILL.md ファイルを自動的に生成します。

UI で作成されたスキルを編集するには:

- スキルリストのスキルに移動し、スキルをクリックして開きます。
- [Edit (編集)] をクリックします。
- 名前、説明、または手順を変更します。
- 保存をクリックしてスキルを更新します。

スキルのアップロード

zip ファイルとしてアップロードされたスキルには、SKILL.md ファイルに加えて、参考資料やアセットなどの追加リソースが含まれています。

スキル構造:

```
my-skill.zip
### SKILL.md              # Required: main skill instructions
### references/          # Optional: reference documentation
#   ### architecture.md
#   ### troubleshooting.md
### assets/              # Optional: images, diagrams, data files
    ### topology.png
    ### metrics.csv
```

SKILL.md フロントマター要件:

zip ファイルとしてアップロードされるスキルには、 nameフィールドと descriptionフィールドを含む SKILL.md のフロントマターが含まれている必要があります。 AWS DevOps Agent はこれらのフィールドを使用して、スキルを有効にするタイミングを決定します。効果的なフロントマターの記述の詳細については、このトピックの前半の「Frontmatter」セクションを参照してください。

```
---
name: rds-performance-analysis
description: Comprehensive RDS performance investigation procedures
  for connection exhaustion, slow queries, and storage capacity issues.
  Use when investigating database latency or read/write degradation.
---


# RDS Performance Analysis


[Your skill instructions here...]
```

zip アップロードでスキルを作成するには:

- 上記の構造に従って、スキルファイルを含むディレクトリを作成します。
- SKILL.md に適切なフロントマター (名前と説明) が含まれていることを確認します。
- ディレクトリを .zip ファイルに圧縮します。

- エージェントスペースオペレーターウェブアプリのスキルページに移動します。
- 「スキルの追加」をクリックします。
- モーダルから「スキルのアップロード」を選択します。
- .zip ファイルをドラッグアンドドロップするか、クリックして参照します (ZIP ファイルのみ、最大 6 MB)。
- このスキルを使用できるエージェントタイプを 1 つ以上選択します (汎用 はデフォルトで選択され、すべてのエージェントタイプに適用されます。特にオンデマンド、インシデントトリアージ、インシデント RCA、インシデント緩和、または評価をターゲットにするには選択を解除します)。
- zip ファイルの要件と検証結果を確認します。
- 「アップロード」をクリックして、エージェントスペースにスキルを追加します。

zip ファイルとしてアップロードされるスキルに関する重要な制限:

- スクリプトは現在サポートされていません – `scripts/` ディレクトリにスクリプトを含むスキルは、アップロード中に拒否されます。スクリプト実行は、エージェントが安全なコーディング環境にアクセスすると、将来のリリースで有効になります。
- サイズ制限 – zip ファイルの合計サイズは 6 MB を超えることはできません (すべてのファイルを含む)。
- SKILL.md 必須 – zip ファイルには、有効なフロントマターを含む SKILL.md ファイルが含まれている必要があります。

命名スキルのベストプラクティス:

一般的な名前ではなく、rds-throttling-investigation」のような明確でわかりやすい名前を使用します。優れたスキル名は、それが対処する特定のシナリオやサービスを反映しているため、適切なスキルを簡単に一目で特定できます。

スキルの管理

AWS DevOps エージェントは、オペレーターウェブアプリを通じて包括的なスキル管理機能を提供します。

スキルの一覧表示 – エージェントスペース内のすべてのスキルを表示します。スキルページには、スキル名、アクティブまたは非アクティブのステータス、作成日、最終更新日、使用可能なアクションが表示されます。

スキルの表示 – 任意のスキルをクリックすると、その詳細ビューが表示されます。UI で作成されたスキルには編集可能なコンテンツが表示され、UI で名前、説明、または手順を直接変更し、「保存」をクリックして更新できます。zip ファイルとしてアップロードされたスキルには、SKILL.md と references/ や assets/ などの追加のディレクトリを示すファイルツリーが表示されます。ツリー内のファイルをクリックして、読み取り専用モードでその内容を表示します。

スキルのエージェントの選択 – 各スキルを作成または編集するときに使用できるエージェントタイプを設定します。エージェントタイプドロップダウンで、汎用 (デフォルト — すべてのエージェントタイプに適用) 、オンデマンド (会話型クエリ) 、インシデントトリアージ (初期インシデント評価) 、インシデント RCA (根本原因分析) 、インシデント軽減 (自動インシデント対応) 、または評価 (事前推奨事項) のチェックボックスを使用して、1 つ以上のエージェントタイプを選択します。汎用 はデフォルトで選択され、スキルがすべてのエージェントタイプで使用できるようにします。特定のエージェントを対象としたスキルは、コンテキストの消費を減らし、エージェントの集中力を向上させます。

スキルのアクティブ化と非アクティブ化 – アクティブ/非アクティブトグルを使用して、スキルを削除せずに一時的に無効にします。スキル詳細ビューを開き、切り替えを「非アクティブ」に切り替えて、エージェントがすべてのコンテンツと設定を保持しながら、新しい調査のためにロードしないようにします。進行中の調査では、スキルが引き続き使用されます。「アクティブ」に戻すと、スキルがすぐに再び利用可能になります。

スキルの更新 – 作成方法に基づいて既存のスキルを変更します。UI で作成されたスキルについては、スキル詳細ビューで「編集」をクリックし、名前、説明、または手順を変更し、「保存」をクリックして更新します。zip ファイルとしてアップロードされたスキルについては、ファイルをローカルで変更し、新しい zip ファイルを作成して、新しいバージョンをアップロードします。

スキルの削除 – エージェントスペースからスキルを完全に削除します。スキルリストビューを開き、その他のオプションメニュー (⋮) をクリックして「削除」を選択し、永久削除に関する警告を確認し、確認するスキル名を入力し、「スキルの削除」をクリックします。削除を元に戻すことはできません。削除されたスキルをロードしようとすると、進行中の調査が影響を受ける可能性があります。zip ファイルとしてアップロードされたスキルについては、バックアップとして を削除する前に zip ファイルをダウンロードしてください。再度必要になる場合は、削除せずにスキルを無効にすることを検討してください。

ランブックからの移行

既存のランブックは、顧客のアクションなしでスキルに自動的に移行されます。エージェントスペースがスキルモデルに移行すると、すべてのランブックがスキルに変換され、スキル UI に表示されます。移行後、次のことができます。

- 移行されたスキルを確認する – 自動移行がランブックを正しく変換したことを確認します。
- 必要に応じて更新する – UI でスキルを直接編集して、指示を絞り込んだり、説明を更新したり、エージェントタイプのターゲティングを設定したりできます。
- リファレンスで拡張する – 追加のリファレンスマテリアルやアーキテクチャ図から恩恵を受けるスキルについては、リファレンス/ または assets/ ディレクトリを使用して zip アップロードスキルとして再作成します。
- 新しいスキルを作成する – Runbooks で以前にカバーされていない調査ワークフローに新しいスキルを追加します。

自動的に移行されたスキルに問題がある場合、または移行後の更新に関するサポートが必要な場合は、 AWS サポートにお問い合わせください。

学習したスキル

学習スキルとは

学習スキルは、DevOps エージェントがエージェントスペースデータから生成する構造化ナレッジファイルです。学習した各スキルは、 AWS DevOps エージェントがタスクを実行するときに使用する特定のタイプの知識をエンコードします。起動時に、エージェントスペースの理解とツールの使用のベストプラクティスの 2 つの学習スキルを利用できます。

エージェントスペースの理解

エージェントスペース理解スキル (`understanding-agent-space`) は、接続されたクラウドアカウント、コードリポジトリ、テレメトリ統合を分析して、エージェントスペース内のリソースと関係のマップを構築します。

このスキルは、メイン`SKILL.md`ファイルと一連のリファレンスファイルを生成します。メインファイルには、主要なドメインの概念、デプロイ環境 (AWS アカウントとリージョンのペア、Azure サブスクリプションとリージョンなど) 、論理サービスの接続方法を示すコンテナレベルのアーキテクチャ図、アプリケーションの中心となるリクエストパスとトラバースするコンポーネント、およびコードリポジトリのコンテナへのマッピングを含むプレーン言語システムの概要が含まれています。

各論理コンテナは、リソースタイプと ARNs、テーブル名、キュー URLs などの物理識別子を使用して、内部コンポーネント (コンピューティング、データ、メッセージング、ネットワークなど) を説明する専用のリファレンスファイルを受け取ります。リファレンスファイルは、各コンポーネント

にリンクされたアラーム、ダッシュボード、モニターなど、オブザーバビリティカバレッジもキャプチャします。また、各コンポーネントを関連するコードリポジトリ、パッケージ、infrastructure-as-code 定義にマッピングし、ソースコードからデプロイされたリソースまでの完全なトレーサビリティチェーンを提供します。

各重要なリクエストパスは、エントリポイントから各中間サービス、データストア、外部依存関係まで、コンポーネントの詳細度でend-to-endのリクエストフローを説明する専用リファレンスファイルを受け取ります。ファイルには、コンポーネント間のオペレーションとインタラクションメカニズムの順序と、各参加者の責任を示すシーケンスフロー図が含まれています。また、各ホップのロググループパターン、アラーム名とディメンションを含む主要なメトリクス (レイテンシー、エラー率、スロットリング、トークンクォータ) 、サービスやアカウント間で相関できる分散トレーススパンなど、パスに関連するオブザーバビリティシグナルをカタログ化します。

ツールのベストプラクティス

Tool Use Best Practices スキルは、 が使用する過去の調査ツールを分析して、効果的な使用パターン、一般的な障害モード、パラメータガイダンスを抽出します。これにより、DevOps エージェントは既知の落とし穴を回避し、無駄なステップを減らして調査を実行できます。このスキルは、メインファイルとツールごとのリファレンスファイルのセットを生成します。メインファイルは、各ツールがサポートする調査シナリオと対応するリファレンスファイルへのリンクを一覧表示するルーティングインデックスとして機能します。

ツールごとの各リファレンスファイルには、最大 3 つのセクションを含めることができます。

- ベストプラクティス — CloudWatch Logs Insights クエリテンプレート、環境固有のメトリクス名前空間とディメンション、CloudTrail イベントソースフィルターなど、ツールの使用の成功から抽出された調査主導の手法。各エントリは調査シナリオを中心に編成されており、過去の調査で観測された具体的なパラメータ値と例が含まれています。
- 一般的なエラー — 繰り返し発生する障害モードとその修正。各エントリは、アクセスできないアカウントのクエリや不正な形式の集計クエリの作成など、特定のエラー状態を記述し、エージェントが調査ステップを無駄にすることなくエラーを回避または回復できるように修正アクションを提供します。
- 出力管理 — 大きなレスポンスを返す傾向があるツール呼び出しのガイダンス。各エントリは、診断値を維持しながら出力サイズを小さくするパラメータ変更または処理戦略を記述します。

ライブインフラストラクチャアクセスが利用可能な場合、スキルはパターンを含める前に環境に対してパターンを検証します。確認済みのパターンは自信を持って記述され、未確認のパターンは注意深

い言語を使用し、未確認のパターンは除外されます。これにより、スキルはインフラストラクチャの現在の状態と一致します。

学習したスキルの管理

更新 — DevOps エージェントは、エージェントスペースのアクティビティに基づいて、学習したスキルを自動的に生成および更新します。各スキルが更新されるタイミングを以下に示します。

DevOps エージェントは、30 回の調査ごとに更新されたツール使用のベストプラクティススキルを生成します。

エージェントスペース理解スキルは、エージェントスペース機能または統合を追加、更新、または削除するたびに実行される学習エージェントによって生成されます。

学習したスキルを手動で再生成するには、オペレーターアプリのトポロジページの再生成ボタンを選択するか、エージェントとチャットして、学習したスキルを更新するように依頼します。

非アクティブ化 — 学習したスキルはデフォルトでアクティブです。アクティブの場合、DevOps エージェントは各 DevOps エージェントタスクの開始時にロードします。学習したスキルの適用を停止するには、オペレーターアプリのスキルビューワーから非アクティブ化します。スキルを非アクティブ化しても、スキルは削除されません。スキルは保持され、いつでも再アクティブ化できます。スキルが非アクティブ化されると、DevOps エージェントはそのスキルを認識せずに動作します。

トポロジビュー — エージェントスペースのウェブアプリのトポロジページは、エージェントスペース理解スキルを使用して、エージェントスペース環境を論理コンテナおよびコンポーネントとして視覚的に表示します。コンテナをクリックすると、そのコンポーネント、リソース識別子、テレメトリが表示されます。

サポートされるリージョン

このトピックでは、 AWS DevOps エージェントを使用できる AWS リージョンについて説明します。 AWS リージョンの詳細については、「 アカウント管理リファレンスガイド」の「アカウントで使用できる AWS リージョンを指定する」を参照してください。 AWS

クロスリージョンリソースのモニタリング

AWS DevOps エージェントは、エージェントスペースを作成するサポートされている AWS リージョンに関係なく、任意のリージョンにある AWS アカウントのリソースをモニタリングおよび調査できます。 AWS アカウントをエージェントスペースに関連付けると、エージェントはそのアカウン

ト内のすべてのリージョンのリソースを検出してマッピングします。つまり、ワークロードが実行されるすべてのリージョンでエージェントスペースは必要ありません。

希望するデータレジデンシー、運用チームへの近接性、または組織の要件に基づいて、サポートされているリージョンを選択します。

サポート対象のリージョン

AWS DevOps エージェントは、次の AWS リージョンで使用できます。

リージョン名	リージョンコード	コンソールリンク
米国東部 (バージニア北部)	us–east–1	コンソールを開く
米国西部 (オレゴン)	us-west-2	コンソールを開く
アジアパシフィック (シドニー)	ap-southeast-2	コンソールを開く
アジアパシフィック (東京)	ap-northeast-1	コンソールを開く
欧州 (フランクフルト)	eu-central-1	コンソールを開く
欧州 (アイルランド)	eu-west-1	コンソールを開く

サービスエンドポイント

リージョン名	リージョンコード	Endpoint	プロトコル
米国東部 (バージニア北部)	us–east–1	aidevops.us-east-1.amazonaws.com	HTTPS
米国西部 (オレゴン)	us-west-2	aidevops.us-west-2.amazonaws.com	HTTPS
アジアパシフィック (シドニー)	ap-southeast-2	aidevops.ap-southeast-2.amazonaws.com	HTTPS

リージョン名	リージョンコード	Endpoint	プロトコル
アジアパシフィック (東京)	ap-northeast-1	aidevops.ap-northeast-1.amazonaws.com	HTTPS
欧州 (フランクフルト)	eu-central-1	aidevops.eu-central-1.amazonaws.com	HTTPS
欧州 (アイルランド)	eu-west-1	aidevops.eu-west-1.amazonaws.com	HTTPS

考慮事項

- エージェントスペースリージョンの選択 — エージェントスペースとそのデータ (調査、

トポロジ、レコメンデーション) は、作成したリージョンに保存されます。データレジデンシー要件を満たすリージョンを選択します。

- クロスリージョンモニタリング — エージェントに関連付けられた AWS アカウントのリソース

これらのリソースがデプロイされているリージョンに関係なく、スペースがモニタリングされます。ワークロードを実行する各リージョンに個別のエージェントスペースを作成する必要はありません。

- サードパーティー統合 — CI/CD プロバイダーへの接続 (GitHub、GitLab)

オブザーバビリティツール (Dynatrace、Datadog、New Relic、Splunk)、および MCP サーバーは、エージェントスペースごとに設定され、リージョンに依存しません。

AWS DevOps エージェントの開始方法

この入門ガイドでは、基本的なエージェントスペースを作成し、最小限のアクセス許可を設定し、最初の AI を活用した調査を行います。

トピック:

エージェントスペースの作成

エージェントスペースは、 AWS DevOps エージェントがアクセスできるツールとインフラストラクチャを定義します。このガイドでは、エージェントスペースの作成、プライマリアカウントアクセスの設定、DevOps エージェントウェブアプリの有効化について説明します。エージェントスペースの概念の詳細については、「エージェントスペースとは」を参照してください。

エージェントスペースの作成

AWS DevOps エージェントコンソールにアクセスする

1. AWS マネジメントコンソールにサインインする
2. AWS DevOps エージェントコンソールに移動する

エージェントスペースに名前を付ける

1. エージェントスペースの作成をクリックします。

エージェントスペースの詳細セクションで、以下を指定します。

1. 名前 フィールドに、エージェントスペースの名前を入力します。

2. (オプション) 説明フィールドに、エージェントスペースの目的に関する詳細を追加します。
3. (オプション) エージェントのレスポンス言語ドロップダウンから、レスポンス、検出結果、調査出力を生成するときにエージェントが使用する言語を選択します。オプションには、インドネシア語、中国語 (簡体字/PRC)、中国語 (繁体字/台湾) 、英語 (英国) 、フランス語 (フランス) 、ドイツ語 (ドイツ) 、イタリア語 (イタリア) 、日本語 (日本) 、韓国語 (韓国) 、ポルトガル語 (ブラジル) 、スペイン語 (南米) 、トルコ語 (トルコ) 、アラビア語 (サウジアラビア) 、タイ語 (タイ) 、ベトナム語 (ベトナム) などがあります。言語が選択されていない場合、エージェントは入力の言語で応答します。

プライマリアカウントアクセスの設定

このエージェントスペース AWS リソースアクセスを許可するセクションでは、プライマリ AWS アカウントへのアクセスをエージェントスペースに許可する IAM ロールを設定します。プライマリアカウントは、エージェントスペースを作成する AWS アカウントです。 AWS DevOps エージェントは、調査中にこのアカウントの AWS リソースを検出してアクセスするために IAM ロールを必要とします。

ロール設定方法を選択します。 以下のオプションのいずれかを選択します。

オプション 1: 新しい AWS DevOps エージェントロールを自動作成する (推奨)

このオプションは、 AWS DevOps Agent がアカウントのリソースを調査するための適切なアクセス許可を持つロールを自動的に作成します。

> **Note**
>
> このオプションを使用するには、新しいロールを作成するための IAM アクセス許可が必要です。

1. 新しい AWS DevOps エージェントロールの自動作成 を選択する
2. (オプション) 作成するエージェントスペースロール名を更新する

オプション 2: 既存のロールを割り当てる

このオプションは、別の管理者が以前に AWS DevOps エージェント専用のロールを作成している場合に使用します。

1. 選択 既存のロールを割り当てる

2. ドロップダウンメニューから、適切なアクセス許可を持つ既存のロールを選択します。

オプション 3: ポリシーテンプレートを使用して新しい AWS DevOps エージェントロールを作成する

このオプションは、エージェントがプライマリアカウントでアクセスできるサービスとリソースを制限する必要がある場合に使用します。

1. ポリシーテンプレートを使用して新しい AWS DevOps エージェントロールを作成する を選択します。
2. 手順に従って、新しいロールの信頼ポリシーとインラインポリシーを作成します。

エージェントスペースウェブアプリの有効化

ウェブアプリは、担当者がインシデント調査とレコメンデーションの確認のために AWS DevOps エージェントとやり取りする場所です。詳細については、 AWS DevOps エージェントコンソールアーキテクチャ〔リンク〕」を参照してください。有効にすると、ユーザーは AWS マネジメントコンソールから IAM 認証リンクを介して エージェントスペースウェブアプリにアクセスできます。

以下のオプションのいずれかを選択します。

オプション 1: 新しい AWS DevOps エージェントロールを自動作成する (推奨)

このオプションは、DevOps エージェントウェブアプリにアクセスするための適切なアクセス許可を持つロールを自動的に作成します。

> **Note**
>
> このオプションを使用するには、新しいロールを作成するための IAM アクセス許可が必要です。

1. 新しい AWS DevOps エージェントロールの自動作成 を選択する
2. ロールに付与されるアクセス許可を確認する

オプション 2: 既存のロールを割り当てる

このオプションは、別の管理者が以前にオペレーターロールを作成している場合に使用します。

1. 選択 既存のロールを割り当てる
2. ドロップダウンメニューから、適切なアクセス許可を持つ既存のロールを選択します。

オプション 3: ポリシーテンプレートを使用して新しい AWS DevOps エージェントロールを作成する

ウェブアプリへのアクセス許可をカスタマイズする必要がある場合は、このオプションを使用します。

1. ポリシーテンプレートを使用して新しい AWS DevOps エージェントロールを作成する を選択します。
2. 手順に従って、新しいロールの信頼ポリシーとインラインポリシーを作成します。

タグの追加 (オプション)

作成時にエージェントスペースに AWS タグを追加できます。タグは、リソースの整理と識別に役立つキーと値のペアです。エージェントスペースごとに最大 50 個のタグを追加できます。タグを追加するには、「エージェントスペースの作成」ページの「タグ」セクションを展開し、「新しいタグの追加」をクリックします。

エージェントスペースの作成を完了する

すべてのセクションに入力したら、作成 をクリックします。

エージェントスペースの設定の確認

設定すると、オペレータアクセスボタンがエージェントスペースの詳細ページに表示されます。クリックすると、ウェブアプリが新しいタブで開き、正常に認証されます。

次の手順

エージェントスペースを設定したら、次のステップを検討してください。

- アプリケーションが複数のアカウントにまたがる場合のセカンダリ AWS アカウントの追加
- オブザーバビリティツールやチケットシステムなどのサードパーティー統合を設定する
- 本番環境の AWS Identity Center 認証を設定する
- AWS DevOps Agent がインフラストラクチャを理解するのに役立つアプリケーションリソースマッピングを調べる

AWS DevOps エージェント CLI オンボーディングガイド

概要:

AWS DevOps エージェントを使用すると、 AWS インフラストラクチャをモニタリングおよび管理できます。このガイドでは、 コマンドラインインターフェイス (AWS CLI) AWS を使用して AWS DevOps エージェントを設定する方法について説明します。IAM ロールを作成し、エージェントスペースをセットアップして、 AWS アカウントを関連付けます。また、オペレーターアプリを有効にし、オプションでサードパーティーの統合を接続します。このガイドの所要時間は約 20 分です。

AWS DevOps エージェントは、米国東部 (バージニア北部) 、米国西部 (オレゴン) 、アジアパシフィック (シドニー) 、アジアパシフィック (東京) 、欧州 (フランクフルト) 、欧州 (アイルランド) の 6 つの AWS リージョンで利用できます。サポートされているリージョンの詳細については、「」を参照してくださいthe section called “サポートされるリージョン”。

前提条件

作業を開始する前に、以下の準備が整っていることを確認します。

- AWS CLI バージョン 2 のインストールと設定
- AWS モニタリングアカウントへの認証
- AWS Identity and Access Management (IAM) ロールを作成し、ポリシーをアタッチするアクセス許可
- モニタリング AWS アカウントとして使用するアカウント
- CLI および JSON AWS 構文に精通していること

このガイド全体で、次のプレースホルダー値を独自のプレースホルダー値に置き換えます。

- `<MONITORING_ACCOUNT_ID>` — モニタリング (プライマリ) AWS アカウントの 12 桁のアカウント ID
- `<EXTERNAL_ACCOUNT_ID>` — モニタリングするセカンダリ AWS アカウントの 12 桁のアカウント ID (ステップ 4 で使用)
- `<REGION>` — エージェントスペースの AWS リージョンコード (例: `us-east-1`または `eu-central-1`)
- `<AGENT_SPACE_ID>` — `create-agent-space` コマンドによって返されるエージェントスペース識別子

IAM ロールの設定

1. DevOps エージェントスペースロールを作成する

次のコマンドを実行して、IAM 信頼ポリシーを作成します。

```
cat > devops-agentspace-trust-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "aidevops.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<MONITORING_ACCOUNT_ID>"
        },
        "ArnLike": {
          "aws:SourceArn":
 "arn:aws:aidevops:<REGION>:<MONITORING_ACCOUNT_ID>:agentspace/*"
        }
      }
    }
  ]
}
EOF
```

IAM ロールを作成します。

```
aws iam create-role \
  --region <REGION> \
  --role-name DevOpsAgentRole-AgentSpace \
  --assume-role-policy-document file://devops-agentspace-trust-policy.json
```

次のコマンドを実行して、ロール ARN を保存します。

```
aws iam get-role --role-name DevOpsAgentRole-AgentSpace --query 'Role.Arn' --output
 text
```

AWS 管理ポリシーをアタッチします。

```
aws iam attach-role-policy \
  --role-name DevOpsAgentRole-AgentSpace \
  --policy-arn arn:aws:iam::aws:policy/AIDevOpsAgentAccessPolicy
```

Resource Explorer サービスにリンクされたロールの作成を許可するインラインポリシーを作成してアタッチします。

```
cat > devops-agentspace-additional-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCreateServiceLinkedRoles",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws:iam::<MONITORING_ACCOUNT_ID>:role/aws-service-role/resource-
explorer-2.amazonaws.com/AWSServiceRoleForResourceExplorer"
      ]
    }
  ]
}
EOF

aws iam put-role-policy \
  --role-name DevOpsAgentRole-AgentSpace \
  --policy-name AllowCreateServiceLinkedRoles \
  --policy-document file://devops-agentspace-additional-policy.json
```

2. オペレーターアプリの IAM ロールを作成する

次のコマンドを実行して、IAM 信頼ポリシーを作成します。

```
cat > devops-operator-trust-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Principal": {
        "Service": "aidevops.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<MONITORING_ACCOUNT_ID>"
        },
        "ArnLike": {
          "aws:SourceArn":
 "arn:aws:aidevops:<REGION>:<MONITORING_ACCOUNT_ID>:agentspace/*"
        }
      }
    }
  ]
}
EOF
```

IAM ロールを作成します。

```
aws iam create-role \
  --role-name DevOpsAgentRole-WebappAdmin \
  --assume-role-policy-document file://devops-operator-trust-policy.json \
  --region <REGION>
```

次のコマンドを実行して、ロール ARN を保存します。

```
aws iam get-role --role-name DevOpsAgentRole-WebappAdmin --query 'Role.Arn' --output
 text
```

AWS マネージドオペレーターアプリポリシーをアタッチします。

```
aws iam attach-role-policy \
  --role-name DevOpsAgentRole-WebappAdmin \
  --policy-arn arn:aws:iam::aws:policy/AIDevOpsOperatorAppAccessPolicy
```

この管理ポリシーは、エージェントスペース機能にアクセスするためのアクセス許可をオペレーターアプリに付与します。これらの機能には、調査、レコメンデーション、ナレッジ管理、チャット、

AWS サポート統合が含まれます。ポリシーは、 aws:PrincipalTag/AgentSpaceId条件を使用して特定のエージェントスペースへのアクセスをスコープします。アクションの完全なリストの詳細については、「」を参照してくださいthe section called “DevOps エージェント IAM アクセス許可”。

オンボーディング手順

1. エージェントスペースを作成する

次のコマンドを実行して、エージェントスペースを作成します。

```
aws devops-agent create-agent-space \
  --name "MyAgentSpace" \
  --description "AgentSpace for monitoring my application" \
  --region <REGION>
```

必要に応じて、カスタマーマネージド KMS AWS キー--kms-key-arnを暗号化に使用するようにを指定します。--tags を使用してリソースタグを追加し--locale、エージェントレスポンスの言語を設定することもできます。

レスポンス (にありますagentSpace.agentSpaceId) agentSpaceIdから を保存します。

後でエージェントスペースを一覧表示するには、次のコマンドを実行します。

```
aws devops-agent list-agent-spaces \
  --region <REGION>
```

2. AWS アカウントを関連付ける

AWS アカウントを関連付けてトポロジ検出を有効にします。accountType を次のいずれかの値に設定します。

- monitor — エージェントスペースが存在するプライマリアカウント。このアカウントはエージェントをホストし、トポロジ検出に使用されます。
- source — エージェントがモニタリングする追加のアカウント。ステップ 4 で外部アカウントを関連付ける場合は、このタイプを使用します。

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id aws \
```

```
  --configuration '{
    "aws": {
      "assumableRoleArn": "arn:aws:iam::<MONITORING_ACCOUNT_ID>:role/DevOpsAgentRole-
AgentSpace",
      "accountId": "<MONITORING_ACCOUNT_ID>",
      "accountType": "monitor"
    }
  }' \
  --region <REGION>
```

3. オペレーターアプリを有効にする

認証フローでは、IAM、IAM アイデンティティセンター (IDC)、または外部 ID プロバイダー (IdP) を使用できます。次のコマンドを実行して、エージェントスペースのオペレータアプリを有効にします。

```
aws devops-agent enable-operator-app \
  --agent-space-id <AGENT_SPACE_ID> \
  --auth-flow iam \
  --operator-app-role-arn "arn:aws:iam::<MONITORING_ACCOUNT_ID>:role/DevOpsAgentRole-
WebappAdmin" \
  --region <REGION>
```

IAM Identity Center 認証の場合は、 を使用して `--auth-flow idc` を指定します`--idc-instance-arn`。外部 ID プロバイダーの場合は、 を使用し`--issuer-url`、、`--idp-client-id`、および `--auth-flow idp`を指定します`--idp-client-secret`。詳細については、「the section called “IAM アイデンティティセンター認証の設定”」および「the section called “外部 ID プロバイダー (IdP) 認証の設定”」を参照してください。

> 注: アカウント内の別のエージェントスペースのオペレーターアプリロールを以前に作成している場合は、そのロール ARN を再利用できます。

4. (オプション) 追加のソースアカウントを関連付ける

AWS DevOps Agent で追加のアカウントをモニタリングするには、IAM クロスアカウントロールを作成します。

外部アカウントでクロスアカウントロールを作成する

外部アカウントに切り替え、信頼ポリシーを作成します。`MONITORING_ACCOUNT_ID` は、ステップ 2 で設定したエージェントスペースをホストするメインアカウントです。この設定により、 AWS

DevOps エージェントサービスは、モニタリングアカウントに代わってセカンダリソースアカウントのロールを引き受けることができます。

次のコマンドを実行して、信頼ポリシーを作成します。

```
cat > devops-cross-account-trust-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "aidevops.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<MONITORING_ACCOUNT_ID>",
          "sts:ExternalId":
 "arn:aws:aidevops:<REGION>:<MONITORING_ACCOUNT_ID>:agentspace/<AGENT_SPACE_ID>"
        }
      }
    }
  ]
}
EOF
```

クロスアカウント IAM ロールを作成します。

```
aws iam create-role \
  --role-name DevOpsAgentCrossAccountRole \
  --assume-role-policy-document file://devops-cross-account-trust-policy.json
```

次のコマンドを実行して、ロール ARN を保存します。

```
aws iam get-role --role-name DevOpsAgentCrossAccountRole --query 'Role.Arn' --output
 text
```

AWS 管理ポリシーをアタッチします。

```
aws iam attach-role-policy \
  --role-name DevOpsAgentCrossAccountRole \
```

```
  --policy-arn arn:aws:iam::aws:policy/AIDevOpsAgentAccessPolicy
```

インラインポリシーをアタッチして、外部アカウントで Resource Explorer サービスにリンクされたロールを作成できるようにします。

```
cat > devops-cross-account-additional-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCreateServiceLinkedRoles",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws:iam::<EXTERNAL_ACCOUNT_ID>:role/aws-service-role/resource-
explorer-2.amazonaws.com/AWSServiceRoleForResourceExplorer"
      ]
    }
  ]
}
EOF

aws iam put-role-policy \
  --role-name DevOpsAgentCrossAccountRole \
  --policy-name AllowCreateServiceLinkedRoles \
  --policy-document file://devops-cross-account-additional-policy.json
```

外部アカウントの関連付け

モニタリングアカウントに切り替え、次のコマンドを実行して外部アカウントを関連付けます。

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id aws \
  --configuration '{
    "sourceAws": {
      "accountId": "<EXTERNAL_ACCOUNT_ID>",
      "accountType": "source",
      "assumableRoleArn": "arn:aws:iam::<EXTERNAL_ACCOUNT_ID>:role/
DevOpsAgentCrossAccountRole"
    }
```

```
  }' \
  --region <REGION>
```

5. (オプション) GitHub を関連付ける

> 注: CLI を介して関連付ける前に、まず OAuth フローを使用して AWS DevOps エージェントコンソールから GitHub を登録する必要があります。

コンソールを使用して GitHub を登録する手順については、「」を参照してくださいthe section called “CI/CD パイプラインへの接続”。

登録されたサービスを一覧表示します。

```
aws devops-agent list-services \
  --region <REGION>
```

serviceType <SERVICE_ID>の を保存します: github。

コンソールで GitHub を登録したら、次のコマンドを実行して GitHub リポジトリを関連付けます。

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id <SERVICE_ID> \
  --configuration '{
    "github": {
      "repoName": "<GITHUB_REPO_NAME>",
      "repoId": "<GITHUB_REPO_ID>",
      "owner": "<GITHUB_OWNER>",
      "ownerType": "organization"
    }
  }' \
  --region <REGION>
```

6. (オプション) ServiceNow を登録して関連付ける

まず、ServiceNow サービスを OAuth 認証情報に登録します。

```
aws devops-agent register-service \
  --service servicenow \
  --service-details  '{
    "servicenow": {
      "instanceUrl": "<SERVICENOW_INSTANCE_URL>",
```

```
      "authorizationConfig": {
        "oAuthClientCredentials": {
            "clientName": "<SERVICENOW_CLIENT_NAME>",
            "clientId": "<SERVICENOW_CLIENT_ID>",
            "clientSecret": "<SERVICENOW_CLIENT_SECRET>"
        }
      }
    }
  }' \
  --region <REGION>
```

返された を保存し<SERVICE_ID>、ServiceNow を関連付けます。

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id <SERVICE_ID> \
  --configuration '{
    "servicenow": {
      "instanceUrl": "<SERVICENOW_INSTANCE_URL>"
    }
  }' \
  --region <REGION>
```

7. (オプション) Dynatrace を登録して関連付ける

まず、Dynatrace サービスを OAuth 認証情報に登録します。

```
aws devops-agent register-service \
  --service dynatrace \
  --service-details '{
  "dynatrace": {
    "accountUrn": "<DYNATRACE_ACCOUNT_URN>",
    "authorizationConfig": {
        "oAuthClientCredentials": {
            "clientName": "<DYNATRACE_CLIENT_NAME>",
            "clientId": "<DYNATRACE_CLIENT_ID>",
            "clientSecret": "<DYNATRACE_CLIENT_SECRET>"
        }
      }
    }
  }' \
  --region <REGION>
```

返された を保存し<SERVICE_ID>、Dynatrace を関連付けます。リソースはオプションです。環境は、関連付ける Dynatrace 環境を指定します。

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id <SERVICE_ID> \
  --configuration '{
    "dynatrace": {
      "envId": "<DYNATRACE_ENVIRONMENT_ID>",
      "resources": [
        "<DYNATRACE_RESOURCE_1>",
        "<DYNATRACE_RESOURCE_2>"
      ]
    }
  }' \
  --region <REGION>
```

レスポンスには、統合用のウェブフック情報が含まれます。このウェブフックを使用して、Dynatrace から調査をトリガーできます。詳細については、「the section called “Dynatrace の接続”」を参照してください。

8. (オプション) Splunk を登録して関連付ける

まず、BearerToken 認証情報を使用して Splunk サービスを登録します。

エンドポイントは次の形式を使用します。 `https://<XXX>.api.scs.splunk.com/<XXX>/mcp/v1/`

```
aws devops-agent register-service \
  --service mcpserversplunk \
  --service-details '{
  "mcpserversplunk": {
    "name": "<SPLUNK_NAME>",
    "endpoint": "<SPLUNK_ENDPOINT>",
    "authorizationConfig": {
        "bearerToken": {
            "tokenName": "<SPLUNK_TOKEN_NAME>",
            "tokenValue": "<SPLUNK_TOKEN_VALUE>"
        }
      }
    }
  }' \
```

```
  --region <REGION>
```

返された を保存し<SERVICE_ID>、Splunk を関連付けます。

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id <SERVICE_ID> \
  --configuration '{
    "mcpserversplunk":  {
      "name": "<SPLUNK_NAME>",
      "endpoint": "<SPLUNK_ENDPOINT>"
    }
  }' \
  --region <REGION>
```

レスポンスには、統合用のウェブフック情報が含まれます。このウェブフックを使用して、Splunk から調査をトリガーできます。詳細については、「the section called “Splunk の接続”」を参照してください。

9. (オプション) New Relic を登録して関連付ける

まず、API キー認証情報を使用して New Relic サービスを登録します。

リージョン: USまたは EU。

オプションフィールド: applicationIds、entityGuids、 alertPolicyIds

```
aws devops-agent register-service \
  --service mcpservernewrelic \
  --service-details '{
    "mcpservernewrelic": {
      "authorizationConfig": {
        "apiKey": {
          "apiKey": "<YOUR_NEW_RELIC_API_KEY>",
          "accountId": "<YOUR_ACCOUNT_ID>",
          "region": "US",
          "applicationIds": ["<APP_ID_1>", "<APP_ID_2>"],
          "entityGuids": ["<ENTITY_GUID_1>"],
          "alertPolicyIds": ["<POLICY_ID_1>"]
        }
      }
    }
  }' \
```

```
  --region <REGION>
```

返された を保存し<SERVICE_ID>、New Relic を関連付けます。

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id <SERVICE_ID> \
  --configuration '{
    "mcpservernewrelic":  {
      "accountId": "<YOUR_ACCOUNT_ID>",
      "endpoint": "https://mcp.newrelic.com/mcp/"
    }
  }' \
  --region <REGION>
```

レスポンスには、統合用のウェブフック情報が含まれます。このウェブフックを使用して、New Relic から調査をトリガーできます。詳細については、「the section called “New Relic の接続”」を参照してください。

10. (オプション) Datadog を登録して関連付ける

CLI を介して関連付ける前に、まず OAuth フローを使用して AWS DevOps エージェントコンソールから Datadog を登録する必要があります。詳細については、「the section called “DataDog の接続”」を参照してください。

登録されたサービスを一覧表示します。

```
aws devops-agent list-services \
  --region <REGION>
```

serviceType <SERVICE_ID>の を保存します: mcpserverdatadog。

次に、Datadog を関連付けます。

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id <SERVICE_ID> \
  --configuration '{
    "mcpserverdatadog": {
      "name": "Datadog-MCP-Server",
      "endpoint": "<DATADOG_MCP_ENDPOINT>"
    }
```

```
  }' \
  --region <REGION>
```

レスポンスには、統合用のウェブフック情報が含まれます。このウェブフックを使用して、Datadog から調査をトリガーできます。詳細については、「the section called “DataDog の接続”」を参照してください。

11. (オプション) エージェントスペースを削除する

エージェントスペースを削除すると、そのエージェントスペースのすべての関連付け、設定、調査データが削除されます。このアクションは元に戻すことができません。

エージェントスペースを削除するには、次のコマンドを実行します。

```
aws devops-agent delete-agent-space \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>
```

検証

セットアップを確認するには、次のコマンドを実行します。

```
# List your agent spaces
aws devops-agent list-agent-spaces \
  --region <REGION>

# Get details of a specific agent space
aws devops-agent get-agent-space \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>

# List associations for an agent space
aws devops-agent list-associations \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>
```

次の手順

- 追加の統合を接続するには、「」を参照してくださいAWS DevOps Agent の機能の設定。
- エージェントのスキルと機能の詳細については、「」を参照してくださいthe section called “DevOps エージェントスキル”。

- オペレーターウェブアプリを理解するには、「」を参照してくださいthe section called “DevOps エージェントウェブアプリとは”。

注意事項

- <AGENT_SPACE_ID>、<MONITORING_ACCOUNT_ID>、、<EXTERNAL_ACCOUNT_ID><REGION>などを実際の値に置き換えます。
- サポートされているリージョンのリストについては「the section called “サポートされるリージョン”」を参照してください。

テスト環境の作成

このガイドでは、サンプルアーキテクチャを使用して AWS DevOps Agent のインシデント対応機能を検証するための実践的なテストを提供します。本番稼働用システムを接続する前に DevOps エージェントをテストする場合は、この補足を使用します。

前提条件

- AWS 管理アクセス権を持つ アカウント
- AWS DevOps エージェントの自動作成ロールフローを使用して作成および設定された DevOps エージェントスペース

コストと安全性の概要

コスト保護

- EC2 テスト: 2 時間無料 (AWS 無料利用枠) または ~0.02 USD
- Lambda テスト: 無料 (1 か月あたりの無料利用枠あたり 100 万リクエスト)
- CloudWatch: 無料 (10 個のアラーム、基本メトリクスを含む)
- 予想される総コスト: テスト完了時に 0.00 ~ 0.05 USD

これらのテストにおける安全機能

- 自動終了: 組み込みの自動シャットダウン
- 無料利用枠対象: 最小のインスタンスタイプを使用

- 制限された範囲: 最小の分離されたテストリソース
- 簡単なクリーンアップ: すべてを削除するためのシンプルなコンソールステップ
- 本番環境への影響なし: テスト環境を完全に分離する

テスト用に AWS アカウントをセットアップする

> **Important**
>
> インフラストラクチャリソースは、DevOps エージェントスペースのプライマリクラウド AWS アカウントを作成したアカウントにデプロイする必要があります。特定のリージョンは関係ありません。

1. AWS コンソールにログインする: https://console.aws.amazon.com
2. DevOps エージェントスペースがあるのと同じ AWS アカウントで作業していることを確認します。
3. テストリソースには任意のリージョンを使用できます

> **Note**
>
> DevOps エージェントのプライマリアカウントと作成するテスト環境リソース間の 1:1 マッピングにより、テストのセットアップが簡素化されます。DevOps エージェントスペースを簡単に拡張してセカンダリアカウントを含めたり、クロスアカウント調査を有効にしたりできます。

テストを選択する

テストは個別に実行することも、両方を同時に実行することもできます。

テストオプション A: EC2 CPU 容量テスト

目的: EC2 パフォーマンスの問題を検出して調査する AWS DevOps Agent の機能を検証する

推定時間: 5 分のセットアップ + 10 分の自動実行

難易度: 完全自動化 (手動ステップは不要)

テストオプション B: Lambda エラー率テスト

目的: Lambda 関数エラーを検出して調査する AWS DevOps エージェントの機能を検証する

推定時間: 10 分のセットアップ + トリガーまで 2 分

難易度: 非常に簡単

テストオプション A: EC2 CPU 容量テスト

ステップ 1: EC2 テスト用に CloudFormation スタックをデプロイする

CloudFormation を使用してテストリソースを作成します。これにより、 AWS DevOps Agent はテストリソースを適切に追跡して調査できます。

1. CloudFormation に移動します。
   a. AWS コンソールでCloudFormation」を検索し、CloudFormation をクリックします。
   b. スタックの作成 > 新しいリソースを使用する (標準) をクリックします。
2. テンプレートのアップロード:
   a. という名前の新しいローカルファイルを作成するAWS-DevOpsAgent-ec2-test.yaml
   b. この CloudFormation テンプレートをコピーしてファイルに貼り付けます。
      i.

```
AWSTemplateFormatVersion: '2010-09-09'
Description: 'AWS DevOps Agent EC2 CPU Test Stack'
Parameters:
  MyIP:
    Type: String
    Description: Your current IP address for SSH access (find at https://
whatismyipaddress.com)
    Default: '0.0.0.0/0'
Resources:
  # Security Group for SSH access
  TestSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: AWS-DevOpsAgent-test-sg
      GroupDescription: AWS DevOps Agent beta testing security group
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          CidrIp: !Ref MyIP
```

```
          Description: SSH access from your IP
      Tags:
        - Key: Name
          Value: AWS-DevOpsAgent-Test-SG
        - Key: Purpose
          Value: AWS-DevOpsAgent-Testing
  # Key Pair for SSH access
  TestKeyPair:
    Type: AWS::EC2::KeyPair
    Properties:
      KeyName: AWS-DevOpsAgent-test-key
      KeyType: rsa
      Tags:
        - Key: Name
          Value: AWS-DevOpsAgent-Test-Key
        - Key: Purpose
          Value: AWS-DevOpsAgent-Testing
  # EC2 Instance for CPU testing
  TestInstance:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: t3.micro
      ImageId: '{{resolve:ssm:/aws/service/ami-amazon-linux-latest/al2023-ami-
kernel-6.1-x86_64}}'
      KeyName: !Ref TestKeyPair
      SecurityGroupIds:
        - !Ref TestSecurityGroup
      UserData:
        Fn::Base64: !Sub |
          #!/bin/bash
          yum update -y
          yum install -y htop

          # Create the CPU stress test script
          cat > /home/ec2-user/cpu-stress-test.sh << 'EOF'
          #!/bin/bash
          echo "Starting AWS DevOpsAgent CPU Stress Test"
          echo "Time: $(date)"
          echo "Instance: $(curl -s http://169.254.169.254/latest/meta-data/
instance-id)"
          echo ""

          # Get number of CPU cores
          CORES=$(nproc)
```

```
          echo "CPU Cores: $CORES"
          echo ""

          echo "Starting stress test (5 minutes)..."
          echo "This will generate >70% CPU usage to trigger CloudWatch alarm"
          echo ""

          # Create CPU load using yes command
          echo "Starting CPU load processes..."
          for i in $(seq 1 $CORES); do
              (yes > /dev/null) &
              CPU_PID=$!
              echo "Started CPU load process $i (PID: $CPU_PID)"
              echo $CPU_PID >> /tmp/cpu_test_pids
          done

          # Auto-cleanup after 5 minutes
          (sleep 300 && echo "Stopping CPU load processes..." && kill $(cat /
tmp/cpu_test_pids 2>/dev/null) 2>/dev/null && rm -f /tmp/cpu_test_pids) &

          echo ""
          echo "CPU load processes started for 5 minutes"
          echo "Check CloudWatch for alarm trigger in 3-5 minutes"
          EOF

          chmod +x /home/ec2-user/cpu-stress-test.sh
          chown ec2-user:ec2-user /home/ec2-user/cpu-stress-test.sh

          # Create auto-shutdown script (safety mechanism)
          cat > /home/ec2-user/auto-shutdown.sh << 'SHUTDOWN_EOF'
          #!/bin/bash
          echo "Auto-shutdown scheduled for 2 hours from now: $(date)"
          sleep 7200
          echo "Auto-shutdown executing at: $(date)"
          sudo shutdown -h now
          SHUTDOWN_EOF

          chmod +x /home/ec2-user/auto-shutdown.sh
          nohup /home/ec2-user/auto-shutdown.sh > /home/ec2-user/auto-
shutdown.log 2>&1 &

          echo "AWS DevOpsAgent test setup completed at $(date)" > /home/ec2-
user/setup-complete.txt
      Tags:
```

```
        - Key: Name
          Value: AWS-DevOpsAgent-Test-Instance
        - Key: Purpose
          Value: AWS-DevOpsAgent-Testing
  # CloudWatch Alarm for CPU utilization
  CPUAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmName: AWS-DevOpsAgent-EC2-CPU-Test
      AlarmDescription: AWS-DevOpsAgent beta test - EC2 CPU utilization alarm
      MetricName: CPUUtilization
      Namespace: AWS/EC2
      Statistic: Average
      Period: 60
      EvaluationPeriods: 1
      Threshold: 70
      ComparisonOperator: GreaterThanThreshold
      Dimensions:
        - Name: InstanceId
          Value: !Ref TestInstance
      TreatMissingData: notBreaching
Outputs:
  InstanceId:
    Description: EC2 Instance ID for testing
    Value: !Ref TestInstance

  SecurityGroupId:
    Description: Security Group ID
    Value: !Ref TestSecurityGroup

  AlarmName:
    Description: CloudWatch Alarm Name
    Value: !Ref CPUAlarm

  SSHCommand:
    Description: SSH command to connect to instance
    Value: !Sub 'ssh -i "AWS-DevOpsAgent-test-key.pem" ec2-user@
${TestInstance.PublicDnsName}'
```

c. CloudFormation コンソールで、テンプレートファイルのアップロードを選択します。

d. ファイルの選択 をクリックします。

e. AWS-DevOpsAgent-ec2-test.yaml ファイルを選択する

f. [次へ] をクリックします。

3. スタックの設定:
   a. スタック名:`AWS-DevOpsAgent-EC2-Test`
   b. パラメータ :
      i. MyIP: デフォルトのままにします `0.0.0.0/0` (必要に応じて後で保護できます)
   c. [次へ] をクリックします。
4. スタックオプションを設定します。
   a. デフォルトのまま、次へ をクリックします。
5. 確認と作成:
   a. AWS CloudFormation が IAM リソースを作成する可能性があることを確認する
   b. 送信 をクリックします。
6. 完了まで待ちます。
   a. スタックの作成には 3～5 分かかります
   b. ステータスは から `CREATE_IN_PROGRESS`に変わります`CREATE_COMPLETE`
   c. 重要: EC2 インスタンスは、 AWS DevOpsAgent が追跡できる CloudFormation スタックの一部になりました。

オプション: 安全な SSH アクセス (インスタンスに接続する場合のみ)

自動テストを実行するだけの場合は、このステップをスキップします。

1. EC2 セキュリティグループに移動します。
   a. AWS コンソールで、EC2 → セキュリティグループに移動します。
   b. 検索`AWS-DevOpsAgent-test-sg`
2. SSH ルールを更新します。
   a. セキュリティグループの選択 → インバウンドルールタブ → インバウンドルールの編集
   b. SSH ルールを検索する (ポート 22)
   c. ソース`0.0.0.0/0`を から IP に変更します。`[YOUR_IP]/32`
   d. https://whatismyipaddress.com から IP を取得する
   e. ルールの保存 をクリックします。

ステップ 2: 自動テスト実行を待機する

1. 自動テスト実行:

- CPU ストレステストはインスタンスの起動から 5 分後に自動的に開始されます
- 手動による介入は不要 - 待機するだけで、テストは完全にバックグラウンドで実行されます

2. テストをモニタリングします。
   - インスタンスはテストを自動的に起動して準備します
   - スクリプトは 5 分間実行され、>70% の CPU 使用率を生成します。
   - CloudWatch アラームは合計 8～10 分以内にトリガーされます (5 分の遅延 + アラームの場合は 3～5 分)
3. オプション: 手動再実行 (追加テスト用):
   - インスタンスに接続する: EC2 コンソール → `AWS-DevOpsAgent-Test-Instance`→ Connect → Session Manager
   - ストレステストを再度実行します。`./cpu-stress-test.sh`
   - AWS DevOpsAgent のレスポンスを複数回テストするのに最適です

テストオプション B: Lambda エラー率テスト

ステップ 1: Lambda テスト用に CloudFormation スタックをデプロイする

1. CloudFormation に移動します。
   a. AWS コンソールで、CloudFormation に移動します。
   b. スタックの作成 → 新しいリソースを使用する (標準) をクリックします。
2. テンプレートをアップロードします。
   a. という名前の新しいローカルファイルを作成する`AWS-DevOpsAgent-lambda-test.yaml`
   b. この CloudFormation テンプレートをコピーしてファイルに貼り付けます。
      i.

```
AWSTemplateFormatVersion: '2010-09-09'
Description: 'AWS DevOpsAgent Lambda Error Test Stack'
Resources:
  # IAM Role for Lambda function
  LambdaExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: AWS-DevOpsAgentLambdaTestRole
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
```

```
            Principal:
              Service: lambda.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole
      Tags:
        - Key: Name
          Value: AWS-DevOpsAgent-Lambda-Test-Role
        - Key: Purpose
          Value: AWS-DevOpsAgent-Testing
  # Lambda function that generates errors
  TestLambdaFunction:
    Type: AWS::Lambda::Function
    Properties:
      FunctionName: AWS-DevOpsAgent-test-lambda
      Runtime: python3.12
      Handler: index.lambda_handler
      Role: !GetAtt LambdaExecutionRole.Arn
      Code:
        ZipFile: |
          import json
          import random
          import time
          from datetime import datetime
          def lambda_handler(event, context):
              print(f"AWS DevOpsAgent Test Lambda - {datetime.now()}")
              print(f"Event: {json.dumps(event)}")

              # Intentionally generate errors for testing
              error_scenarios = [
                  "Simulated database connection timeout",
                  "Test API rate limit exceeded",
                  "Intentional validation error for AWS DevOpsAgent testing"
              ]

              # Always throw an error for testing purposes
              error_message = random.choice(error_scenarios)
              print(f"Generating test error: {error_message}")

              # This will create a Lambda error that CloudWatch will detect
              raise Exception(f"AWS DevOpsAgent Test Error: {error_message}")
      Description: AWS DevOpsAgent beta test function - intentionally generates
errors
      Timeout: 30
```

```
      Tags:
        - Key: Name
          Value: AWS-DevOpsAgent-Test-Lambda
        - Key: Purpose
          Value: AWS-DevOpsAgent-Testing
  # CloudWatch Alarm for Lambda errors
  LambdaErrorAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmName: AWS-DevOpsAgent-Lambda-Error-Test
      AlarmDescription: AWS-DevOpsAgent beta test - Lambda error rate alarm
      MetricName: Errors
      Namespace: AWS/Lambda
      Statistic: Sum
      Period: 60
      EvaluationPeriods: 1
      Threshold: 0
      ComparisonOperator: GreaterThanThreshold
      Dimensions:
        - Name: FunctionName
          Value: !Ref TestLambdaFunction
      TreatMissingData: notBreaching
Outputs:
  LambdaFunctionName:
    Description: Lambda Function Name for testing
    Value: !Ref TestLambdaFunction

  LambdaFunctionArn:
    Description: Lambda Function ARN
    Value: !GetAtt TestLambdaFunction.Arn

  AlarmName:
    Description: CloudWatch Alarm Name
    Value: !Ref LambdaErrorAlarm

  TestCommand:
    Description: AWS CLI command to test the function
    Value: !Sub 'aws lambda invoke --function-name ${TestLambdaFunction} --
payload "{\"test\":\"AWS DevOpsAgent validation\"}" response.json'
```

c. CloudFormation コンソールで、テンプレートファイルのアップロードを選択します。

d. ファイルの選択 をクリックします。

e. AWS-DevOpsAgent-lambda-test.yaml ファイルを選択する

f. [次へ] をクリックします。

3. スタックの設定:

a. スタック名:AWS-DevOpsAgent-Lambda-Test

b. [次へ] をクリックします。

4. スタックオプションを設定します。

a. デフォルトのまま、次へ をクリックします。

5. 確認と作成:

a. AWS CloudFormation が IAM リソースを作成する可能性があることを確認する

b. 送信 をクリックします。

6. 完了まで待ちます。

a. スタックの作成には 2 ~ 3 分かかります

b. ステータスは に変わりますCREATE_COMPLETE

ステップ 2: Lambda エラーをトリガーする

1. Lambda コンソールに移動します。

a. AWS Lambda コンソールに移動する

b. 関数を検索するAWS-DevOpsAgent-test-lambda

2. 関数をテストします。

a. テストタブをクリックします。

b. 新しいイベントの作成 をクリックします。

c. イベント名:AWS-DevOpsAgent-test-event

d. この JSON ペイロードを使用します。

i.
```
{
"test": "AWS DevOpsAgent validation",
"timestamp": "2024-01-01T00:00:00Z"
}
```

e. 保存 をクリックします。

3. エラーの生成:

a. テストボタンを 3 回クリックします (それぞれ 10 秒待機)

b. 各テストは意図的なエラーを生成します

c. CloudWatch アラームは 2～3 分以内にトリガーされます

d. AWS DevOpsAgent は、次にセットアップするオペレーターアプリで調査を使用してアラームを検出できるようになりました。

AWS DevOps エージェントの検出を検証する

ステップ 1: CloudWatch アラームのサニティチェック (オプション)

このステップは、上記のテストがアラーム状態になったことを確認するためのものです。

EC2 テストの場合:

- CloudWatch コンソールで、アラームに移動します。
- ストレステストを開始してから 3～5 分待ちます。
- アラームにアラーム状態が表示されます
- それでも「OK」の場合: さらに 2～3 分待ちます (CloudWatch メトリクスは遅延する可能性があります)

Lambda テストの場合:

- `AWS-DevOpsAgent-Lambda-Error-Test`アラームを確認する
- テストが実行されてから 2～3 分以内にアラーム内が表示されるはずです

ステップ 2: AWS DevOps エージェント調査を開始する

1. AWS DevOps Agent AgentSpace を開く
2. 管理者アクセスをクリックします。これにより、DevOps エージェントスペースウェブアプリが新しいウィンドウで開きます。
3. 画面の右側にある調査の開始ボタンをクリックします。
4. 次のフォームに入力します。
   a. 調査の詳細: 実行する調査を記述します。調査目標、調査対象領域、または関連情報についての詳細を含めます。
   b. 調査の開始点: 調査を開始する情報を記述します。アラーム、メトリクス、ログスニペットなどについて言及して、DevOps エージェントに作業の開始点を与えることができます。この場合、先ほど作成したアラームの概要を入力します。

c. インシデント日時 (ISO 8601 推奨): YYYY-MM-DDTHH:MMZ

d. 調査に名前を付けます。例: `Oncall_investigation_1:2025-10-27`

e. インシデントのAWS アカウント ID

f. インシデントが発生したリージョン

g. Priority - AWS DevOpsAgent では、2 つの同時調査が可能です。Priority を使用すると、調査の実行順序を定義できます。

5. 調査をクリックして調査を開始します。
6. ダッシュボードにリストされている調査をクリックします。調査の詳細画面が表示され、DevOps エージェントが実行している詳細な手順を表示できます。

期待される結果

EC2 テスト結果:

- EC2 CPU アラームを検出する
- 根本原因を特定します：「CPU ストレステストワークロード」
- タイムラインを表示: ストレステスト → CPU スパイク → アラーム
- モニタリングとスケーリングに関する推奨事項を提供します

Lambda テスト結果:

- Lambda エラー率の急増を検出
- 根本原因を特定します：「意図的なテスト例外」
- タイムラインを表示: 関数の呼び出し → エラー → アラーム
- エラー処理とモニタリングに関する推奨事項を提供します

クリーンアップ手順

クリーンアップテスト A (EC2 テスト)

自動クリーンアップ

- インスタンスは 2 時間後に自動終了します (CloudFormation テンプレートに組み込まれています)

手動クリーンアップ (即時)

1. CloudFormation スタックの削除:
   a. CloudFormation コンソールに移動する
   b. AWS-DevOpsAgent-EC2-Testスタックの選択
   c. 削除をクリックします
   d. 削除の確認
   e. これにより、EC2 インスタンス、セキュリティグループ、キーペア、CloudWatch アラームなどのすべてのリソースが自動的に削除されます。 EC2 CloudWatch

クリーンアップテスト B (Lambda テスト)

1. CloudFormation スタックの削除:
   a. CloudFormation コンソールに移動する
   b. AWS-DevOpsAgent-Lambda-Testスタックの選択
   c. 削除 をクリックします。
   d. 削除の確認
   e. これにより、Lambda 関数、IAM ロール、CloudWatch アラームなどのすべてのリソースが自動的に削除されます。 CloudWatch

トラブルシューティング

一般的な問題

EC2 インスタンスに接続できません」

- セキュリティグループを確認する: SSH (ポート 22) が IP に対して開いていることを確認します。
- キーアクセス許可の確認: 実行chmod 400 AWS-DevOpsAgent-test-key.pem
- パブリック IP の検証: インスタンスにはパブリック IP を割り当てる必要があります
- インスタンスの待機: インスタンスが「実行中」状態であることを確認します

「アラームがトリガーされない」

- メトリクスの待機: CloudWatch メトリクスが表示されるまでに 2～5 分かかる場合があります

- CPU 負荷の確認: インスタンスへの SSH と を実行して CPU >70% `top`を検証する
- ストレステストの検証: を実行して`ps aux | grep yes`、ロードプロセスが実行されているかどうかを確認します。
- 延長待機: 最初のアラームトリガーには最大 7～8 分かかることがあります

テスト検証

Your AWS DevOp エージェントのテストは、次の場合に成功します。

技術検証

- 調査精度: EC2 テストの結果は、CPU 負荷が原因でアラームがトリガーされたことを正しく示す必要があります。Lambda テストの結果は、これが意図的な失敗であることを示す必要があります。
- タイムラインの精度: 表示されているイベントの正しいシーケンス
- Recommendation Quality: 提示された実用的な提案

AWS CDK を使用した AWS DevOps エージェントの開始方法

概要:

このガイドでは、 AWS クラウド開発キット (AWS CDK) を使用して AWS DevOps エージェントリソースを作成およびデプロイする方法を示します。 AWS CDK アプリケーションは、 AWS CloudFormation を通じてエージェントスペース、 AWS Identity and Access Management (IAM) ロール、オペレーターアプリ、および AWS アカウントの関連付けの作成を自動化します。

AWS CDK アプローチは、必要なすべてのリソースをコードとしてのインフラストラクチャとして定義することで、 CLI オンボーディングガイドで説明されている手動ステップを自動化します。

AWS DevOps エージェントは、米国東部 (バージニア北部) 、米国西部 (オレゴン) 、アジアパシフィック (シドニー) 、アジアパシフィック (東京) 、欧州 (フランクフルト) 、欧州 (アイルランド) の 6 つの AWS リージョンで利用できます。サポートされているリージョンの詳細については、「」を参照してくださいthe section called “サポートされるリージョン”。

前提条件

作業を開始する前に、以下の準備が整っていることを確認します。

- AWS コマンドラインインターフェイス (AWS CLI) がインストールされ、適切な認証情報で設定されている
- Node.js バージョン 18 以降
- AWS CDK コマンドラインインターフェイス (CLI) がグローバルにインストールされています。AWS CDK CLI をインストールするには、次のコマンドを実行します。

```
npm install -g aws-cdk
```

- モニタリング (プライマリ) AWS アカウントの 1 つのアカウント
- (オプション) クロス AWS アカウントモニタリングを設定する場合は 2 番目のアカウント

このガイドの内容

このガイドは 2 つのパートに分かれています。

- パート 1 — オペレーターアプリと AWS 関連付けを使用してエージェントスペースをモニタリングアカウントにデプロイします。このパートを完了すると、エージェントはそのアカウントの問題をモニタリングできます。
- パート 2 (オプション) — サービスアカウントのソース AWS 関連付けを追加し、クロスアカウント IAM ロールをそのアカウントにデプロイします。この設定により、エージェントスペースはアカウント全体のリソースをモニタリングできます。

作成されるリソース

パート 1: DevOpsAgentStack (モニタリングアカウント)

- IAM ロール (`DevOpsAgentRole-AgentSpace`) — アカウントをモニタリングするために DevOps エージェントサービスによって引き受けられます。Resource Explorer サービスにリンクされたロールの作成を許可する `AIDevOpsAgentAccessPolicy`管理ポリシーとインラインポリシーが含まれます。
- IAM ロール (`DevOpsAgentRole-WebappAdmin`) — エージェントオペレーションの `AIDevOpsOperatorAppAccessPolicy`管理ポリシーを持つオペレーターアプリロール。
- エージェントスペース (`MyCDKAgentSpace`) — `AWS::DevOpsAgent::AgentSpace` CloudFormation リソースを使用して作成された中央エージェントスペース。オペレーターアプリの設定が含まれます。

- 関連付け (AWS モニター) — `AWS::DevOpsAgent::Association` CloudFormation リソースを使用して、モニタリングアカウントをエージェントスペースにリンクします。
- 関連付け (AWS ソース) — (オプション) サービスアカウントをエージェントスペースにリンクして、クロスアカウントモニタリングを行います。

パート 2: ServiceStack (サービスアカウント、オプション)

- IAM ロール (`DevOpsAgentRole-SecondaryAccount`) — 固定名を持つクロスアカウントロール。モニタリングアカウントのエージェントスペースによって信頼されます。Resource Explorer サービスにリンクされたロールの作成を許可する `AIDevOpsAgentAccessPolicy`管理ポリシーとインラインポリシーが含まれます。
- Lambda 関数 (`echo-service`) — 入力イベントをエコーバックするシンプルなサービス例。

セットアップ

ステップ 1: サンプルリポジトリのクローンを作成する

次のコマンドを実行してリポジトリのクローンを作成し、プロジェクトディレクトリに変更します。

```
git clone https://github.com/aws-samples/sample-aws-devops-agent-cdk.git
cd sample-aws-devops-agent-cdk
```

ステップ 2: 依存関係をインストールする

次のコマンドを実行して、プロジェクトの依存関係をインストールします。

```
npm install
```

パート 1: エージェントスペースをデプロイする

このセクションでは、エージェントスペース、IAM ロール、オペレーターアプリ、および AWS 関連付けをモニタリングアカウントに作成します。

ステップ 1: モニタリングアカウント ID を設定する

を開き`lib/constants.ts`、モニタリングアカウント ID を設定します。

次の例は、更新する定数を示しています。

```
export const MONITORING_ACCOUNT_ID = "<YOUR_MONITORING_ACCOUNT_ID>";
```

ステップ 2: AWS CDK 環境をブートストラップする

モニタリングアカウントで AWS CDK をブートストラップしていない場合は、次のコマンドを実行します。

```
cdk bootstrap aws://<MONITORING_ACCOUNT_ID>/<REGION> --profile monitoring
```

ステップ 3: ビルドとデプロイ

次のコマンドを実行して TypeScript コードを構築し、スタックをデプロイします。

```
npm run build
cdk deploy DevOpsAgentStack --profile monitoring
```

ステップ 4: スタック出力を記録する

デプロイが完了すると、 AWS CDK はスタック出力を出力します。これらの値は後で使用するために記録します。

次の例は、予想される出力を示しています。

```
Outputs:
DevOpsAgentStack.AgentSpaceArn = arn:aws:aidevops:<REGION>:123456789012:agentspace/
abc123
DevOpsAgentStack.AgentSpaceRoleArn = arn:aws:iam::123456789012:role/DevOpsAgentRole-
AgentSpace
DevOpsAgentStack.OperatorRoleArn = arn:aws:iam::123456789012:role/DevOpsAgentRole-
WebappAdmin
DevOpsAgentStack.AssociationId = assoc-xyz
```

パート 2 を完了する予定がある場合は、AgentSpaceArn値を保存します。サービスアカウントスタックを設定するには、これが必要です。

ステップ 5: デプロイを検証する

エージェントスペースが正常に作成されたことを確認するには、次の AWS CLI コマンドを実行します。

```
aws devopsagent get-agent-space \
```

```
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>
```

この時点で、エージェントスペースはオペレーターアプリが有効で、モニタリングアカウントが関連付けられている状態でデプロイされます。エージェントはこのアカウントの問題をモニタリングできます。

パート 2 (オプション): クロスアカウントモニタリングを追加する

このセクションでは、エージェントスペースが 2 番目の AWS アカウント (サービスアカウント) のリソースをモニタリングできるようにセットアップを拡張します。これには、次の 2 つのアクションが含まれます。

1. サービスアカウントを指すソース AWS 関連付けを DevOpsAgentStack に追加します。
2. エージェントスペースを信頼する IAM ロールを使用して ServiceStack をサービスアカウントにデプロイします。

> **Important**
>
> 続行する前にパート 1 を完了する必要があります。ServiceStack には、DevOpsAgentStack デプロイ出力`AgentSpaceArn`から が必要です。

ステップ 1: サービスアカウント ID を設定する

を開き`lib/constants.ts`、サービスアカウント ID を設定します。

次の例は、更新する定数を示しています。

```
export const SERVICE_ACCOUNT_ID = "<YOUR_SERVICE_ACCOUNT_ID>";
```

DevOpsAgentStack は、このアカウント ID を使用してソース AWS の関連付けを作成します。この値を設定する前に DevOpsAgentStack をデプロイした場合は、再デプロイして関連付けを作成します。

次のコマンドを実行して再デプロイします。

```
npm run build
cdk deploy DevOpsAgentStack --profile monitoring
```

ステップ 2: エージェントスペース ARN を設定する

DevOpsAgentStack 出力 (パート 1、ステップ 4) からAgentSpaceArn値をコピーし、 で設定しますlib/constants.ts。

次の例は、更新する定数を示しています。

```
export const AGENT_SPACE_ARN =
 "arn:aws:aidevops:<REGION>:<MONITORING_ACCOUNT_ID>:agentspace/<SPACE_ID>";
```

ServiceStack はこの値を使用して、セカンダリアカウントロールの信頼ポリシーの範囲を設定します。ServiceStack は、この値が設定されている場合にのみ合成されます。

ステップ 3: サービスアカウントをブートストラップする

サービスアカウントで AWS CDK をブートストラップしていない場合は、次のコマンドを実行します。

```
cdk bootstrap aws://<SERVICE_ACCOUNT_ID>/<REGION> --profile service
```

ステップ 4: ServiceStack をデプロイする

次のコマンドを実行して、サービスアカウントの認証情報を使用して ServiceStack を構築およびデプロイします。

```
npm run build
cdk deploy ServiceStack --profile service
```

これにより、サービスアカウントに次のリソースが作成されます。

- モニタリングアカウントのエージェントスペースを信頼する IAM ロール (DevOpsAgentRole-SecondaryAccount)
- サンプルサービスとしてのエコー Lambda 関数 (echo-service)

ステップ 5: デプロイを検証する

Lambda 関数が正常にデプロイされたことを確認するには、次のコマンドを実行してエコーサービスをテストします。

```
aws lambda invoke \
  --function-name echo-service \
  --payload '{"test": "hello world"}' \
  --profile service \
  response.json
cat response.json
```

トラブルシューティング

このセクションでは、一般的な問題とその解決方法について説明します。

CloudFormation リソースタイプが見つかりません

- にデプロイしていることを確認しますthe section called “サポートされるリージョン”。
- CLI AWS に適切なアクセス許可が設定されていることを確認します。

IAM ロールの作成に失敗しました

- デプロイロールに IAM ロールを作成するアクセス許可があることを確認します。
- 信頼ポリシーの条件がアカウント ID と一致していることを確認します。

クロスアカウントデプロイが「ターゲットアカウントでロールを引き受けることができませんでした」で失敗する

- 各スタックは、ターゲットアカウントの認証情報を使用してデプロイする必要があります。`--profile` フラグを使用して、正しい CLI AWS プロファイルを指定します。
- AWS CDK がターゲットアカウントでブートストラップされていることを確認します。

IAM 伝達の遅延

- IAM ロールの変更が反映されるまでに数分かかる場合があります。ロールの作成直後にエージェントスペースの作成が失敗した場合、数分待ってから再デプロイします。

クリーンアップ

すべてのリソースを削除するには、スタックを逆の順序で破棄します。

次のコマンドを実行して、スタックを破棄します。

```
# If you deployed the ServiceStack, destroy it first
cdk destroy ServiceStack --profile service
# Then destroy the DevOpsAgentStack
cdk destroy DevOpsAgentStack --profile monitoring
```

警告: このアクションは、エージェントスペースと関連するすべてのデータを完全に削除します。このアクションは元に戻すことができません。続行する前に、重要な情報がバックアップされていることを確認してください。

セキュリティに関する考慮事項

- AWS CDK アプリケーションは、`aidevops.amazonaws.com`サービスプリンシパルのみが引き受けることを許可する信頼ポリシーを持つ IAM ロールを作成します。
- 信頼ポリシーには、特定の AWS アカウントとエージェントスペースの ARN へのアクセスを制限する条件が含まれます。
- すべてのポリシーは最小特権の原則に従います。組織のセキュリティ要件に基づいて IAM ポリシーを確認してカスタマイズします。
- クロスアカウントロール (`DevOpsAgentRole-SecondaryAccount`) は固定名を使用し、特定のエージェントスペース ARN に限定されます。

次の手順

AWS CDK を使用して AWS DevOps エージェントをデプロイした後:

1. DevOps エージェント機能の全範囲については、「DevOps AWS DevOps エージェントユーザーガイド」を参照してください。
2. 自動インフラストラクチャ管理のために AWS 、CDK デプロイを CI/CD パイプラインに統合することを検討してください。

その他のリソース

- AWS DevOps エージェントユーザーガイド
- GitHub ウェブサイトのサンプル CDK リポジトリ
- CLI オンボーディングガイド

AWS CloudFormation を使用した AWS DevOps エージェントの開始方法

概要:

このガイドでは、 AWS CloudFormation テンプレートを使用して AWS DevOps エージェントリソースを作成およびデプロイする方法を示します。テンプレートは、エージェントスペース、 AWS Identity and Access Management (IAM) ロール、オペレーターアプリ、 AWS およびアカウント関連付けをコードとしてのインフラストラクチャとして作成することを自動化します。

CloudFormation アプローチは、宣言型 YAML テンプレートで必要なすべてのリソースを定義することで、CLI オンボーディングガイドで説明されている手動ステップを自動化します。

AWS DevOps エージェントは、米国東部 (バージニア北部) 、米国西部 (オレゴン) 、アジアパシフィック (シドニー) 、アジアパシフィック (東京) 、欧州 (フランクフルト) 、欧州 (アイルランド) の 6 つの AWS リージョンで利用できます。サポートされているリージョンの詳細については、「」を参照してくださいthe section called “サポートされるリージョン”。

前提条件

作業を開始する前に、以下の準備が整っていることを確認します。

- AWS コマンドラインインターフェイス (AWS CLI) がインストールされ、適切な認証情報で設定されている
- IAM ロールと CloudFormation スタックを作成するアクセス許可
- モニタリング (プライマリ) AWS アカウントの 1 つのアカウント
- (オプション) クロス AWS アカウントモニタリングを設定する場合は 2 番目のアカウント

このガイドの内容

このガイドは 2 つのパートに分かれています。

- パート 1 — オペレーターアプリと AWS 関連付けを使用してエージェントスペースをモニタリングアカウントにデプロイします。このパートを完了すると、エージェントはそのアカウントの問題をモニタリングできます。

- パート 2 (オプション) — クロスアカウント IAM ロールをセカンダリアカウントにデプロイし、ソースの AWS 関連付けを追加します。この設定により、エージェントスペースはアカウント全体のリソースをモニタリングできます。

パート 1: エージェントスペースをデプロイする

このセクションでは、モニタリングアカウントのエージェントスペース、IAM ロール、オペレータアプリ、および AWS 関連付けをプロビジョニングする CloudFormation テンプレートを作成します。

ステップ 1: CloudFormation テンプレートを作成する

次のテンプレートを として保存しますdevops-agent-stack.yaml。

```
AWSTemplateFormatVersion: '2010-09-09'
Description: AWS DevOps Agent - Agent Space with IAM roles, operator app, and AWS
 association

Parameters:
  AgentSpaceName:
    Type: String
    Default: MyCloudFormationAgentSpace
    Description: Name for the agent space
  AgentSpaceDescription:
    Type: String
    Default: Agent space deployed with CloudFormation
    Description: Description for the agent space

Resources:
  # IAM role assumed by the DevOps Agent service to monitor the account
  DevOpsAgentSpaceRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: DevOpsAgentRole-AgentSpace
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: aidevops.amazonaws.com
            Action: sts:AssumeRole
            Condition:
```

```
              StringEquals:
                aws:SourceAccount: !Ref AWS::AccountId
              ArnLike:
                aws:SourceArn: !Sub arn:aws:aidevops:${AWS::Region}:
${AWS::AccountId}:agentspace/*
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/AIDevOpsAgentAccessPolicy
      Policies:
        - PolicyName: AllowCreateServiceLinkedRoles
          PolicyDocument:
            Version: '2012-10-17'
            Statement:
              - Sid: AllowCreateServiceLinkedRoles
                Effect: Allow
                Action:
                  - iam:CreateServiceLinkedRole
                Resource:
                  - !Sub arn:aws:iam::${AWS::AccountId}:role/aws-service-role/resource-
explorer-2.amazonaws.com/AWSServiceRoleForResourceExplorer

  # IAM role for the operator app interface
  DevOpsOperatorRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: DevOpsAgentRole-WebappAdmin
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: aidevops.amazonaws.com
            Action:
              - sts:AssumeRole
              - sts:TagSession
            Condition:
              StringEquals:
                aws:SourceAccount: !Ref AWS::AccountId
              ArnLike:
                aws:SourceArn: !Sub arn:aws:aidevops:${AWS::Region}:
${AWS::AccountId}:agentspace/*
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/AIDevOpsOperatorAppAccessPolicy

  # The agent space resource
```

```
  AgentSpace:
    Type: AWS::DevOpsAgent::AgentSpace
    DependsOn:
      - DevOpsAgentSpaceRole
      - DevOpsOperatorRole
    Properties:
      Name: !Ref AgentSpaceName
      Description: !Ref AgentSpaceDescription
      OperatorApp:
        Iam:
          OperatorAppRoleArn: !GetAtt DevOpsOperatorRole.Arn

  # Association linking the monitoring account to the agent space
  MonitorAssociation:
    Type: AWS::DevOpsAgent::Association
    Properties:
      AgentSpaceId: !GetAtt AgentSpace.AgentSpaceId
      ServiceId: aws
      Configuration:
        Aws:
          AssumableRoleArn: !GetAtt DevOpsAgentSpaceRole.Arn
          AccountId: !Ref AWS::AccountId
          AccountType: monitor

Outputs:
  AgentSpaceId:
    Description: The agent space ID
    Value: !GetAtt AgentSpace.AgentSpaceId
  AgentSpaceArn:
    Description: The agent space ARN
    Value: !GetAtt AgentSpace.Arn
  AgentSpaceRoleArn:
    Description: The agent space IAM role ARN
    Value: !GetAtt DevOpsAgentSpaceRole.Arn
  OperatorRoleArn:
    Description: The operator app IAM role ARN
    Value: !GetAtt DevOpsOperatorRole.Arn
```

ステップ 2: スタックをデプロイする

次のコマンドを実行してスタックをデプロイします。を <REGION>に置き換えます the section called “サポートされるリージョン” (例: us-east-1)。

```
aws cloudformation deploy \
  --template-file devops-agent-stack.yaml \
  --stack-name DevOpsAgentStack \
  --capabilities CAPABILITY_NAMED_IAM \
  --region <REGION>
```

ステップ 3: スタック出力を記録する

デプロイが完了したら、次のコマンドを実行してスタック出力を取得します。これらの値は後で使用するために記録します。

```
aws cloudformation describe-stacks \
  --stack-name DevOpsAgentStack \
  --query 'Stacks[0].Outputs' \
  --region <REGION>
```

次の例は、予想される出力を示しています。

```
[
  {
    "OutputKey": "AgentSpaceId",
    "OutputValue": "abc123def456"
  },
  {
    "OutputKey": "AgentSpaceArn",
    "OutputValue": "arn:aws:aidevops:<REGION>:<ACCOUNT_ID>:agentspace/abc123def456"
  },
  {
    "OutputKey": "AgentSpaceRoleArn",
    "OutputValue": "arn:aws:iam::<ACCOUNT_ID>:role/DevOpsAgentRole-AgentSpace"
  },
  {
    "OutputKey": "OperatorRoleArn",
    "OutputValue": "arn:aws:iam::<ACCOUNT_ID>:role/DevOpsAgentRole-WebappAdmin"
  }
]
```

パート 2 を完了する予定がある場合は、AgentSpaceArn値を保存します。クロスアカウントロールを設定するには、これが必要です。

ステップ 4: デプロイを検証する

エージェントスペースが正常に作成されたことを確認するには、次の AWS CLI コマンドを実行します。

```
aws devops-agent get-agent-space \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>
```

この時点で、エージェントスペースはオペレーターアプリが有効で、モニタリングアカウントが関連付けられている状態でデプロイされます。エージェントはこのアカウントの問題をモニタリングできます。

パート 2 (オプション): クロスアカウントモニタリングを追加する

このセクションでは、エージェントスペースが 2 番目の AWS アカウント (サービスアカウント) のリソースをモニタリングできるようにセットアップを拡張します。これには、次の 2 つのアクションが含まれます。

1. エージェントスペースを信頼するサービスアカウントに IAM ロールをデプロイします。
2. サービスアカウントを指すソース AWS 関連付けをモニタリングアカウントに追加します。

> 注: 続行する前にパート 1 を完了する必要があります。サービスアカウントテンプレートには、パート 1 スタック出力AgentSpaceArnの が必要です。

ステップ 1: サービスアカウントテンプレートを作成する

次のテンプレートを として保存しますdevops-agent-service-account.yaml。このテンプレートは、セカンダリアカウントにクロスアカウント IAM ロールを作成します。

```
AWSTemplateFormatVersion: '2010-09-09'
Description: AWS DevOps Agent - Cross-account IAM role for secondary account monitoring

Parameters:
  MonitoringAccountId:
    Type: String
    Description: The 12-digit AWS account ID of the monitoring account
  AgentSpaceArn:
    Type: String
```

```
    Description: The ARN of the agent space from the monitoring account

Resources:
  # Cross-account IAM role trusted by the agent space
  DevOpsSecondaryAccountRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: DevOpsAgentRole-SecondaryAccount
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: aidevops.amazonaws.com
            Action: sts:AssumeRole
            Condition:
              StringEquals:
                aws:SourceAccount: !Ref MonitoringAccountId
              ArnLike:
                aws:SourceArn: !Ref AgentSpaceArn
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/AIDevOpsAgentAccessPolicy
      Policies:
        - PolicyName: AllowCreateServiceLinkedRoles
          PolicyDocument:
            Version: '2012-10-17'
            Statement:
              - Sid: AllowCreateServiceLinkedRoles
                Effect: Allow
                Action:
                  - iam:CreateServiceLinkedRole
                Resource:
                  - !Sub arn:aws:iam::${AWS::AccountId}:role/aws-service-role/resource-
explorer-2.amazonaws.com/AWSServiceRoleForResourceExplorer

Outputs:
  SecondaryAccountRoleArn:
    Description: The cross-account IAM role ARN
    Value: !GetAtt DevOpsSecondaryAccountRole.Arn
```

ステップ 2: サービスアカウントスタックをデプロイする

サービスアカウントの認証情報を使用して、次のコマンドを実行します。

```
aws cloudformation deploy \
  --template-file devops-agent-service-account.yaml \
  --stack-name DevOpsAgentServiceAccountStack \
  --capabilities CAPABILITY_NAMED_IAM \
  --parameter-overrides \
    MonitoringAccountId=<MONITORING_ACCOUNT_ID> \
    AgentSpaceArn=<AGENT_SPACE_ARN> \
  --region <REGION>
```

ステップ 3: ソースの AWS 関連付けを追加する

モニタリングアカウントに切り替えて、ソースの AWS 関連付けを作成します。これを行うには、別のスタックを作成するか、元のテンプレートを更新します。次の例では、スタンドアロンテンプレートを使用しています。

次のテンプレートを として保存しますdevops-agent-source-association.yaml。

```
AWSTemplateFormatVersion: '2010-09-09'
Description: AWS DevOps Agent - Source AWS association for cross-account monitoring

Parameters:
  AgentSpaceId:
    Type: String
    Description: The agent space ID from the monitoring account stack
  ServiceAccountId:
    Type: String
    Description: The 12-digit AWS account ID of the service account
  ServiceAccountRoleArn:
    Type: String
    Description: The ARN of the DevOpsAgentRole-SecondaryAccount role in the service
 account

Resources:
  SourceAssociation:
    Type: AWS::DevOpsAgent::Association
    Properties:
      AgentSpaceId: !Ref AgentSpaceId
      ServiceId: aws
      Configuration:
        SourceAws:
          AccountId: !Ref ServiceAccountId
          AccountType: source
```

```
          AssumableRoleArn: !Ref ServiceAccountRoleArn

Outputs:
  SourceAssociationId:
    Description: The source association ID
    Value: !Ref SourceAssociation
```

モニタリングアカウントの認証情報を使用して関連付けスタックをデプロイします。

```
aws cloudformation deploy \
  --template-file devops-agent-source-association.yaml \
  --stack-name DevOpsAgentSourceAssociationStack \
  --parameter-overrides \
    AgentSpaceId=<AGENT_SPACE_ID> \
    ServiceAccountId=<SERVICE_ACCOUNT_ID> \
    ServiceAccountRoleArn=arn:aws:iam::<SERVICE_ACCOUNT_ID>:role/DevOpsAgentRole-
SecondaryAccount \
  --region <REGION>
```

検証

次の AWS CLI コマンドを実行して、セットアップを確認します。

```
# List your agent spaces
aws devops-agent list-agent-spaces \
  --region <REGION>

# Get details of a specific agent space
aws devops-agent get-agent-space \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>

# List associations for an agent space
aws devops-agent list-associations \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>
```

トラブルシューティング

このセクションでは、一般的な問題とその解決方法について説明します。

CloudFormation リソースタイプが見つかりません

- にデプロイしていることを確認しますthe section called “サポートされるリージョン”。
- CLI AWS に適切なアクセス許可が設定されていることを確認します。

IAM ロールの作成に失敗しました

- デプロイ認証情報に、カスタム名 () で IAM ロールを作成するアクセス許可があることを確認しますCAPABILITY_NAMED_IAM。
- 信頼ポリシーの条件がアカウント ID と一致していることを確認します。

クロスアカウントデプロイが失敗する

- 各スタックは、ターゲットアカウントの認証情報を使用してデプロイする必要があります。`--profile` フラグを使用して、正しい CLI AWS プロファイルを指定します。
- `AgentSpaceArn` パラメータがパート 1 スタック出力の正確な ARN と一致することを確認します。

IAM 伝達の遅延

- IAM ロールの変更が反映されるまでに数分かかる場合があります。ロールの作成直後にエージェントスペースの作成が失敗した場合、数分待ってから再デプロイします。

クリーンアップ

すべてのリソースを削除するには、スタックを逆の順序で削除します。

警告: このアクションは、エージェントスペースと関連するすべてのデータを完全に削除します。このアクションは元に戻すことができません。続行する前に、重要な情報がバックアップされていることを確認してください。

次のコマンドを実行して、スタックを削除します。

```
# If you deployed the source association stack, delete it first
aws cloudformation delete-stack \
  --stack-name DevOpsAgentSourceAssociationStack \
  --region <REGION>

aws cloudformation wait stack-delete-complete \
  --stack-name DevOpsAgentSourceAssociationStack \
```

```
  --region <REGION>

# If you deployed the service account stack, delete it next (using service account
 credentials)
aws cloudformation delete-stack \
  --stack-name DevOpsAgentServiceAccountStack \
  --region <REGION>

aws cloudformation wait stack-delete-complete \
  --stack-name DevOpsAgentServiceAccountStack \
  --region <REGION>

# Delete the main stack last
aws cloudformation delete-stack \
  --stack-name DevOpsAgentStack \
  --region <REGION>
```

次の手順

AWS CloudFormation を使用して AWS DevOps エージェントをデプロイした後:

- 追加の統合を接続するには、「」を参照してくださいAWS DevOps Agent の機能の設定。
- エージェントのスキルと機能の詳細については、「」を参照してくださいthe section called “DevOps エージェントスキル”。
- オペレーターウェブアプリを理解するには、「」を参照してくださいthe section called “DevOps エージェントウェブアプリとは”。

Terraform を使用した AWS DevOps エージェントの開始方法

概要:

このガイドでは、Terraform を使用して AWS DevOps エージェントリソースを作成およびデプロイする方法を示します。Terraform 設定は、エージェントスペース、IAM ロール、オペレーターアプリ、および AWS アカウントの関連付けの作成を自動化します。

Terraform アプローチは、必要なすべてのリソースをコードとしてのインフラストラクチャとして定義することで、 CLI オンボーディングガイドで説明されている手動ステップを自動化します。

AWS DevOps エージェントは、米国東部 (バージニア北部) 、米国西部 (オレゴン) 、アジアパシフィック (シドニー) 、アジアパシフィック (東京) 、欧州 (フランクフルト) 、欧州 (アイルランド)

の 6 つの AWS リージョンで利用できます。サポートされているリージョンの詳細については、「」を参照してくださいthe section called “サポートされるリージョン”。

前提条件

作業を開始する前に、次の項目があることを確認します。

- Terraform >= 1.0 がインストールされている
- AWS CLI がインストールされ、適切な認証情報で設定されている
- モニタリング (プライマリ) AWS アカウントの 1 つのアカウント
- (オプション) クロス AWS アカウントモニタリングを設定する場合の 2 番目のアカウント

このガイドで取り上げる内容

このガイドは 2 つのパートに分かれています。

- パート 1 — オペレーターアプリと AWS 関連付けを使用してエージェントスペースをモニタリングアカウントにデプロイします。このパートを完了すると、エージェントはそのアカウントの問題をモニタリングできます。
- パート 2 (オプション) — サービスアカウントのソース AWS 関連付けを追加し、クロスアカウント IAM ロールとエコー Lambda をそのアカウントにデプロイします。これにより、エージェントスペースはアカウント全体のリソースをモニタリングできます。

作成されるリソース

パート 1: アカウントのモニタリング

- IAM ロール (`DevOpsAgentRole-AgentSpace-*`) — アカウントをモニタリングするために DevOps エージェントサービスによって引き受けられます。Resource Explorer サービスにリンクされたロールの作成を許可する `AIDevOpsAgentAccessPolicy`管理ポリシーとインラインポリシーが含まれます。
- IAM ロール (`DevOpsAgentRole-WebappAdmin-*`) — エージェントオペレーション用の `AIDevOpsOperatorAppAccessPolicy`管理ポリシーを持つオペレーターアプリケーションロール。
- エージェントスペース (設定可能な名前) — `awscc_devopsagent_agent_space`リソースを使用して作成された中央エージェントスペース。オペレーターアプリの設定が含まれます。

- 関連付け (AWS モニター) — awscc_devopsagent_associationリソースを使用して、モニタリングアカウントをエージェントスペースにリンクします。
- 関連付け (AWS ソース) — (オプション) サービスアカウントをエージェントスペースにリンクして、クロスアカウントモニタリングを行います。

パート 2: サービスアカウント (オプション)

- IAM ロール (DevOpsAgentRole-SecondaryAccount-TF) — 固定名を持つクロスアカウントロール。モニタリングアカウントのエージェントスペースによって信頼されます。Resource Explorer サービスにリンクされたロールの作成を許可する AIDevOpsAgentAccessPolicy管理ポリシーとインラインポリシーが含まれます。
- Lambda 関数 (echo-service-tf) — 入力イベントをエコーバックするシンプルなサービス例。

セットアップ

ステップ 1: サンプルリポジトリのクローンを作成する

```
git clone https://github.com/aws-samples/sample-aws-devops-agent-terraform.git
cd sample-aws-devops-agent-terraform
```

ステップ 2: 変数を設定する

サンプル変数ファイルをコピーし、環境に合わせてカスタマイズします。

```
cp terraform.tfvars.example terraform.tfvars
```

エージェントスペースの名前と説明terraform.tfvarsを使用して を編集します。

```
agent_space_name        = "MyCompanyAgentSpace"
agent_space_description = "DevOps Agent Space for monitoring production workloads"
```

パート 1: エージェントスペースをデプロイする

このセクションでは、モニタリングアカウントにエージェントスペース、IAM ロール、オペレーターアプリ、および AWS 関連付けを作成します。

ステップ 1: 自動化を使用してデプロイする (推奨)

提供されているデプロイスクリプトを使用して、セットアップを合理化します。

```
./deploy.sh
```

このスクリプトは自動的に次のようになります。

- 前提条件を確認する (Terraform、 AWS CLI、認証情報)
- 必要に応じて例`terraform.tfvars`から を作成します。
- Terraform の初期化、検証、計画、適用

または、手動制御が必要な場合は、次の操作を行います。

```
terraform init
terraform plan
terraform apply
```

デプロイを確認するプロンプトが表示され`yes`たら、 と入力します。

ステップ 2: 出力を記録する

デプロイが完了すると、Terraform は出力を出力します。後で使用するために、これらの値を記録します。

```
Outputs:
agent_space_id              = "abc123"
agent_space_arn             =
 "arn:aws:aidevops:<REGION>:<MONITORING_ACCOUNT_ID>:agentspace/abc123"
agent_space_name            = "MyCompanyAgentSpace"
devops_agentspace_role_arn  = "arn:aws:iam::<MONITORING_ACCOUNT_ID>:role/
DevOpsAgentRole-AgentSpace-a1b2c3d4"
devops_operator_role_arn    = "arn:aws:iam::<MONITORING_ACCOUNT_ID>:role/
DevOpsAgentRole-WebappAdmin-a1b2c3d4"
primary_account_id          = "<MONITORING_ACCOUNT_ID>"
primary_account_association_id = "assoc-xyz"
```

パート 2 を完了する予定がある場合は、`agent_space_arn`値を保存します。サービスアカウントリソースを設定するには、これが必要です。

ステップ 3: デプロイを検証する

デプロイ後の検証スクリプトを実行します。

```
./post-deploy.sh
```

または、CLI AWS を使用して、エージェントスペースが正常に作成されたことを確認します。

```
aws devops-agent get-agent-space \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>
```

この時点で、エージェントスペースはオペレーターアプリが有効で、モニタリングアカウントが関連付けられている状態でデプロイされます。エージェントはこのアカウントの問題をモニタリングできます。

パート 2 (オプション): クロスアカウントモニタリングを追加する

このセクションでは、エージェントスペースが 2 番目の AWS アカウント (サービスアカウント) のリソースをモニタリングできるようにセットアップを拡張します。これには、次の 2 つのアクションが含まれます。

1. サービスアカウントを指すソース AWS の関連付けを追加します。
2. クロスアカウント IAM ロールとエコー Lambda 関数をサービスアカウントにデプロイします。

> **Important**
>
> 続行する前にパート 1 を完了する必要があります。サービスアカウントリソースには、パート 1 デプロイ出力`agent_space_arn`の が必要です。

ステップ 1: サービスアカウント ID を設定する

で`terraform.tfvars`、サービスアカウント ID を設定します。

```
service_account_id = "<YOUR_SERVICE_ACCOUNT_ID>"
```

ステップ 2: エージェントスペース ARN を設定する

パート 1 出力 (ステップ 2) からagent_space_arn値をコピーし、 で設定しますterraform.tfvars。

```
agent_space_arn = "arn:aws:aidevops:<REGION>:<MONITORING_ACCOUNT_ID>:agentspace/
<SPACE_ID>"
```

サービスアカウントリソースは、この値を使用して、セカンダリアカウントロールの信頼ポリシーの範囲を設定します。これらのリソースは、この値が設定されている場合にのみ作成されます。

ステップ 3: 「aws.service」プロバイダーを設定する

でmain.tf、サービスアカウントの認証情報を使用してaws.serviceプロバイダーエイリアスを設定します。名前付きプロファイルまたは継承ロールを使用できます。

プロファイルの使用:

```
provider "aws" {
  alias   = "service"
  region  = var.aws_region
  profile = "your-service-account-profile"
}
```

または、継承ロールを使用します。

```
provider "aws" {
  alias  = "service"
  region = var.aws_region
  assume_role {
    role_arn = "arn:aws:iam::<SERVICE_ACCOUNT_ID>:role/OrganizationAccountAccessRole"
  }
}
```

ステップ 4: デプロイする

更新された設定を適用します。

```
terraform apply
```

これにより、サービスアカウントに次のリソースが作成されます。

- モニタリングアカウントのエージェントスペースを信頼する IAM ロール (DevOpsAgentRole-SecondaryAccount-TF)
- サンプルサービスとしてのエコー Lambda 関数 (echo-service-tf)

また、サービスアカウントをリンクするソース AWS 関連付けをモニタリングアカウントに作成します。

ステップ 5: デプロイを検証する

エコーサービスをテストして、Lambda 関数が正常にデプロイされたことを確認します。

```
aws lambda invoke \
  --function-name echo-service-tf \
  --payload '{"test": "hello world"}' \
  --profile <your-service-account-profile> \
  --region <REGION> \
  response.json
cat response.json
```

トラブルシューティング

IAM 伝達の遅延

- 設定には、IAM ロールの作成からエージェントスペースの作成time_sleepまでの 30 秒が含まれます。DevOps エージェントサービスは、エージェントスペースの作成中にオペレーターロールの信頼ポリシーを検証します。IAM が完全に伝播されていない場合、これは失敗する可能性があります。それでも信頼ポリシーエラーが表示される場合は、1 分待ってterraform applyからもう一度実行してください。IAM ロールはすでに存在し、適用は中断した場所を選択します。

アクセス許可エラー

- AWS 認証情報にロールとポリシーを作成するために必要な IAM アクセス許可があることを確認します。
- 信頼ポリシーの条件がアカウント ID と一致していることを確認します。

クロスアカウントデプロイが失敗する

- aws.service プロバイダーは、サービスアカウントの認証情報で設定する必要があります。名前付きプロファイルまたは継承ロールブロックを使用します。
- agent_space_arn 値がパート 1 出力の ARN と一致することを確認します。

Terraform リソースタイプが見つかりません

- awscc プロバイダーのバージョンが ~> 1.0以降であることを確認します。awscc_devopsagent_agent_space および awscc_devopsagent_associationリソースには AWS Cloud Control プロバイダーが必要です。

クリーンアップ

すべてのリソースを削除するには、パート 2 をデプロイした場合は逆の順序で破棄します。

```
./cleanup.sh
```

または、手動で以下を実行します。

```
terraform destroy
```

警告: これにより、エージェントスペースと関連するすべてのデータが完全に削除されます。続行する前に、重要な情報をバックアップしていることを確認してください。

セキュリティに関する考慮事項

- Terraform 設定は、aidevops.amazonaws.comサービスプリンシパルのみがロールを引き受けることを許可する信頼ポリシーを持つ IAM ロールを作成します。
- 信頼ポリシーには、特定の AWS アカウントとエージェントスペースの ARN へのアクセスを制限する条件が含まれます。
- すべてのポリシーは最小特権の原則に従います。組織のセキュリティ要件に基づいて IAM ポリシーを確認してカスタマイズします。
- クロスアカウントロール (DevOpsAgentRole-SecondaryAccount-TF) は固定名を使用し、特定のエージェントスペース ARN に限定されます。

次の手順

Terraform を使用して AWS DevOps エージェントをデプロイした後:

1. DevOps エージェント機能の全範囲については、「DevOps AWS DevOps エージェントユーザーガイド」を参照してください。
2. 自動インフラストラクチャ管理のために、Terraform デプロイを CI/CD パイプラインに統合することを検討してください。

その他のリソース

- AWS DevOps エージェントユーザーガイド
- サンプル Terraform リポジトリ
- CLI オンボーディングガイド

DevOps エージェントの使用

DevOps エージェントの使用

AWS DevOps Agent は、検出から調査、復旧、防止まで、インシデントライフサイクル全体で運用チームと協力して作業します。以下のトピックでは、DevOps エージェントを使用してこのライフサイクルの各フェーズを管理する方法について説明します。

自律型インシデント対応

インシデントが検出されると、チケット発行システムとの組み込み統合、モニタリングツールからのウェブフック、手動トリガーのいずれによっても、DevOps エージェントは自動的に調査を開始します。エージェントは、メトリクス、ログ、トレース、コード変更、デプロイ履歴を分析して根本原因を特定し、緩和計画を提案します。さらにサポートが必要な場合は、DevOps Agent Space ウェブアプリから AWS Support に直接エスカレーションできます。これにより、調査コンテキストがサポートエンジニアと自動的に共有されるため、エージェントが既に見つけた作業を繰り返す必要はありません。詳細については、「the section called “自律型インシデント対応”」を参照してください。

オンデマンド DevOps タスク

インシデントライフサイクル中はいつでも、会話型チャットインターフェイスを介して DevOps エージェントとやり取りできます。自然言語を使用して、 AWS リソース、システムの状態、アラームステータス、デプロイ履歴について質問します。チャットはコンテキスト対応です。特定の調査を表示している場合、エージェントを誘導して特定の仮説を探索したり、特定のログに集中したり、根本原因分析を更新したりできます。コンソール間を移動することなく、環境全体のリソース設定、エラー傾向、調査インサイトをクエリすることもできます。詳細については、「the section called “オンデマンド DevOps タスク”」を参照してください。

プロアクティブなインシデント防止

インシデントを解決した後、DevOps Agent は調査履歴全体のパターンを分析して、将来のインシデントを防ぎ、平均検出時間を短縮する推奨事項を生成します。推奨事項は、オブザーバビリティ体制、テストギャップ、コード変更、インフラストラクチャアーキテクチャの 4 つの領域に及びます。エージェントは毎週評価を実行し、新しいインシデントが発生したときにレコメンデーションを更新します。レコメンデーションを承認、拒否、または追跡できます。エージェントはフィードバッ

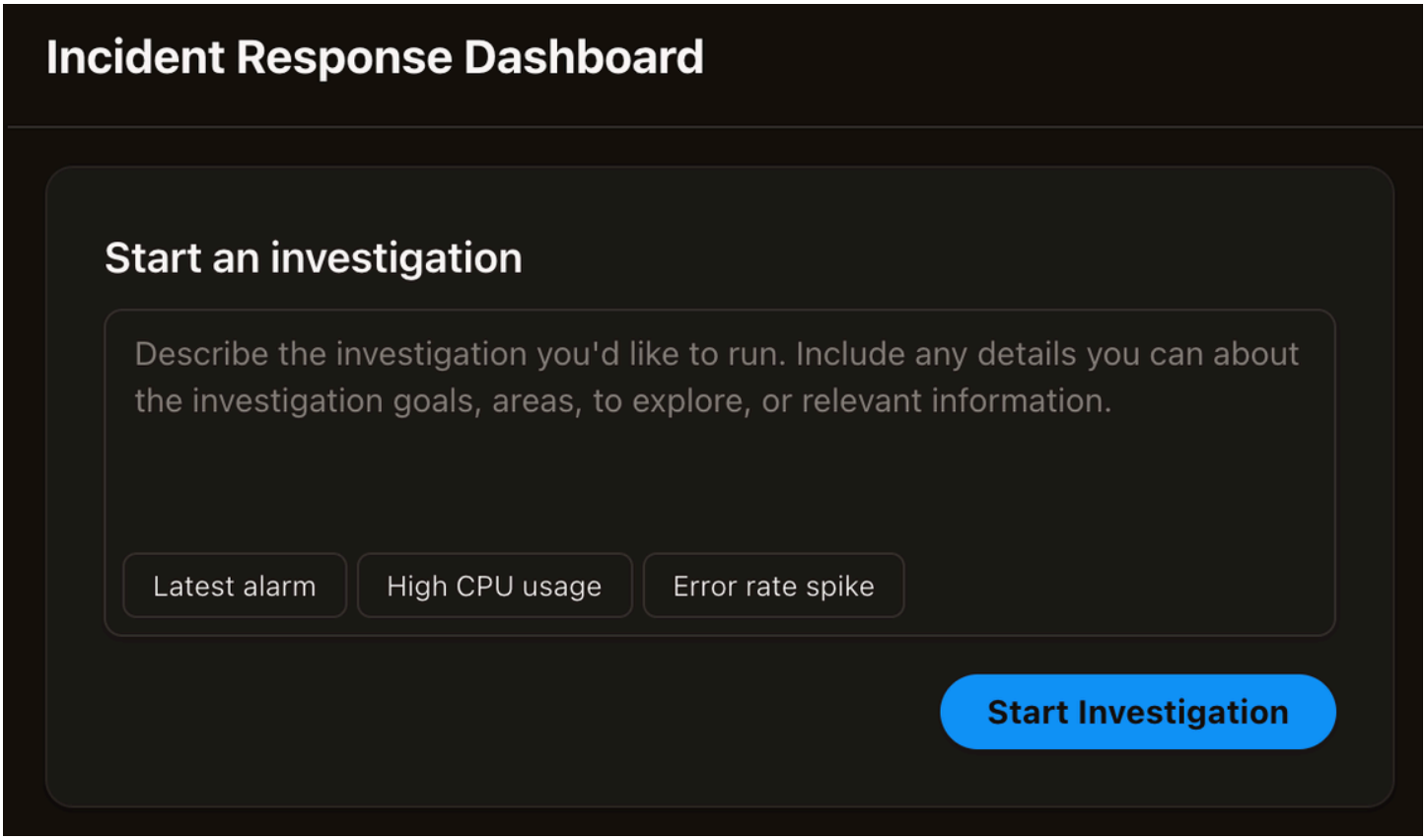
クから学び、今後の提案を絞り込みます。詳細については、「the section called “プロアクティブインシデント防止”」を参照してください。

自律型インシデント対応

調査の開始

インシデント対応調査は、3 つの方法のいずれかで開始できます。

- 組み込み統合 - 組み込み統合を使用して、DevOps エージェントスペースを ServiceNow などのチケット発行システムに接続できます。接続されると、DevOps Agent のインシデント対応調査はサポートチケットから自動的にトリガーされ、DevOps Agent は主要な検出結果、根本原因分析、緩和計画の更新を元のチケットに提供します。
- ウェブフック - ウェブフックを使用して、 AWS DevOps エージェントにイベントを送信できます。たとえば、ウェブフックを使用して PagerDuty チケットまたは Grafana アラームからインシデント対応調査をトリガーできます。
- 手動 - DevOps エージェントスペースウェブアプリケーションのインシデント対応タブからインシデント対応調査を手動で開始できます。DevOps エージェントが調査するインシデントを説明する自由形式のテキストを入力すると、調査計画の作成、結果の収集、根本原因の特定、緩和計画の生成の提案を行うことができます。いくつかの事前設定された開始点から選択して、調査をすばやく開始することもできます。 最後にトリガーされたアラームを調査し、基盤となるメトリクスとログを分析して根本原因を特定する最新のアラーム。 CPU 使用率が高く、コンピューティングリソース全体で CPU 使用率の高いメトリクスを調査し、過剰なリソースを消費しているプロセスやサービスを特定します。 またはエラー率の急増により、メトリクスを分析してアプリケーションエラー率の最近の増加を調査します。 アプリケーションログ、 と で障害の原因を特定します。



「調査の開始」をクリックすると、エージェントが作業に集中するのに役立つ追加の詳細を入力するように求められます。調査ダイアログには、次のフィールドが含まれます。

- 調査の詳細 – 説明があらかじめ入力されています。これを編集して調査範囲を絞り込むことができます。
- 調査の開始点 – 必要に応じて、エージェントの特定のアラーム、メトリクス、ログスニペット、またはその他の開始点を記述します。
- インシデントの日時 – 現在の時刻が UTC 形式で自動入力されます。インシデントが以前に発生したかどうかを調整します。
- 調査に名前を付ける – タイムスタンプ付きで自動生成されます。これをカスタマイズできます (最大 400 文字) 。
- Priority – ドロップダウンから調査優先度を選択します (Medium がデフォルトです) 。

必要に応じてこれらのフィールドを確認して調整し、「調査を開始する...」をクリックして開始します。その後、調査の詳細ページに移動し、DevOps エージェントの動作を確認できます。

インシデントのトリアージ

トリアージフェーズは、 AWS DevOps エージェントのインシデント対応システムの最初のステージです。Datadog からのアラーム、ServiceNow からのインシデントチケット、Dynatrace の問題などの外部イベントがトリガーされると、 AWS DevOps Agent はそれを数秒以内に自動的に処理して、個別に調査するか、既存の調査にリンクするかを判断します。

トリアージステージの主な機能はインシデント相関です。関連するインシデントを特定し、それらを 1 つの調査に統合して、重複する作業とリソースの無駄を回避します。新しいインシデントが到着すると、 AWS DevOps Agent はルックバックウィンドウ (通常は 20 分) 内でアクティブな調査とともに分析します。AI を活用した分析を使用して、コンポーネントの類似性、地理的リージョン、タイミングパターンなどの要因を調べ、インシデント間の関係を判断します。

AWS DevOps Agent は、次の 2 つの決定のいずれかを行います。

- リンク済み – インシデントを既存の調査に関連付け、新しいインシデントに関するコンテキストを含むステアリングメッセージをその調査に送信します。
- 続行 — インシデントの新しい独立した調査をスケジュールします。

トリアージ決定の表示

インシデントがリンクされると、プライマリ調査は、リンクされたインシデントの詳細と相関の推論を含むステアリングメッセージを受信します。 AWS DevOps エージェントスペースウェブアプリでは、インシデントがリンクされた理由を説明する相関推論とともに、LINKED のステータスが表示されます。プライマリ調査では、リンクされたすべてのインシデントのリストが表示され、一緒に調査されている関連する問題の全範囲を確認できます。外部チケットシステム (ServiceNow、PagerDuty など) と通信チャネル (Slack) は、相関推論とともにインシデントがリンクされたという通知を受け取ります。

インシデントとカスタム相関ルールのリンク解除

AWS DevOps Agent が誤ってインシデントを関連付けている場合は、 AWS DevOps Agent Space ウェブアプリを使用して手動でリンクを解除できます。これにより、リンクされていないインシデントが独立した調査として再スケジュールされます。相関ロジックを含む AWS DevOps エージェントスキルを作成し、トリアージステージに関連付けることで、 AWS DevOps エージェントをガイドするカスタム相関ルールを指定することもできます。

人間によるサポートを依頼する

AWS DevOps エージェントは AWS サポートに直接接続して、インシデント対応プロセスを合理化できます。 AWS サポートから追加のサポートが必要な場合は、DevOps Agent Space ウェブアプリからサポートケースを作成して、調査コンテキストをサポート AWS エンジニアと自動的に共有し、問題を説明するのに必要な時間を短縮できます。

仕組み

インシデントを調査する際、 AWS DevOps Agent は分析の包括的なログを構築します。これには以下が含まれます。

- 根本原因の調査結果
- 分析されたメトリクス、ログ、トレース
- コード変更とデプロイ履歴の確認
- 推奨される修復アクション
- イベントとシステム動作のタイムライン

AWS DevOps エージェントスペースウェブアプリから直接、調査を AWS サポートにエスカレーションできます。これを行うと、 AWS DevOps Agent は自動的に調査ログを AWS サポートに渡し、詳細を手動で収集して説明することなく、調査に関する完全なコンテキストをサポートエンジニアに提供します。

AWS サポートとのチャット

サポートケースを作成したら、 AWS DevOps エージェントスペースウェブアプリ内の別のチャットウィンドウで AWS サポートと通信できます。これにより、次のことが可能になります。

- AWS DevOps エージェントの調査タイムラインとともに AWS サポートエンジニアと問題について話し合う
- AWS DevOps Agent の自動分析と AWS Support のエキスパートガイダンスの両方を同じインターフェイスで表示する
- 必要に応じて追加情報や明確化をシームレスに共有する

チャットエクスペリエンスにより、 AWS DevOps Agent の調査と AWS サポートの会話に簡単にアクセスできるため、コラボレーションと解決が迅速になります。

サポートプランの要件

AWS DevOps エージェントを通じてサポートケースを作成して操作できるかどうかは、 AWS サポートプランによって異なります。使用権限の詳細については、 Support Plans ユーザーガイドを参照してください。

注 ベーシックサポートのお客様はテクニカルサポートケースを作成できないため、 AWS DevOps エージェントの調査を AWS Support Developer Support にエスカレーションすることはできません。したがって、お客様は AWS DevOps Agent を通じてケースを作成できますが、サポートセンターにアクセスしてサポートエンジニアAWSに対応する必要があります。デベロッパーサポートにはチャットベースのサポートが含まれていないためです。他のすべてのプランでは、 AWS DevOps Agent 内で統合されたチャットエクスペリエンスを使用できます。応答時間や利用可能なケース重要度など、サポートプランの使用権限の詳細については、AWS 「サポートプランユーザーガイド」を参照してください。

AWS サポートと共有される情報

AWS DevOps Agent Space ウェブアプリからサポートケースを作成すると、次の情報が AWS Support に自動的に共有されます。

- 調査タイムライン: AWS DevOps Agent の分析の時系列レコード
- リソース情報: 影響を受ける AWS リソース
- オブザーバビリティデータ: 統合されたモニタリングツールからの関連するメトリクス、ログ、トレース
- 最近の変更: コードのデプロイ、インフラストラクチャの変更、設定の更新
- 修復の試行: Actions AWS DevOps Agent を推奨
- 影響評価: インシデントの範囲と重大度

AWS サポートと共有されるすべてのデータは、既存の AWS データレジデンシーとセキュリティ設定に従います。 AWS DevOps Agent は、特定の調査に関連する情報のみを共有し、組織のデータガバナンスポリシーを尊重します。

開始方法

AWS DevOps Agent AWS のサポート統合を使用するには:

1. アクティブな AWS サポートプランがあることを確認します。

2. AWS DevOps エージェントの IAM アクセス許可にサポートケースの作成 (support:CreateCase、support:DescribeCases) が含まれていることを確認します。
3. AWS DevOps Agent が問題を調査していて、 AWS サポートサポートが必要な場合は、DevOps Agent Space ウェブアプリから人間によるサポートを依頼を選択します。
4. AWS サポートと共有される調査の概要を確認します。
5. サポートプランの使用権限に基づいて、適切なケース重要度を選択します。
6. ケースを送信する - AWS DevOps Agent には調査ログが自動的に含まれます。

チャットウィンドウが自動的に開き、すぐに AWS サポートとのコラボレーションを開始できます。

プロアクティブインシデント防止

AWS DevOps Agent は、インシデント調査全体のパターンを分析し、運用体制を継続的に改善し、将来のインシデントを防ぐターゲットを絞った推奨事項を提供します。Operator Web App の Ops Backlog ページからプロアクティブインシデント防止にアクセスします。

プロアクティブインシデント防止の仕組み

AWS DevOps エージェントは、最近のインシデント調査を評価して、今後のインシデントを防ぎ、平均検出時間 (MTTD) を短縮するための継続的な改善を特定します。エージェントは、複数のインシデントを分析して、将来のインシデントのクラス全体を妨げる可能性のある推奨事項を特定し、最も影響の大きい推奨事項に焦点を当てて、それらが実行可能であることを確認します。

デフォルトでは、エージェントは評価を毎週自動的に実行します。オンデマンドでのみ評価を実行する場合は、スケジュールを一時停止できます。手動評価は常に利用可能であり、最近の調査で推奨される改善を迅速に実施する必要がある場合に役立ちます。

エージェントは、「Ops Backlog」ページの「Recommendation Categorization」チャートに示されている 4 つのカテゴリにわたる改善点を特定します。

- オブザーバビリティ – モニタリング、アラート、ログ記録、システムの可視性を強化して、問題を迅速かつ正確に検出するための推奨事項。
- インフラストラクチャ — リソース設定、容量調整、アーキテクチャの耐障害性を最適化するための推奨事項。
- ガバナンス – デプロイプロセス、パイプラインの改善、テストプラクティス、運用コントロールを強化するための推奨事項。

- コードの最適化 – アプリケーションコードの品質、エラー処理、コードの耐障害性を向上させるための推奨事項。

この分類は、運用上の改善が最も必要な場所を理解し、チームの重点分野に基づいてレコメンデーションに優先順位を付けるのに役立ちます。

利点

- 繰り返し発生するインシデントの防止 – 同じタイプの問題に繰り返し対応するのではなく、根本原因に体系的に対処する
- 運用上の煩雑さの軽減 – チームが反復的な発砲から解放され、イノベーションと戦略的改善に集中できるようになります。
- システムの耐障害性の向上 – 実際のインシデントデータに基づいてインフラストラクチャ、オブザーバビリティ、デプロイプロセスを強化する
- 過去のパターンから学ぶ – 過去のインシデントからのインサイトを活用して、最も大きな影響を与えるターゲットを絞った改善を行います。

エージェントの概要

ウェブアプリの Ops Backlog ページのエージェント概要には、最近のインシデントの前回の評価の結果の説明が表示されます。概要では、分析されたインシデント調査の数、過去のインシデントと類似しているインシデント、新しい情報で作成または更新された推奨事項について説明します。

概要は、エージェントが最新の評価中に検出した内容をすばやく理解し、運用体制に最も影響を与える可能性のある最も注目すべき推奨事項を強調するのに役立ちます。

評価の制御

AWS DevOps Agent がインシデントを評価し、レコメンデーションを生成するタイミングを制御できます。

- 評価を手動で実行する – Ops Backlog ページの Run Now ボタンをクリックして、すぐに評価を開始します。これは、最近の調査で推奨される改善を迅速に実施する必要がある場合に便利です。
- アクティブな評価の停止 – Ops Backlog ページの評価の停止ボタンをクリックして、現在進行中の評価を停止します。

レコメンデーションの管理

AWS DevOps Agent は、Ops Backlog ページでレコメンデーションを表示および管理できます。

- レコメンデーションの詳細の表示 – レコメンデーションをクリックしてレコメンデーションの詳細ページを開きます。レコメンデーションの詳細ページには、レコメンデーションに通知したインシデント、予想される影響、次のステップなど、推奨される改善に関する詳細情報が表示されます。コード変更に関する推奨事項については、実装のためにコーディングエージェントに渡すことができるエージェント対応仕様を表示することもできます。
- キープ — 「キープ」をクリックして、追跡する推奨事項をバックログに保持します。これにより、実装する予定の改善点をモニタリングし、その進捗状況を追跡できます。
- 破棄 – 「破棄」をクリックして、バックログからレコメンデーションを削除します。レコメンデーションを破棄するときは、その理由を自然言語で説明できます。エージェントは、このフィードバックから学習し、それを使用して将来のレコメンデーションを知らせ、時間の経過とともに運用上の優先順位と要件により合わせるようにします。
- 実装済み – 「実装済み」をクリックして、レコメンデーションを完了済みとしてマークします。これにより、どの改善が適用されたかを追跡し、エージェントが時間の経過とともにレコメンデーションの有効性を測定できるようになります。
- 自動削除 – レコメンデーションを実装することで新しいインシデントを防ぐことができなかった場合、約 6 週間後にキープまたは実装としてマークされていないレコメンデーションを削除できます。これにより、Ops Backlog ページは、運用上の課題に最も関連性の高い改善に焦点を当てます。
- レコメンデーションの更新 – 既存のレコメンデーションは、レコメンデーションによって防止された新しいインシデントが見つかったときに更新されます。更新により、レコメンデーションの優先度が変更されたり、新しいインサイトに基づいてレコメンデーションが絞り込まれたりすることがあります。

エージェント対応仕様

コードまたは設定の変更に関する推奨事項については、 AWS DevOps Agent はエージェント対応仕様を生成できます。この仕様は、実装のためにコーディングエージェントに直接渡すことができる構造化ドキュメントを提供します。

仕様には以下が含まれます。

- 問題ステートメント – 問題とその根本原因の概要

- ソリューションの概要 — 推奨されるアプローチの概要の説明
- ターゲットリポジトリ – 変更が必要な特定のリポジトリ
- コード変更 – 変更が必要な内容と理由の詳細な説明と、特定のファイルパスと実装に関する考慮事項
- テスト要件 – テストする必要があるシナリオ
- 実装計画 – 変更を実装するための段階的なアプローチ

エージェント対応仕様は、コーディングエージェントに本番環境に対応した変更を行うために必要なコンテキストを提供することで、実装を高速化します。エンジニアとの広範なback-and-forthは必要ありません。

レコメンデーションの実装

プロアクティブなインシデント防止レコメンデーションの価値を最大化するには、それらに対応するための以下のプラクティスを検討してください。

- エージェント対応仕様の使用 – コード変更に関する推奨事項については、生成された仕様を使用して、コーディングエージェントに渡すか、手動実装の詳細なガイドとして使用して実装を高速化します。
- チケットバックログへのレコメンデーションの追加 – レコメンデーションをチームのチケットシステムまたはプロジェクト管理ツールにコピーして、他のエンジニアリング作業とともに優先されるようにします。
- 影響に基づいてレコメンデーションを優先する – 最も頻繁または重大なインシデントタイプ、または重要なシステムに影響するインシデントタイプに対応するレコメンデーションに重点を置きます。
- 実装の進行状況の追跡 – 実装された推奨事項を監視し、同様のインシデントが時間の経過とともに減少するかどうかを観察して、その有効性を測定します。
- 開発チームとの調整 – 影響を受けるシステムを所有する適切なチームとレコメンデーションを共有し、改善の実装に必要なコンテキストとリソースを確保します。

オンデマンド DevOps タスク

AWS DevOps エージェントオンデマンドタスクは、生成人工知能 (AI) を活用した会話型アシスタントです。これにより、運用チームはアプリケーションアーキテクチャのクエリ、システムの状態の分析、自然言語を使用した調査インサイトへのアクセスが可能になります。 AWS リソース、システ

ムメトリクス、アラームステータス、デプロイ履歴、インシデントパターンについて質問できます。チャットは、実際のインフラストラクチャと運用データに基づいて即座に回答を提供するため、複数の AWS コンソールやモニタリングツール間を移動する必要はありません。

チャットは DevOps エージェントスペースウェブアプリ全体に統合され、表示しているページに基づいてコンテキスト対応のレスポンスを提供します。インターフェイスは会話履歴を保持するため、以前の議論を続行し、以前のクエリを構築できます。

タスク機能

AWS DevOps エージェントオンデマンドタスクは、インフラストラクチャの管理と理解に役立つ包括的な機能を提供します。

リソースクエリ – Lambda 関数、DynamoDB テーブル、EKS デプロイ、証明書、インフラストラクチャ設定など、エージェントスペース内の AWS リソースについて質問します。チャットでは、ランタイムバージョン、容量設定、デプロイステータスなどの属性に基づいてリソースをフィルタリングおよび分析できます。たとえば、「Python 3.8 を使用している Lambda の数」と尋ねます。または「有効期限が切れる証明書はありますか？」

システムヘルス分析 – アラームステータス、エラー率、CPU 使用率、サービスの可用性など、現在および過去のシステムヘルスメトリクスをクエリします。チャットでは、特定の期間をカバーするヘルスサマリーを生成し、システム動作の傾向を特定できます。「過去 24 時間に発生したアラームはどれですか？」などの質問をします。または「過去 1 時間の 5xx エラーはありますか？」

調査インサイト – 根本原因分析、調査された仮説、レビューされたログ、解決パターンなど、完了した調査と進行中の調査からの情報にアクセスします。チャットでは、一般的なインシデントの原因を特定し、履歴データに基づいてレコメンデーションを提供できます。クエリ「先月のインシデントの最も一般的な原因は何ですか？」 または「完了した調査の平均解決時間を教えてください」

調査ステアリング – 調査の詳細ページを表示するときは、エージェントに特定のログに焦点を当てるように指示したり、特定の仮説を調べたり、根本原因分析を更新したりして、調査をガイドします。「支払いサービスのログに注目して RCA を更新する」やDynamoDB スロットリングが原因で問題が発生したという仮説を詳しく調べる」などのステアリング入力を提供します。

チャットアーティファクト – 運用状態の概要、エラーレポート、インシデント分析など、構造化されたレポートとドキュメントを生成します。アーティファクトは専用パネルに表示され、会話内のバージョニング編集をサポートします。

レコメンデーションフィルタリング – 特定のサービスや運用上の懸念に関連するレコメンデーションなど、特定の基準でインシデント防止レコメンデーションをクエリします。Chat では、各レコメ

ンデーションの影響と実装に関する考慮事項について説明します。例えば、DynamoDB に関連するインシデントを防ぐレコメンデーションを表示する」または「どのレコメンデーションがリクエストレイテンシーの問題をより迅速に検出するのに役立ちますか？」などです。

チャットへのアクセス

チャットは、DevOps エージェントスペースウェブアプリケーションの左側にある永続パネルとして使用できます。左側のサイドバーには、+ 新しいチャットボタン、インシデント、オペレーションバックログ、トポロジに移動するためのページセクション、最近の会話を表示するチャットセクションが含まれます。すべて表示を選択して、会話履歴全体を表示します。

チャットは、アクセスする場所に基づいてコンテキスト対応のレスポンスを提供します。

トポロジ – エージェントスペースのリソース、アーキテクチャ、運用状態に関する一般的な質問をします。チャットは、接続されたすべてのアカウントとサービスを完全に可視化します。このコンテキストから、リソース設定、デプロイ履歴、トポロジ情報、オブザーバビリティツールの統合をクエリできます。

インシデント対応 – インシデント対応ページを表示するときは、エージェントスペース全体の調査の傾向、解決時間、インシデントパターンについて質問します。チャットは過去の調査データを分析して、一般的な原因と改善の機会を特定できます。

調査の詳細 – 特定の調査を表示している間、チャットはその調査に関するコンテキスト対応応答を提供します。レビューされたログ、調査された仮説、根本原因の結論、緩和計画について質問します。調査の焦点を導き出すためにステアリング入力を指定することもできます。

防止 — 防止ページから、フィルターを使用してレコメンデーションをクエリし、レコメンデーションが行われた理由を理解し、実装アプローチを検討します。チャットは、インシデント防止の推奨事項の影響に優先順位を付け、理解するのに役立ちます。

ページを切り替えるとチャットインターフェイスは引き続き使用できますが、コンテキストは現在のビューに関連する情報を提供するように変更されます。新しい会話を開始すると、以前のコンテキストなしで会話が開始されます。既存の会話を続行すると、チャットはフォローアップの質問の会話履歴全体を保持します。

コンテキスト対応レスポンス

Chat は、DevOps エージェントスペースウェブアプリで表示しているページに基づいてレスポンスを調整します。このコンテキスト認識により、どの調査やリソーススコープについて質問するかを指定しなくても、関連情報を受け取ることができます。

調査の詳細ページを表示すると、チャットはその特定の調査について質問していることを自動的に理解します。「どのログを見ましたか？」などの質問 または「どの仮説を検討しましたか？」 は、現在表示されている調査を参照してください。ステアリング入力を指定すると、Chat はそれをアクティブな調査に適用し、必要に応じて新しい根本原因バージョンを作成します。

防止ページで、チャットはインシデント防止の推奨事項に関心があることを理解します。クエリは、エージェントスペースコンテキスト内でレコメンデーションを自動的にフィルタリングおよび分析します。システムは、一般的なレコメンデーションについて質問しているのか、特定のレコメンデーションの詳細について質問しているのかを認識します。

トポロジページからチャットにアクセスすると、チャットはエージェントスペース内のすべてのリソース、メトリクス、履歴データを広範囲に可視化します。調査やレコメンデーションのコンテキストを指定せずに、リソース、サービス、または運用上の懸念について質問できます。

このコンテキスト認識により、参照する調査、レコメンデーション、またはリソーススコープを繰り返し指定する必要がなくなり、より自然な会話フローが作成されます。

会話の管理をする

チャットは会話履歴を保持し、以前のディスカッションを続行して以前のクエリを参照できるようにします。

新しい会話の作成 – チャットパネルの「新しいセッション」ボタンをクリックして、以前のコンテキストなしで新しい会話を開始します。新しい会話では、以前のチャットからの情報は引き継がれず、混乱することなく無関係な質問をすることができます。

会話履歴へのアクセス –「履歴」をクリックすると、エージェントスペース内の以前の会話がすべて表示されます。会話は、タイムスタンプとプレビューテキストを使用して時系列的に整理されます。会話履歴は 90 日間保持され、エージェントスペース内のユーザーアカウントに対してプライベートです。

会話の継続 – 中断した場所を再開するには、履歴から会話を選択します。チャットは以前のメッセージの完全なコンテキストを維持し、会話の前半を参照するフォローアップの質問をすることができます。会話の表示中にページを切り替えると、会話コンテキストは残りますが、現在の場所に基づいてページ固有のコンテキストが更新されます。

会話履歴は各エージェントスペース内で分離されることに注意してください。1 つのエージェントスペースの会話は、他のエージェントスペースからは表示またはアクセスできません。この分離により、機密情報は組織の境界に従って分割されたままになります。

アーティファクトの生成

AWS DevOps エージェントは、会話中にエージェントによって生成された構造化されたバージョン管理されたドキュメントであるチャットアーティファクトをサポートします。アーティファクトは、オペレーションレポート、エラー概要、ヘルス評価など、AI が生成したコンテンツを確認および編集するための専用のインタラクティブパネルをチャット UI に提供します。

DevOps エージェントスペースウェブアプリの任意のページからアーティファクトをリクエストできます。チャットは、現在のページコンテキストを使用してアーティファクトコンテンツをスコープします。

アーティファクトの仕組み

Chat にコンテンツの作成または更新を依頼すると、Chat はアーティファクト、通常はフォーマットされたドキュメントを生成し、会話とともにアーティファクトパネルに表示します。

Generate – 自然言語リクエストを送信して、レポートまたはドキュメントを作成します。たとえば、「エージェントスペースの週次オペレーションヘルスレポートを生成する」または「先週の 4xx エラーのレポートを表示する」と尋ねます。

レビュー – アーティファクトは会話とともに専用パネルに表示されます。チャットの操作を継続しながら、コンテンツ全体を確認できます。

編集 – Chat を使用してアーティファクトへの変更をリクエストします。たとえば、「Lambda コールドスタートにセクションを追加する」または「先月のデータを含めるようにレポートを更新する」と尋ねます。Chat は、リクエストされた変更を含むアーティファクトの新しいバージョンを作成します。

サンプルクエリ

次の例は、チャットに質問できる質問のタイプを示しています。これらの例は、ユースケースとコンテキスト別に整理されています。

アーティファクト生成クエリ

DevOps エージェントスペースウェブアプリの任意のページから:

- エージェントスペースの毎週の運用状態の概要を生成する
- 先週のすべての 4xx エラーのレポートを作成する

- 過去 30 日間のインシデント概要レポートを作成する
- 今週の支払いサービスのアラームアクティビティの概要を作成する
- 過去 7 日間のデプロイ履歴レポートを生成する
- すべてのオープンレコメンデーションをレポートにまとめる

リソース情報クエリ

DevOps エージェントスペースウェブアプリの任意のページから:

- Python 3.8 を使用している Lambda 関数の数
- 有効期限が切れる証明書はありますか?
- オンデマンド請求ですべての DynamoDB テーブルを一覧表示する
- 本番環境で EKS クラスターを表示する
- 過去 90 日間にデプロイされていない Lambda 関数はどれですか?
- バージョニングが有効になっていない S3 バケットを一覧表示する
- データベースバージョン X を実行している RDS インスタンス

システムヘルスクエリ

トポロジまたはインシデント対応ページから:

- 過去 24 時間に発生したアラームはどれですか?
- 過去 1 時間に 5xx エラーがありましたか?
- 支払いサービスの Lambda エラーの傾向を表示する
- ECS クラスターの CPU 使用率を教えてください。
- ロードバランサーに異常なターゲットはありますか?
- 昨日の API Gateway スロットリングイベントを表示する
- 先週エラー率が最も高いのはどのサービスですか?
- 過去 24 時間に関する全体的なヘルスレポートを提供する

オブザーバビリティツールクエリ

トポロジから:

- Splunk ロググループを一覧表示する
- Prometheus メトリクスとそのアラームしきい値を表示する
- このサービスにはどのような Datadog モニターが設定されていますか?
- New Relic アラートポリシーを一覧表示する
- Dynatrace ダッシュボード設定を表示する

調査インサイトクエリ

インシデント対応ページから:

- 先月のインシデントの最も一般的な原因は何ですか?
- 完了した調査の平均解決時間を教えてください。
- 先週の調査とその RCA を要約する
- DynamoDB スロットリングによって発生したインシデントの数
- 過去 4 四半期の調査傾向を表示する
- 最も頻繁に発生するインシデントがあるのはどのサービスですか?

調査詳細クエリ

調査の詳細ページから:

- どのログを調べましたか?
- どの仮説を検討しましたか?
- 提案する緩和アクションはどの程度リスクがありますか?
- このインシデント中のイベントのタイムラインは?
- これが根本原因であると判断したのはなぜですか?
- 根本原因分析を裏付ける証拠は何ですか?
- 調査中に誰がステアリングを提供しましたか?
- このインシデント調査の概要を教えてください

調査ステアリングクエリ

調査の詳細ページから:

- 14:00 ~ 15:00 UTC の支払いサービスのログに焦点を当て、RCA を更新する
- DynamoDB スロットリングが問題の原因であったという仮説を調べる
- ECS クラスター設定をチェックして、アラームの原因となったかどうかを確認します。
- 丸 1 日ではなく、過去 2 時間のログのみをチェックする
- エラーの急増を午後 3 時に調査する
- Lambda ログではなく API Gateway ログを確認する

予防レコメンデーションクエリ

防止ページから:

- インシデント防止に関する上位 3 つの推奨事項は何ですか?
- DynamoDB に関連するインシデントを防ぐレコメンデーションを表示する
- リクエストのレイテンシーの問題をより迅速に検出するのに役立つ推奨事項はどれですか?
- 同様のインシデントを防ぐ可能性のあるオブザーバビリティの改善点を一覧表示する
- 支払いサービスのインフラストラクチャレコメンデーションを表示する
- システムの耐障害性に最も大きな影響を与える推奨事項はどれですか?

エージェントスペースでのチャットの有効化

チャットは、すべての DevOps エージェントスペースウェブアプリケーションで利用できます。セットアッププロセスは、新規または既存のエージェントスペースがあるかどうかによって異なります。

新しいエージェントスペース

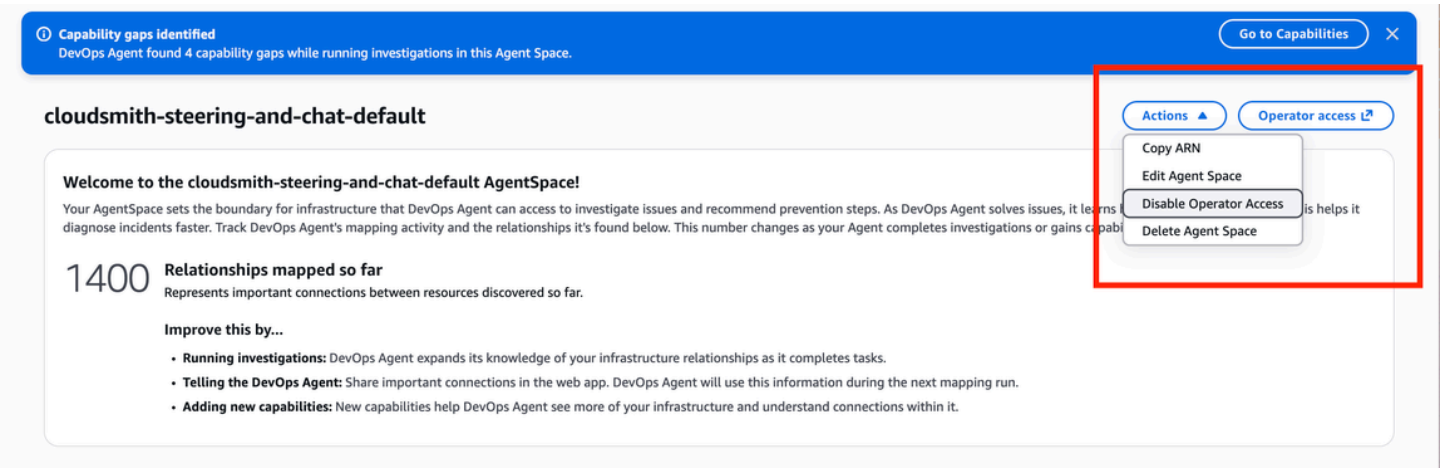
チャットは、新しいエージェントスペースを作成すると自動的に有効になります。追加の設定や IAM アクセス許可の設定は必要ありません。DevOps エージェントスペースウェブアプリを設定すると、チャットは任意のページの左側にある永続パネルとしてすぐに利用できるようになります。

既存のエージェントスペース

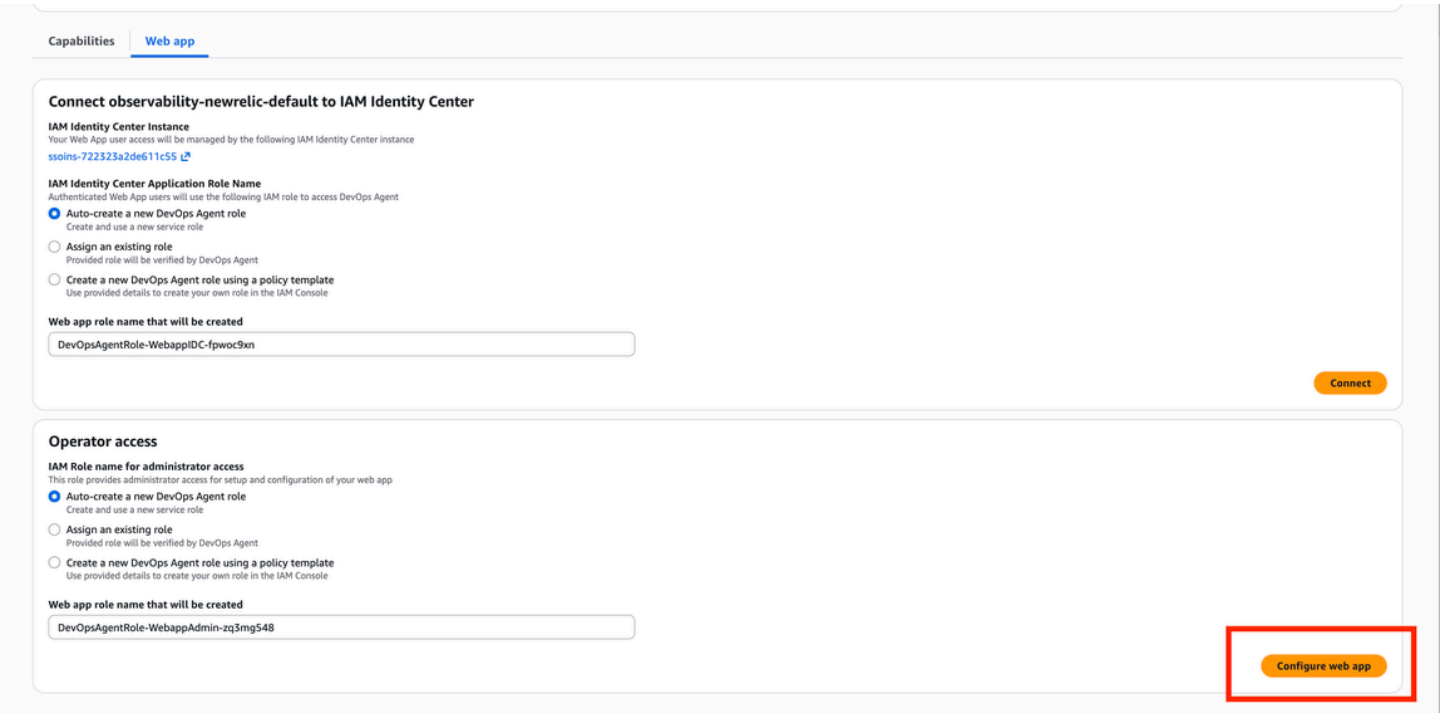
チャットがリリースされる前にエージェントスペースを作成した場合は、必要な IAM アクセス許可を有効にする必要があります。これには 2 つのオプションがあります。

オプション 1: オペレーターアプリへのアクセスを取り消して再度有効にする

AWS DevOps エージェント管理コンソールに移動し、右上隅にあるアクションドロップダウンを見つけ、現在のオペレーターアクセス設定を無効にします。



次に、オペレータアクセスの自動作成オプションを有効にします。



これにより、チャットに必要な IAM アクセス許可と、他のすべての現在のオペレーターアクセス許可が自動的に適用されます。

オプション 2: IAM アクセス許可を手動で追加する

既存のオペレーターアクセスロールに次の IAM アクセス許可を追加します。

- `aidevops:ListChats` – チャット会話履歴を表示する

- `aidevops:CreateChat` – 新しいチャット会話を作成する
- `aidevops:SendMessage` – メッセージの送信とレスポンスの受信

IAM AWS コンソールに移動し、DevOps エージェントオペレーターロールを見つけ、これらのアクセス許可をロールポリシーに追加します。チャットは、アクセス許可が追加された直後に利用可能になります。

いずれかのオプションを完了したら、DevOps Agent Space ウェブアプリを更新すると、チャットパネルがページの左側に表示されます。

AWS DevOps Agent の機能の設定

AWS DevOps エージェント機能は、エージェントの機能を既存のツールやインフラストラクチャに接続することで拡張します。これらの機能を設定して、包括的なインシデント調査、自動対応ワークフロー、DevOps エコシステムとのシームレスな統合を可能にします。

以下の機能は、DevOps エージェントの有効性を最大化するのに役立ちます。

- AWS EKS アクセスセットアップ - パブリック EKS 環境とプライベート EKS 環境の両方で、Kubernetes クラスター、ポッドログ、クラスターイベントのイントロスペクションを有効にします。
- Azure 統合 - Azure サブスクリプションと Azure DevOps 組織を接続して Azure リソースを調査し、Azure DevOps デプロイをインシデントに関連付ける
- CI/CD パイプライン統合 - GitHub パイプラインと GitLab パイプラインを接続してデプロイをインシデントと関連付け、調査中のコード変更を追跡する
- MCP サーバー接続 - Model Context Protocol を使用して外部オブザーバビリティツールとカスタムモニタリングシステムを接続することで調査機能を拡張
- マルチアカウント AWS アクセス - インシデント対応中に組織全体のリソースを調査するようにセカンダリ AWS アカウントを設定する
- Telemetry Source Integration - Datadog、Dynatrace、Grafana、New Relic、Splunk などのモニタリングプラットフォームを接続して、包括的なオブザーバビリティデータアクセスを実現
- チケット作成とチャット統合 - ServiceNow、PagerDuty、Slack を接続してインシデント対応ワークフローを自動化し、チームのコラボレーションを可能にする
- Webhook 設定 - 外部システムが HTTP リクエストを通じて DevOps エージェント調査を自動的にトリガーできるようにします。
- Amazon EventBridge 統合 - 調査と緩和ライフサイクルイベントを Amazon EventBridge ターゲットにルーティングすることで、 AWS DevOps エージェントをイベント駆動型アプリケーションに組み込みます。

チーム固有のニーズと既存のツールスタックに基づいて、各機能を個別に設定できます。インシデント対応ワークフローにとって最も重要な統合から開始し、必要に応じて追加機能に拡張します。

パブリックプレビューから一般提供への移行

パブリックプレビュー中に AWS DevOps エージェントを使用した場合は、GA リリース前に IAM ロールを更新する必要があります。このガイドでは、アカウントのモニタリングロールとオペレーターロールの更新について説明します。

変更点

1. プレビュー中のオンデマンドチャット履歴にアクセスできなくなります
2. 新しい マネージドポリシーは、プレビュー中に利用可能なポリシーを置き換えます
3. エージェントスペースには、古い IAM Identity Center アプリケーションアクセススコープがある可能性があります

パブリックプレビューからのオンデマンドチャット履歴

GA リリースでは、チャット履歴のアクセスコントロールを強化するための追加のセキュリティ対策が導入されています。これらの変更により、パブリックプレビュー期間 (2026 年 3 月 30 日以前) のオンデマンドチャット履歴にアクセスできなくなります。公開プレビュー中に作成された調査ジャーナルや調査結果は影響を受けません。この変更は、オンデマンドチャット会話にのみ適用されます。

新しい管理ポリシー

GA の場合、 はプレビュー時代のポリシーを置き換える新しい管理ポリシー AWS を提供します。

ロールタイプ	削除	Add
モニタリング	`AIOpsAssistantPolicy` マネージドポリシー	`AIDevOpsAgentAccessPolicy` マネージドポリシー
オペレーター (IAM および IDC)	インラインポリシー	`AIDevOpsOperatorAppAccessPolicy` マネージドポリシー

さらに、オペレータロールには更新された信頼ポリシーが必要であり、IDC オペレータロールには新しいインラインポリシーが必要です。

前提条件

- DevOps エージェントロールが設定されている AWS アカウントへのアクセス (プライマリアカウントとすべてのセカンダリアカウント)
- ロール、ポリシー、信頼関係を変更する IAM アクセス許可
- エージェントスペース ID、 AWS アカウント ID、リージョン (DevOps エージェントコンソールで表示)

ステップ 1: モニタリングロールを更新する

プライマリアカウントと各セカンダリアカウントのモニタリングロールを更新します。これらは、エージェントスペースの Capabilities タブで設定されたプライマリ/セカンダリソースロールです (プライマリ/セカンダリロールの例: `DevOpsAgentRole-AgentSpace-3xj2396z`)。

1. DevOps エージェントコンソールで、エージェントスペースに移動し、機能タブを選択します。
2. プライマリ/セカンダリソース (などDevOpsAgentRole-AgentSpace-3xj2396z) のモニタリングロールを見つけ、編集を選択します。
3. アクセス許可ポリシーで、 AIOpsAssistantPolicy AWS 管理ポリシーを削除します。
4. アクセス許可の追加、ポリシーのアタッチ、AIDevOpsAgentAccessPolicy管理ポリシーのアタッチを選択します。
5. インラインポリシーを編集し、その内容を以下に置き換えて、アカウント ID を置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCreateServiceLinkedRoles",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": [
                "arn:aws:iam::<account-id>:role/aws-service-role/resource-
explorer-2.amazonaws.com/AWSServiceRoleForResourceExplorer"
            ]
        }
    ]
```

```
}
```

1. モニタリングロールの信頼ポリシーは変更を必要としません。以下と一致することを確認します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "aidevops.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<account-id>"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:aidevops:<region>:<account-
id>:agentspace/*"
                }
            }
        }
    ]
}
```

- 各セカンダリアカウントのモニタリングロールに対してステップ 2～6 を繰り返します。

ステップ 2: オペレーターロールを更新する (IAM)

1. DevOps エージェントコンソールで、アクセスタブを選択し、オペレーターロールを見つけます。
2. IAM コンソールで、オペレーターロールから既存のインラインポリシーを削除します。
3. アクセス許可の追加、ポリシーのアタッチ、`AIDevOpsOperatorAppAccessPolicy`管理ポリシーのアタッチを選択します。
4. 信頼関係タブを選択し、信頼ポリシーの編集を選択します。信頼ポリシーを以下に置き換え、アカウント ID、リージョン、エージェントスペース ID を置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "aidevops.amazonaws.com"
            },
            "Action": ["sts:AssumeRole", "sts:TagSession"],
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<account-id>"
                },
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:aidevops:<region>:<account-
id>:agentspace/<agentspace-id>"
                }
            }
        }
    ]
}
```

ステップ 3: オペレーターロールを更新する (IDC)

DevOps エージェントで IAM Identity Center を使用する場合は、各 IDC オペレーターロールを更新します。

1. IAM コンソールで、 ロールに移動し、 を検索`WebappIDC`して DevOps エージェント IDC ロール (など) を検索します`DevOpsAgentRole-WebappIDC-<id>`。
2. 各 IDC ロールについて:

a. 既存のインラインポリシーを削除します。

b. アクセス許可の追加、ポリシーのアタッチ、`AIDevOpsOperatorAppAccessPolicy`管理ポリシーのアタッチを選択します。

c。信頼関係タブを選択し、信頼ポリシーの編集を選択します。信頼ポリシーを以下に置き換え、アカウント ID、リージョン、エージェントスペース ID を置き換えます。

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "aidevops.amazonaws.com"
            },
            "Action": ["sts:AssumeRole", "sts:TagSession"],
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<account-id>"
                },
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:aidevops:<region>:<account-
id>:agentspace/<agentspace-id>"
                }
            }
        },
        {
            "Sid": "TrustedIdentityPropagation",
            "Effect": "Allow",
            "Principal": {
                "Service": "aidevops.amazonaws.com"
            },
            "Action": "sts:SetContext",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<account-id>"
                },
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:aidevops:<region>:<account-
id>:agentspace/<agentspace-id>"
                },
                "ForAllValues:ArnEquals": {
                    "sts:RequestContextProviders": [
                        "arn:aws:iam::aws:contextProvider/IdentityCenter"
                    ]
                },
                "Null": {
                    "sts:RequestContextProviders": "false"
                }
            }
        }
    ]
```

```
}
```

d。アカウント ID を置き換えて、次のアクセス許可を持つ新しいインラインポリシーを作成します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowDevOpsAgentSSOAccess",
            "Effect": "Allow",
            "Action": [
                "sso:ListInstances",
                "sso:DescribeInstance"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowDevOpsAgentIDCUserAccess",
            "Effect": "Allow",
            "Action": "identitystore:DescribeUser",
            "Resource": [
                "arn:aws:identitystore::<account-id>:identitystore/*",
                "arn:aws:identitystore:::user/*"
            ]
        }
    ]
}
```

IAM アイデンティティセンターを再接続する (該当する場合)

パブリックプレビュー中に作成されたエージェントスペースには、IAM Identity Center アプリケーションが古いアクセススコープで設定されている場合があります。GA の場合、正しいスコープは です**aidevops:read_write**。IAM Identity Center アプリケーションに以前のスコープ (**awsaidevops:read_write**) がある場合は、IAM Identity Center を切断して再接続する必要があります。

IAM Identity Center アプリケーションの範囲を確認する方法

次の AWS CLI コマンドを実行して、IAM Identity Center アプリケーションのスコープを確認します。アプリケーション ARN は、「アプリケーション」の IAM Identity Center コンソールにあります。

```
aws sso-admin list-application-access-scopes \
  --application-arn arn:aws:sso::<account-id>:application/<instance-id>/<application-id>
```

出力には正しいスコープ が表示されます**aidevops:read_write**。

```
{
    "Scopes": [
        {
            "Scope": "aidevops:read_write"
        }
    ]
}
```

スコープに が表示されている場合は**awsaidevops:read_write**、古いものです。以下の手順に従って更新します。

IAM Identity Center を再接続する方法

AWS マネージド IAM アイデンティティセンターアプリケーションのアクセススコープを直接更新することはできません。切断して再接続する必要があります。

1. AWS DevOps エージェントコンソールで、エージェントスペースに移動し、アクセスタブを選択します。
2. IAM Identity Center 設定の横にある切断を選択します。
3. 切断を確認します。
4. Connect を選択して、IAM Identity Center を再度セットアップします。サービスは、正しいスコープを持つ新しい IAM Identity Center アプリケーションを作成します。
5. IAM Identity Center コンソールでユーザーとグループを新しいアプリケーションに再割り当てします。

> **Important**
>
> 切断すると、IAM Identity Center ユーザーアカウントに関連付けられた個々のユーザーチャットとアーティファクト履歴が削除されます。ユーザーは再接続後に再度ログインする必要があります。

検証

すべてのステップを完了した後:

1. DevOps エージェントコンソールに戻り、エージェントスペースアクセスタブにアクセス許可エラーが表示されないことを確認します。
2. オペレーターウェブアプリをテストして、正しくロードおよび機能することを確認します。
3. IDC を使用する場合は、ユーザーがオペレーターエクスペリエンスを認証してアクセスできることを確認します。

トラブルシューティング

移行後のアクセス許可拒否エラー

- `AIOpsAssistantPolicy` が削除され、モニタリングロールにアタッチ`AIDevOpsAgentAccessPolicy`されていることを確認します。
- 古いインラインポリシーが削除され、オペレーターロールに`AIDevOpsOperatorAppAccessPolicy`アタッチされていることを確認します。
- 演算子の信頼ポリシーに が含まれていることを確認します`sts:TagSession`。
- すべてのプレースホルダー値 (`<account-id>`、`<region>`、`<agentspace-id>`) を実際の値に置き換えたことを確認します。

セカンダリアカウントが機能しない

- 各セカンダリアカウントのモニタリングロールは個別に更新する必要があります。各アカウントにログインし、ステップ 1 を繰り返します。

IDC 認証の失敗

- IDC 信頼ポリシーに `sts:AssumeRole/sts:TagSession` ステートメントと `TrustedIdentityPropagation`ステートメントの両方が含まれていることを確認します。
- インラインポリシーが `sso:ListInstances`、`sso:DescribeInstance`、および で作成された`identitystore:DescribeUser`ことを確認します。

移行後にオンデマンドチャット履歴が欠落している

- GA リリース後は、パブリックプレビュー期間のオンデマンドチャット履歴にアクセスできません。これは、GA で導入されたセキュリティ対策の強化による予想される動作です。調査ジャーナルやパブリックプレビューの結果は影響を受けません。

AWS EKS アクセス設定

AWS DevOps Agent は、パブリッククラスターとプライベートクラスターの両方に対して読み取り専用`kubectl`コマンドを実行して、Amazon EKS クラスターの問題を調査することができます。任意の数の EKS クラスターを同じエージェントスペースに接続できます。

接続すると、エージェントはリソースの説明、ポッドログの取得、クラスターイベントの検査、ノードの状態の確認など、クラスターの運用上の問題の診断に役立ちます。エージェントは、クラスター内のリソースを作成、変更、または削除することはできません。

前提条件

EKS アクセスを設定する前に、EKS クラスターの認証モードに EKS API が含まれていることを確認してください。これは、Amazon EKS コンソールのアクセスタブで確認できます。モードに EKS API が含まれていない場合は、続行する前に が実行するモードを選択します。

セットアップ

これらのステップは、アクセスエントリを作成するクラスターごとに Amazon EKS コンソールから完了する必要があります。IAM ロール ARN は、エージェントスペース (「」を参照the section called “エージェントスペースの作成”) の「機能 > クラウド > プライマリソース > 編集」にあります。

1. アクセスタブに移動します。認証モードがすでに EKS API と表示されている場合は、アクセスエントリを追加できます。それ以外の場合は、EKS API を含むモードを選択します。

2. アクセスタブから、新しい IAM アクセスエントリを作成します。プライマリクラウドソースの IAM ロール ARN をコピーし、アクセスエントリの IAM プリンシパルとして入力します。[次へ] をクリックします。
3. Managed AWS AmazonAIOpsAssistantPolicy アクセスポリシーを選択し、アクセススコープのクラスターを選択します。(または、エージェントが特定の名前空間にのみアクセスできるようにする場合は、目的の Kubernetes 名前空間を選択します) 。ポリシーの追加をクリックし、次へをクリックします。
4. 変更を確認し、正しいアクセスエントリポリシーと IAM ロールが選択されていることを確認し、「作成」をクリックしてアクセスエントリを作成します。

EKS アクセスが正しく設定されていることを確認するには、オペレーターアプリに移動して新しい調査を開始し、「デフォルトの名前空間にすべてのポッドを一覧表示する」や「クラスターに最近のイベントを表示する」など、クラスターについてエージェントに質問します。

トラブルシューティング

エージェントがクラスターに到達できない場合は、セットアップダイアログに表示される正しい IAM ロール ARN がアクセスエントリで使用されており、AmazonAIOpsAssistantPolicy アクセスポリシーがアタッチされていることを確認します。

Azure の接続

Azure 統合により、 AWS DevOps エージェントは Azure 環境内のリソースを調査し、Azure DevOps パイプラインのデプロイを運用上のインシデントに関連付けることができます。Azure を接続することで、エージェントは Azure インフラストラクチャを可視化し、 AWS と Azure リソースの両方で根本原因分析を実行できます。

Azure 統合は、2 つの独立した機能で構成されています。

- Azure リソース – エージェントが仮想マシン、Azure Kubernetes Service (AKS) クラスター、データベース、ネットワークコンポーネントなどの Azure クラウドリソースを検出して調査できるようにします。エージェントは Azure リソースグラフを使用して、インシデント調査中にリソースをクエリします。
- Azure DevOps – エージェントが Azure DevOps リポジトリとパイプライン実行履歴にアクセスできるようにします。エージェントは、コードの変更とデプロイをインシデントと関連付けて、潜在的な根本原因を特定するのに役立ちます。

各機能は AWS アカウントレベルで登録され、個々のエージェントスペースに関連付けることができます。

登録方法

AWS DevOps Agent は、Azure に接続するための 2 つの方法をサポートしています。

- 管理者の同意 – Azure テナントで AWS DevOps Agent Entra アプリケーションを承認する、合理化された同意ベースのフロー。コンソールでは、これは管理者の同意オプションとして表示されます。この方法では、Microsoft Entra ID で管理者の同意を実行する権限を持つアカウントでサインインする必要があります。
- アプリケーション登録 – アウトバウンド ID フェデレーションを使用してフェデレーション ID 認証情報を使用して独自の Entra アプリケーションを作成するセルフマネージド型アプローチ。コンソールでは、これはアプリ登録オプションとして表示されます。この方法は、アプリケーション設定をより詳細に制御する必要がある場合や、管理者の同意許可がない場合に適しています。

どちらの方法も同じ機能を提供します。同じ AWS アカウント内で 1 つまたは両方の方法を使用できます。

既知の制限事項

- 管理者の同意: Azure テナントごとに 1 つの AWS アカウント – 各 Azure テナントは、DevOps AWS DevOps Agent Entra App を一度に 1 つの AWS アカウントにのみ関連付けることができます。同じテナントを別の AWS アカウントに関連付けるには、まず既存の登録を解除する必要があります。
- アプリ登録: 登録ごとに一意のアプリケーション – アプリ登録ごとに異なるアプリケーション (クライアント ID) を使用する必要があります。同じクライアント ID で複数の設定を登録することはできません。
- Azure DevOps: ソースコードアクセス – Azure DevOps 統合は、ソースコードがホストされている場所に関係なく、パイプライン実行履歴へのアクセスを提供します。ただし、実際のソースコードにアクセスするには、サポートされているソースプロバイダー (など) を介してリポジトリを個別に接続する必要がありますthe section called “GitHub の接続”。Bitbucket でホストされているソースコードは、Azure DevOps 統合から直接アクセスすることはできません。

トピック

- the section called “Azure リソースの接続”

- the section called “Azure DevOps の接続”

Azure リソースの接続

Azure リソースの統合により、 AWS DevOps Agent はインシデント調査中に Azure サブスクリプション内のリソースを検出して調査できます。エージェントは、リソース検出に Azure リソースグラフを使用し、Azure 環境全体のメトリクス、ログ、および設定データにアクセスできます。

この統合は、 AWS アカウントレベルで Azure を登録し、特定の Azure サブスクリプションを個々のエージェントスペースに関連付けるという 2 つのステップのプロセスに従います。

前提条件

Azure リソースを接続する前に、以下を確認してください。

- AWS DevOps エージェントコンソールへのアクセス
- ターゲットサブスクリプションにアクセスできる Azure アカウント
- 管理者の同意方法: Microsoft Entra ID で管理者の同意を実行するアクセス許可を持つアカウント
- アプリ登録方法の場合: フェデレーティッド ID 認証情報を設定するアクセス許可を持ち、 AWS アカウントでアウトバウンド ID フェデレーションが有効になっている Entra アプリケーション

> 注: エージェントスペース内から登録を開始することもできます。セカンダリソースに移動し、追加をクリックして Azure を選択します。Azure Cloud がまだ登録されていない場合は、コンソールが最初に登録を案内します。

管理者の同意による Azure リソースの登録

管理者同意メソッドは、 AWS DevOps Agent マネージドアプリケーションで同意ベースのフローを使用します。

ステップ 1: 登録を開始する

1. AWS マネジメントコンソールにサインインし、 AWS DevOps エージェントコンソールに移動します。
2. 機能プロバイダーページに移動する
3. Azure Cloud セクションを見つけて Register をクリックします。
4. 管理者の同意登録方法を選択する

ステップ 2: 管理者の同意を完了する

1. リクエストされているアクセス許可を確認する
2. クリックして続行 — Microsoft Entra 管理者同意ページにリダイレクトされます
3. 管理者の同意を実行するアクセス許可を持つユーザープリンシパルアカウントでサインインする
4. AWS DevOps Agent アプリケーションを確認して同意する

ステップ 3: ユーザー認可を完了する

1. 管理者の同意後、承認されたテナントのメンバーとして ID を検証するユーザー認可を求められます。
2. 同じ Azure テナントに属するアカウントでサインインする
3. 認可後、成功ステータスで AWS DevOps エージェントコンソールにリダイレクトされます。

ステップ 4: ロールを割り当てる

以下の Azure ロールの割り当てを参照してください。メンバーを選択するときに AWS DevOps エージェントを検索します。

アプリ登録による Azure リソースの登録

アプリ登録メソッドは、フェデレーティッド ID 認証情報を使用して独自の Entra アプリケーションを使用します。

ステップ 1: 登録を開始する

1. AWS DevOps エージェントコンソールで、機能プロバイダーページに移動します。
2. Azure Cloud セクションを見つけて Register をクリックします。
3. アプリ登録方法を選択する

ステップ 2: Entra アプリケーションを作成して設定する

コンソールに表示される手順に従って、次の操作を行います。

1. AWS アカウントでアウトバウンド ID フェデレーションを有効にする (IAM コンソールで、アカウント設定 → アウトバウンド ID フェデレーションに移動)

2. Microsoft Entra ID で Entra アプリケーションを作成するか、既存のアプリケーションを使用します。
3. アプリケーションでフェデレーション ID 認証情報を設定する

ステップ 3: 登録の詳細を入力する

登録フォームに以下を入力します。

- テナント ID – Azure テナント識別子
- テナント名 – テナントの表示名
- クライアント ID – 作成した Entra アプリケーションのアプリケーション (クライアント) ID
- 対象者 – フェデレーティッド認証情報の対象者識別子

ステップ 4: IAM ロールを作成する

コンソールから登録を送信すると、IAM ロールが自動的に作成されます。これにより、 AWS DevOps エージェントは認証情報を引き受けて を呼び出すことができますsts:GetWebIdentityToken。

ステップ 5: ロールを割り当てる

以下の「Azure ロールの割り当て」を参照してください。メンバーの選択時に作成した Entra アプリケーションを検索します。

ステップ 6: 登録を完了する

1. AWS DevOps エージェントコンソールで設定を確認する
2. 送信をクリックして登録を完了します

Azure ロールの割り当て

登録後、Azure サブスクリプションへの読み取りアクセスをアプリケーションに付与します。このステップは、管理者の同意方法とアプリ登録方法の両方で同じです。

1. Azure Portal で、ターゲットサブスクリプションに移動します。
2. アクセスコントロール (IAM) に移動する
3. Add > Add role assignment をクリックします。

4. リーダーロールを選択し、次へ をクリックします。
5. メンバーの選択をクリックし、アプリケーション (管理者の同意のための AWS DevOps エージェント、またはアプリ登録のための独自の Entra アプリケーション) を検索します。
6. アプリケーションを選択し、レビュー + 割り当て をクリックします。
7. (オプション) エージェントが Azure Kubernetes Service (AKS) クラスターにアクセスできるようにするには、次の AKS アクセス設定を完了します。

> セキュリティ要件: サービスプリンシパルには、リーダーロール (およびオプションで以下に示す AKS 読み取り専用ロール) のみを割り当てる必要があります。リーダーロールは、エージェントを読み取り専用オペレーションに制限し、間接的なプロンプトインジェクション攻撃の影響を制限するセキュリティ境界として機能します。書き込みまたはアクションのアクセス許可を持つロールを割り当てると、プロンプトインジェクションの爆発範囲が大幅に増加し、Azure リソースが侵害される可能性があります。 AWS DevOps Agent は読み取りオペレーションのみを実行します。エージェントは Azure リソースを変更、作成、または削除しません。

AKS アクセス設定 (オプション)

ステップ 1: Azure Resource Manager (ARM) レベルのアクセス

Azure Kubernetes Service Cluster ユーザーロールをアプリケーションに割り当てます。

Azure ポータルで、サブスクリプション → サブスクリプションの選択 → アクセスコントロール (IAM) → ロールの割り当ての追加 → Azure Kubernetes サービスクラスターユーザーロールの選択 → アプリケーション (管理者同意用の AWS DevOps エージェント、またはアプリ登録用の独自の Entra アプリケーション) への割り当てに移動します。

これは、サブスクリプション内のすべての AKS クラスターを対象としています。特定のクラスターに限定するには、代わりにリソースグループまたは個々のクラスターレベルで を割り当てます。

ステップ 2: Kubernetes API アクセス

クラスターの認証設定に基づいて 1 つのオプションを選択します。

オプション A: Azure Role-Based Access Control (RBAC) for Kubernetes (推奨)

1. まだ有効になっていない場合は、クラスターで Azure RBAC を有効にする: Azure Portal → AKS クラスター → 設定 → セキュリティ設定 → 認証と認可 → Azure RBAC を選択する

2. 読み取り専用ロールの割り当て: Azure Portal → サブスクリプション → サブスクリプションの選択 → アクセスコントロール (IAM) → ロールの割り当ての追加 → Azure Kubernetes Service RBAC Reader の選択 → アプリケーションへの割り当て

これは、サブスクリプション内のすべての AKS クラスターを対象としています。

オプション B: Azure Active Directory (Azure AD) + Kubernetes RBAC

クラスターが既にデフォルトの Azure AD 認証設定を使用していて、Azure RBAC を有効にしない場合は、これを使用します。これには、クラスターごとのkubectlセットアップが必要です。

1. 次のマニフェストを として保存しますdevops-agent-reader.yaml。

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: devops-agent-reader
rules:
  - apiGroups: [""]
    resources: ["namespaces", "pods", "pods/log", "services", "events", "nodes"]
    verbs: ["get", "list"]
  - apiGroups: ["apps"]
    resources: ["deployments", "replicasets", "statefulsets", "daemonsets"]
    verbs: ["get", "list"]
  - apiGroups: ["metrics.k8s.io"]
    resources: ["pods", "nodes"]
    verbs: ["get", "list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: devops-agent-reader-binding
subjects:
  - kind: User
    name: "<SERVICE_PRINCIPAL_OBJECT_ID>"
    apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: devops-agent-reader
  apiGroup: rbac.authorization.k8s.io
```

1. をサービスプリンシパルのオブジェクト ID <SERVICE_PRINCIPAL_OBJECT_ID>に置き換えます。これを見つけるには: Azure Portal → Entra ID → Enterprise Applications → search for the application name (AWS DevOps Agent for Admin consent, or your own Entra application for App Registration)。
2. を各クラスターに適用します。

```
az aks get-credentials --resource-group <rg> --name <cluster-name>
kubectl apply -f devops-agent-reader.yaml
```

> 注: ローカルアカウントのみ (Azure AD なし) を使用するクラスターはサポートされていません。この機能を使用するには、クラスターで Azure AD 統合を有効にすることをお勧めします。

最小特権のカスタムロール (オプション)

アクセスコントロールを強化するには、広範なリーダーロールではなく、 AWS DevOps Agent が使用するリソースプロバイダーのみを対象とするカスタム Azure ロールを作成できます。

```
{
  "Name": "AWS DevOps Agent - Azure Reader",
  "Description": "Least-privilege read-only access for AWS DevOps Agent incident
 investigations.",
  "Actions": [
    "Microsoft.AlertsManagement/*/read",
    "Microsoft.Compute/*/read",
    "Microsoft.ContainerRegistry/*/read",
    "Microsoft.ContainerService/*/read",
    "Microsoft.ContainerService/managedClusters/commandResults/read",
    "Microsoft.DocumentDB/*/read",
    "Microsoft.Insights/*/read",
    "Microsoft.KeyVault/vaults/read",
    "Microsoft.ManagedIdentity/*/read",
    "Microsoft.Monitor/*/read",
    "Microsoft.Network/*/read",
    "Microsoft.OperationalInsights/*/read",
    "Microsoft.ResourceGraph/resources/read",
    "Microsoft.ResourceHealth/*/read",
    "Microsoft.Resources/*/read",
    "Microsoft.Sql/*/read",
    "Microsoft.Storage/*/read",
```

```
    "Microsoft.Web/*/read"
  ],
  "NotActions": [],
  "DataActions": [],
  "NotDataActions": [],
  "AssignableScopes": [
    "/subscriptions/{your-subscription-id}"
  ]
}
```

サブスクリプションとエージェントスペースの関連付け

アカウントレベルで Azure を登録したら、特定のサブスクリプションを エージェントスペースに関連付けます。

1. AWS DevOps エージェントコンソールで、エージェントスペースを選択します。
2. 機能タブに移動する
3. 「セカンダリソース」セクションで、「追加」をクリックします。
4. Azure を選択する
5. 関連付ける Azure サブスクリプションのサブスクリプション ID を指定します。
6. 追加をクリックして関連付けを完了します

複数のサブスクリプションを同じエージェントスペースに関連付けることで、Azure 環境全体でエージェントを可視化できます。

Azure リソース接続の管理

- 接続されたサブスクリプションの表示 – 機能タブのセカンダリソースセクションには、接続されたすべての Azure サブスクリプションが一覧表示されます。
- サブスクリプションの削除 – エージェントスペースからサブスクリプションを切断するには、セカンダリソースリストでサブスクリプションを選択し、削除をクリックします。これはアカウントレベルの登録には影響しません。
- 登録の削除 – Azure Cloud 登録を完全に削除するには、「機能プロバイダー」ページに移動し、登録を削除します。最初にすべてのエージェントスペースの関連付けを削除する必要があります。

Azure DevOps の接続

Azure DevOps 統合により、 AWS DevOps エージェントは Azure DevOps 組織のリポジトリとパイプライン実行履歴にアクセスできます。エージェントは、コードの変更とデプロイを運用上のインシデントと関連付けて、潜在的な根本原因を特定するのに役立ちます。

> 注: Azure DevOps パイプラインは、Azure Repos、GitHub、または Bitbucket のソースコードを使用できます。Azure DevOps 統合は、ソースプロバイダーに関係なく、パイプライン実行履歴へのアクセスを提供します。ただし、調査中に実際のソースコードにアクセスするには、 などのサポートされている統合を通じてリポジトリを個別に接続する必要がありますthe section called “GitHub の接続”。Bitbucket のソースコードは、この統合を通じて直接アクセスすることはできません。

この統合は、 AWS アカウントレベルで Azure DevOps を登録し、特定のプロジェクトを個々のエージェントスペースに関連付けるという 2 つのステップのプロセスに従います。

前提条件

Azure DevOps を接続する前に、以下を確認してください。

- AWS DevOps エージェントコンソールへのアクセス
- リポジトリとパイプライン履歴を含むプロジェクトが少なくとも 1 つある Azure DevOps 組織
- Azure DevOps 組織にユーザーを追加するアクセス許可
- 管理者の同意方法: Microsoft Entra ID で管理者の同意を実行するアクセス許可を持つアカウント
- アプリ登録方法の場合: フェデレーティッド ID 認証情報を設定するアクセス許可を持ち、 AWS アカウントでアウトバウンド ID フェデレーションが有効になっている Entra アプリケーション

> 注: エージェントスペース内から登録を開始することもできます。Pipelines セクションに移動し、追加をクリックして Azure DevOps を選択します。Azure DevOps がまだ登録されていない場合は、コンソールが最初に登録を案内します。

管理者の同意による Azure DevOps の登録

管理者同意の方法では、 AWS DevOps エージェントマネージドアプリケーションで同意ベースのフローを使用します。

ステップ 1: 登録を開始する

1. AWS マネジメントコンソールにサインインし、 AWS DevOps エージェントコンソールに移動します。
2. 機能プロバイダーページに移動する
3. Azure DevOps セクションを見つけ、登録をクリックします。
4. プロンプトが表示されたら、Azure DevOps 組織名を入力します。

ステップ 2: 管理者の同意を完了する

1. クリックして続行 - Microsoft Entra 管理者の同意ページにリダイレクトされます
2. 管理者の同意を実行するアクセス許可を持つユーザープリンシパルアカウントでサインインする
3. AWS DevOps Agent アプリケーションを確認して同意する

ステップ 3: ユーザー認可を完了する

1. 管理者の同意後、承認されたテナントのメンバーとして ID を検証するユーザー認可を求められます。
2. 同じ Azure テナントに属するアカウントでサインインする
3. 認可後、成功ステータスで AWS DevOps エージェントコンソールにリダイレクトされます。

ステップ 4: Azure DevOps でアクセス権を付与する

以下の Azure DevOps でのアクセス権の付与を参照してください。ユーザーを追加するときに AWS DevOps エージェントを検索します。

アプリ登録による Azure DevOps の登録

アプリ登録は、Azure リソースと Azure DevOps の間で共有されます。Azure リソースのアプリ登録をすでに完了している場合は、「Azure DevOps でのアクセス権の付与」に進むことができます。

ステップ 1: ADO アプリの登録を開始する

1. AWS DevOps エージェントコンソールで、機能プロバイダーページに移動します。
2. Azure Cloud セクションを見つけて Register をクリックします。
3. アプリ登録方法を選択する

ステップ 2: Entra アプリケーションを作成して設定する

コンソールに表示される手順に従って、次の操作を行います。

1. AWS アカウントでアウトバウンド ID フェデレーションを有効にする (IAM コンソールで、アカウント設定 → アウトバウンド ID フェデレーションに移動)
2. Microsoft Entra ID で Entra アプリケーションを作成するか、既存のアプリケーションを使用します。
3. アプリケーションでフェデレーション ID 認証情報を設定する

ステップ 3: 登録の詳細を入力する

登録フォームに以下を入力します。

- テナント ID – Azure テナント識別子
- テナント名 – テナントの表示名
- クライアント ID – Entra アプリケーションのアプリケーション (クライアント) ID
- 対象者 – フェデレーティッド認証情報の対象者識別子

ステップ 4: IAM ロールを作成する

コンソールから登録を送信すると、IAM ロールが自動的に作成されます。これにより、 AWS DevOps エージェントは認証情報を引き受け、 を呼び出すことができますsts:GetWebIdentityToken。

ステップ 5: 登録を完了する

1. AWS DevOps エージェントコンソールで設定を確認する
2. 送信をクリックして登録を完了します

ステップ 6: Azure DevOps でアクセス権を付与する

以下の「Azure DevOps でのアクセス権の付与」を参照してください。ユーザーを追加するときに、アプリ登録中に作成した Entra アプリケーションを検索します。

Azure DevOps でのアクセス権の付与

登録後、アプリケーションに Azure DevOps 組織へのアクセス権を付与します。このステップは、管理者の同意方法とアプリ登録方法の両方で同じです。

1. Azure DevOps で、組織設定 > ユーザー > ユーザーの追加 に移動します。
2. アプリケーション (管理者の同意のための AWS DevOps エージェント、またはアプリ登録のための独自の Entra アプリケーション) を検索します。
3. アクセスレベルを Basic に設定する
4. プロジェクトに追加で、エージェントがアクセスするプロジェクトを選択します。
5. Azure DevOps グループで、プロジェクトリーダーを選択します。
6. 追加をクリックして完了します

> セキュリティ要件: Project Readers グループのみを割り当てます。読み取り専用アクセスは、エージェントを読み取り専用オペレーションに制限し、間接的なプロンプトインジェクション攻撃の影響を制限するセキュリティ境界として機能します。書き込みまたはアクションのアクセス許可を持つグループを割り当てると、プロンプトインジェクションのブラスト半径が大幅に増加し、Azure DevOps リソースが侵害される可能性があります。

プロジェクトとエージェントスペースの関連付け

アカウントレベルで Azure DevOps を登録したら、特定のプロジェクトをエージェントスペースに関連付けます。

1. AWS DevOps エージェントコンソールで、エージェントスペースを選択します。
2. 機能タブに移動する
3. 「パイプライン」セクションで、「追加」をクリックします。
4. 利用可能なプロバイダーのリストから Azure DevOps を選択する
5. 利用可能なプロジェクトのドロップダウンからプロジェクトを選択する
6. 追加をクリックして関連付けを完了します

Azure DevOps 接続の管理

- 接続されたプロジェクトの表示 – 機能タブのパイプラインセクションには、接続されたすべての Azure DevOps プロジェクトが一覧表示されます。

- プロジェクトの削除 – エージェントスペースからプロジェクトを切断するには、パイプラインセクションでプロジェクトを選択し、削除をクリックします。
- 登録の削除 – Azure DevOps 登録を完全に削除するには、「機能プロバイダー」ページに移動し、登録を削除します。最初にすべてのエージェントスペースの関連付けを削除する必要があります。

CI/CD パイプラインへの接続

CI/CD パイプライン統合により、 AWS DevOps Agent はデプロイをモニタリングし、調査中にコードの変更を運用上のインシデントに関連付けることができます。CI/CD プロバイダーを接続することで、エージェントはデプロイイベントを追跡し、 AWS リソースに関連付けて、インシデント対応中に潜在的な根本原因を特定できます。

AWS DevOps Agent は、2 ステップのプロセスを通じて一般的な CI/CD プラットフォームとの統合をサポートします。

1. アカウントレベルの登録 – CI/CD プロバイダーを AWS アカウントレベルで 1 回登録します。
2. エージェントスペース接続 – 組織のニーズに基づいて、特定のプロジェクトまたはリポジトリを個々のエージェントスペースに接続します。

このアプローチにより、各スペースでモニタリングされるプロジェクトをきめ細かく制御しながら、複数のエージェントスペース間で CI/CD プロバイダー登録を共有できます。

サポートされている CI/CD プロバイダー

AWS DevOps Agent は、次の CI/CD プラットフォームをサポートしています。

- GitHub – AWS DevOps Agent GitHub アプリを使用して GitHub.com からリポジトリを接続します。 GitHub
- GitLab – GitLab.com,マネージド GitLab インスタンス、またはパブリックにアクセス可能なセルフホスト GitLab デプロイからプロジェクトを接続します。

トピック

- the section called “GitHub の接続”
- the section called “GitLab の接続”

GitHub の接続

GitHub 統合により、 AWS DevOps Agent はコードリポジトリにアクセスし、インシデント調査中にデプロイイベントを受信できます。この統合は、GitHub のアカウントレベルの登録と、特定のリポジトリを個々のエージェントスペースに接続するという 2 つのステップのプロセスに従います。

AWS DevOps Agent は、GitHub.com (SaaS) インスタンスと GitHub Enterprise Server (セルフホスト) インスタンスの両方をサポートしています。

前提条件

GitHub を接続する前に、以下を確認してください。

- AWS DevOps エージェント管理コンソールへのアクセス
- 管理者権限を持つ GitHub ユーザーアカウントまたは組織
- アカウントまたは組織に GitHub アプリをインストールする認可

GitHub Enterprise Server の場合、以下も必要です。

- HTTPS 経由でアクセス可能な GitHub Enterprise Server インスタンス (バージョン 3.x 以降)
- GitHub Enterprise Server インスタンスの HTTPS URL (例: `https://github.example.com`)
- (オプション) GitHub Enterprise Server インスタンスがパブリックにアクセスできない場合のプライベート接続

GitHub の登録 (アカウントレベル)

GitHub は AWS アカウントレベルで登録され、そのアカウントのすべてのエージェントスペース間で共有されます。 AWS アカウントごとに 1 回だけ GitHub を登録する必要があります。

ステップ 1: パイプラインプロバイダーに移動する

1. AWS マネジメントコンソールにサインインする
2. AWS DevOps エージェントコンソールに移動する
3. 機能タブに移動する
4. 「パイプライン」セクションで、「追加」をクリックします。
5. 利用可能なプロバイダーのリストから GitHub を選択する

GitHub がまだ登録されていない場合は、最初に登録するように求められます。

ステップ 2: 接続タイプを選択する

GitHub アカウント/組織を登録する」画面で、ユーザーとして接続するか組織として接続するかを選択します。

- ユーザー – ユーザーネームとプロファイルを持つ個人 GitHub アカウント
- 組織 – 複数のユーザーが一度に複数のプロジェクトでコラボレーションできる共有 GitHub アカウント

GitHub Enterprise Server インスタンスに接続する場合は、GitHub Enterprise Server の使用チェックボックスをオンにし、インスタンスの HTTPS URL (例:) を入力しますhttps://github.example.com。

GitHub Enterprise Server インスタンスがパブリックにアクセスできない場合は、オプションでプライベート接続を設定して、 AWS DevOps Agent がインスタンスに安全に到達できるようにします。詳細については、「the section called “プライベートにホストされたツールへの接続”」を参照してください。

> **Note**
>
> URL に /api/v3または末尾のパスを含めないでください。基本 URL のみを入力します。

ステップ 3: GitHub アプリを設定する

送信をクリックして、アプリのセットアッププロセスを開始します。次のステップは、GitHub.com と GitHub Enterprise Server のどちらに接続するかによって異なります。

GitHub.com の場合

1. GitHub にリダイレクトされ、 AWS DevOps Agent GitHub アプリがインストールされます。
2. アプリをインストールするアカウントまたは組織を選択します。
3. アプリを使用すると、 AWS DevOps Agent は、デプロイイベントなど、接続されたリポジトリからイベントを受信できます。

GitHub Enterprise Server の場合

GitHub Enterprise Server は GitHub App Manifest フローを使用します。これにより、インスタンスに新しい GitHub App が自動的に設定されます。これには、GitHub Enterprise Server インスタンスへの 2 つのリダイレクトが含まれます。

1. ブラウザが GitHub Enterprise Server インスタンスのGitHub アプリの作成」ページにリダイレクトされます。
2. アプリ名があらかじめ入力されています。必要に応じて名前を自由に変更できます。GitHub アプリの作成をクリックします。
3. AWS DevOps エージェントにリダイレクトされ、マニフェストコードをアプリケーションの認証情報と交換します。

ステップ 4: リポジトリを選択してインストールを完了する

1. GitHub アプリのインストールと認可ページが表示されます。
2. アプリがアクセスできるようにするリポジトリを選択します。
   - すべてのリポジトリ – 現在および将来のすべてのリポジトリへのアクセスを許可します。
   - リポジトリのみを選択する – アカウントまたは組織から特定のリポジトリを選択します。
3. Install & Authorize をクリックします。
4. AWS DevOps エージェントコンソールにリダイレクトされ、GitHub がアカウントレベルで登録済みとして表示されます。

エージェントスペースへのリポジトリの接続

アカウントレベルで GitHub を登録したら、特定のリポジトリを個々のエージェントスペースに接続できます。

1. AWS DevOps エージェントコンソールで、エージェントスペースを選択します。
2. 機能タブに移動する
3. パイプラインセクションで、追加 をクリックします。
4. 利用可能なプロバイダーのリストから GitHub を選択する
5. このエージェントスペースに関連するリポジトリのサブセットを選択する
6. 追加をクリックして接続を完了します

組織のニーズに応じて、異なるリポジトリのセットを異なるエージェントスペースに接続できます。

GitHub アプリについて

AWS DevOps エージェント GitHub アプリ:

- リポジトリへの読み取り専用アクセスをリクエストする
- デプロイイベントおよびその他のリポジトリイベントを受信します
- AWS DevOps エージェントによるコード変更と運用インシデントの関連付けを許可する
- GitHub 設定を使用していつでもアンインストールできます

GitHub Enterprise Server の場合、GitHub アプリは登録時にインスタンスに自動的に作成されます。アプリのリポジトリアクセスを管理したり、設定 > アプリケーション > インストールされた GitHub Apps を使用してアンインストールしたりできます。アプリ定義を完全に削除するには、設定 > デベロッパー設定 > GitHub Apps に移動します。

GitHub 接続の管理

- リポジトリアクセスの更新 – GitHub アプリがアクセスできるリポジトリを変更するには、GitHub アカウントまたは組織設定 (または GitHub Enterprise Server インスタンス設定) に移動し、インストールされている GitHub アプリに移動し、 AWS DevOps エージェントアプリ設定を変更します。
- 接続されたリポジトリの表示 – AWS DevOps エージェントコンソールで、エージェントスペースを選択し、機能タブに移動して、パイプラインセクションの接続されたリポジトリを表示します。
- GitHub 接続の削除 – GitHub をエージェントスペースから切断するには、パイプラインセクションで接続を選択し、削除をクリックします。GitHub アプリを完全にアンインストールするには、GitHub アカウントまたは組織設定からアンインストールします。GitHub Enterprise Server の場合、GitHub アプリは登録時にインスタンスに直接作成されるため、オプションで以下の両方を実行してアプリを完全にクリーンアップできます。
  - アプリをアンインストールする – 設定 > アプリケーション > インストールされた GitHub アプリに移動し、アプリで設定をクリックしてアンインストールします。
  - アプリの削除 – 設定 > デベロッパー設定 > GitHub アプリに移動し、アプリを選択し、詳細タブに移動して、GitHub アプリの削除を選択します。警告: GitHub アプリの削除は永続的であり、元に戻すことはできません。削除した場合は、 AWS DevOps エージェントコンソールの最初から GitHub Enterprise Server を再登録して、新しいアプリケーションを作成する必要があります。

GitLab の接続

GitLab 統合により、 AWS DevOps Agent は GitLab Pipelines からのデプロイをモニタリングし、インシデント対応中に因果調査を通知できます。この統合は、GitLab のアカウントレベルの登録と、特定のプロジェクトを個々のエージェントスペースに接続する 2 つのステップのプロセスに従います。

GitLab の登録 (アカウントレベル)

GitLab は AWS アカウントレベルで登録され、そのアカウントのすべてのエージェントスペース間で共有されます。個々のエージェントスペースは、エージェントスペースに適用する特定のプロジェクトを選択できます。

ステップ 1: パイプラインプロバイダーに移動する

1. AWS マネジメントコンソールにサインインする
2. AWS DevOps エージェントコンソールに移動する
3. 機能プロバイダーページに移動する (サイドナビゲーションからアクセス可能)
4. Pipeline の「利用可能なプロバイダー」セクションで GitLab を検索し、登録をクリックします

ステップ 2: GitLab 接続を設定する

GitLab 登録ページで、以下を設定します。

接続タイプ – 人として接続するかグループとして接続するかを選択します。

- Personal (デフォルト) – ユーザー名とプロファイルを持つ個々の GitLab ユーザーアカウント
- グループ – GitLab では、グループを使用して 1 つ以上の関連プロジェクトを同時に管理します。

GitLab インスタンスタイプ – 接続先の GitLab インスタンスのタイプを選択します。

- GitLab.com (デフォルト) – パブリック GitLab サービス
- パブリックにアクセス可能なセルフホスト型 GitLab – GitLab セルフホスト型エンドポイントの使用チェックボックスをオンにし、GitLab インスタンスへの URL を指定します。

> **Note**
>
> 現在、パブリックにアクセス可能な GitLab インスタンスのみがサポートされています。

アクセストークン – GitLab 個人用アクセストークンを提供します。

1. 別のブラウザタブで、GitLab アカウントにログインします。
2. ユーザー設定に移動し、アクセストークンを選択する
3. 次のアクセス許可を持つ新しい個人用アクセストークンを作成します。
   - `read_repository` – リポジトリコンテンツにアクセスするために必要です
   - `read_virtual_registry` – 仮想レジストリ情報にアクセスするために必要です
   - `read_registry` – レジストリ情報にアクセスするために必要です
   - `api` – 読み取りおよび書き込み API アクセスに必要です
   - `self_rotate` - トークンのローテーションに必要です。この機能は現在、 AWS DevOps エージェントではサポートされていませんが、後日サポートされます。これで を追加すると、今後新しいトークンを作成する必要がなくなります。
4. トークンの有効期限を現在の日付から最大 365 日に設定します。
5. 生成されたトークンをコピーする
6. AWS DevOps エージェントコンソールに戻る
7. トークンを「アクセストークン」フィールドに貼り付けます。

ステップ 3: 登録を完了する

(オプション) タグ – 組織の目的で GitLab 登録に AWS タグを追加します。

次へをクリックして設定を確認し、送信をクリックして GitLab 登録プロセスを完了します。システムはアクセストークンを検証し、接続を確立します。

エージェントスペースへのプロジェクトの接続

アカウントレベルで GitLab を登録したら、特定のプロジェクトを個々のエージェントスペースに接続できます。

1. AWS DevOps エージェントコンソールで、エージェントスペースを選択します。
2. 機能タブに移動する

3. パイプラインセクションで、追加をクリックします。
4. 利用可能なプロバイダーのリストから GitLab を選択する
5. エージェントスペースに関連する GitLab プロジェクトを選択する
6. 保存 をクリックします。

AWS DevOps エージェントは、GitLab Pipelines からのデプロイについてこれらのプロジェクトをモニタリングし、因果調査を通知します。

GitLab 接続の管理

- アクセストークンの更新 – アクセストークンの有効期限が切れるか、更新する必要がある場合は、アカウントレベルで GitLab 登録を変更することで、 AWS DevOps エージェントコンソールで更新できます。
- 接続されたプロジェクトの表示 – AWS DevOps エージェントコンソールで、エージェントスペースを選択し、機能タブに移動して、パイプラインセクションで接続されたプロジェクトを表示します。
- GitLab 接続の削除 – GitLab プロジェクトをエージェントスペースから切断するには、パイプラインセクションで接続を選択し、削除をクリックします。GitLab 登録を完全に削除するには、まずすべてのエージェントスペースから削除してから、アカウントレベルで登録を削除します。

MCP サーバーの接続

Model Context Protocol (MCP) サーバーは、外部オブザーバビリティツール、カスタムモニタリングシステム、運用データソースからのデータへのアクセスを提供することで、 AWS DevOps Agent の調査機能を拡張します。このガイドでは、MCP サーバーを AWS DevOps エージェントに接続する方法について説明します。

要件

MCP サーバーを接続する前に、サーバーが次の要件を満たしていることを確認してください。

- ストリーミング可能な HTTP トランスポートプロトコル – ストリーミング可能な HTTP トランスポートプロトコルを実装する MCP サーバーのみがサポートされています。
- 認証のサポート – MCP サーバーは OAuth 2.0 認証フローまたは API キー/トークンベースの認証をサポートしている必要があります。

セキュリティに関する考慮事項

MCP サーバーを AWS DevOps エージェントに接続するときは、次のセキュリティ面を考慮してください。

- ツールの許可リスト – MCP サーバーからすべてのツールを公開するのではなく、エージェントスペースが必要とする特定のツールのみを許可リストに登録する必要があります。エージェントスペースごとにリストツールを許可する方法については、「エージェントスペースでの MCP ツールの設定」を参照してください。

MCP ツールの最大長は 64 です。

- プロンプトインジェクションリスク – カスタム MCP サーバーは、プロンプトインジェクション攻撃のリスクを高める可能性があります。詳細については、「プロンプトインジェクション保護: AWS DevOps エージェントセキュリティ」を参照してください。
- 読み取り専用ツールとアクセス – 読み取り専用 MCP ツールのみを許可リストに登録し、認証情報が読み取り専用アクセスのみを許可されていることを確認します。

プロンプトインジェクションと責任共有モデルの詳細については、AWS DevOps エージェントセキュリティ「」を参照してください。

> **Note**
>
> MCP サーバーがプライベートネットワーク上にある場合は、「」を参照してください。 the section called “プライベートにホストされたツールへの接続”

MCP サーバーの登録 (アカウントレベル)

MCP サーバーは AWS アカウントレベルで登録され、そのアカウントのすべてのエージェントスペース間で共有されます。個々のエージェントスペースは、各 MCP サーバーから必要な特定のツールを選択できます。

ステップ 1: MCP サーバーの詳細

1. AWS マネジメントコンソールにサインインする
2. AWS DevOps エージェントコンソールに移動する

3. 機能プロバイダーページに移動する (サイドナビゲーションからアクセス可能)
4. 利用可能なプロバイダーセクションで MCP サーバーを検索し、登録をクリックします
5. MCP サーバーの詳細ページで、次の情報を入力します。
   - 名前 – MCP サーバーのわかりやすい名前を入力します。
   - エンドポイント URL – MCP サーバーエンドポイントの完全な HTTPS URL を入力します。
   - 説明 (オプション) – サーバーの目的を特定するのに役立つ説明を追加します。
   - 動的クライアント登録を有効にする – AWS DevOps エージェントが MCP サーバーの認可サーバーに自動的に登録できるようにする場合は、このチェックボックスをオンにします。
6. [次へ] をクリックします。

> **Note**
>
> MCP サーバーエンドポイント URL は、アカウントの AWS CloudTrail ログに表示されます。

ステップ 2: 認可フロー

MCP サーバーの認証方法を選択します。

OAuth クライアント認証情報 – MCP サーバーが OAuth クライアント認証情報フローを使用している場合:

1. OAuth クライアント認証情報の選択
2. [次へ] をクリックします。

OAuth 3LO (Three-Legged OAuth) – MCP サーバーが認証に OAuth 3LO を使用している場合:

1. OAuth 3LO を選択する
2. [次へ] をクリックします。

API キー – MCP サーバーが API キー認証を使用する場合:

1. API キーの選択
2. [次へ] をクリックします。

ステップ 3: 認可設定

選択した認証方法に基づいて追加の認可パラメータを設定します。

OAuth クライアント認証情報の場合:

1. クライアント ID – OAuth クライアントのクライアント ID を入力します。
2. クライアントシークレット – OAuth クライアントのクライアントシークレットを入力します。
3. Exchange URL – OAuth トークン交換エンドポイント URL を入力します。
4. Exchange Parameters – サービスで認証するための OAuth トークン交換パラメータを入力します。
5. スコープの追加 – 認証用の OAuth スコープの追加
6. [次へ] をクリックします。

OAuth 3LO の場合:

1. クライアント ID – OAuth クライアントのクライアント ID を入力します。
2. クライアントシークレット – OAuth クライアントで必要な場合は、OAuth クライアントのクライアントシークレットを入力します。
3. Exchange URL – OAuth トークン交換エンドポイント URL を入力します。
4. 認可 URL - OAuth 認可エンドポイント URL を入力します
5. コードチャレンジのサポート - OAuth クライアントがコードチャレンジをサポートしている場合は、このチェックボックスをオンにします
6. スコープの追加 – 認証用の OAuth スコープの追加
7. [次へ] をクリックします。

API キーの場合:

1. API キー名を入力する
2. リクエストに API キーを含むヘッダーの名前を入力します。
3. API キーの値を入力する
4. [次へ] をクリックします。

ステップ 4: 確認して送信する

1. すべての MCP サーバー設定の詳細を確認する
2. 送信をクリックして登録を完了します
3. AWS DevOps エージェントは MCP サーバーへの接続を検証します
4. 検証に成功すると、MCP サーバーはアカウントレベルで登録されます。

エージェントスペースでの MCP ツールの設定

アカウントレベルで MCP サーバーを登録したら、そのサーバーのどのツールを特定のエージェントスペースで使用できるかを設定できます。

1. AWS DevOps エージェントコンソールで、エージェントスペースを選択します。
2. 機能タブに移動する
3. MCP サーバーセクションで、追加
4. このエージェントスペースに接続する登録済み MCP サーバーを選択します。
5. エージェントスペースで使用できるように、この MCP サーバーのツールを設定します。
   - すべてのツールを許可する – MCP サーバーからすべてのツールを使用できるようにします
   - 特定のツールの選択 – 許可リストを作成するツールを選択できます。
6. 追加をクリックして、MCP サーバーをエージェントスペースに接続します。

AWS DevOps Agent は、このエージェントスペースの調査中に MCP サーバーから許可リストに登録されたツールを使用できるようになりました。

MCP サーバー接続の管理

認証情報の更新 – 認証情報を更新する必要がある場合は、MCP サーバーを再登録する必要があります。 AWS DevOps エージェントコンソールの Capability Providers ページに移動し、MCP サーバーを見つけ、アクティブな関連付けを削除して、登録解除をクリックします。次に、新しい認証情報で MCP サーバーを登録し、エージェントスペースと必要な関連付けを再作成します。

接続された MCP サーバーの表示 – エージェントスペースに接続されているすべての MCP サーバーを表示するには、エージェントスペースを選択し、機能タブに移動して MCP サーバーセクションを確認します。選択したツールをここで更新することもできます。

MCP サーバー接続の削除 – MCP サーバーをエージェントスペースから切断するには、MCP サーバーセクションでサーバーを選択し、削除をクリックします。MCP サーバー登録を完全に削除するには、まずすべてのエージェントスペースから削除してから、アカウントレベルの登録を削除します。

関連トピック

- AWS DevOps エージェントのセキュリティ
- エージェントスペースのセットアップ
- プロンプトインジェクション保護

複数の AWS アカウントを接続する

セカンダリ AWS アカウントを使用すると、 AWS DevOps Agent は組織内の複数の AWS アカウントのリソースを調査できます。アプリケーションが複数のアカウントにまたがる場合、セカンダリアカウントを追加すると、エージェントはインシデント調査中にすべての関連リソースを可視化できます。アプリケーションを構成するアカウントとリソースへのアクセスを増やすと、調査の精度が向上します。

前提条件

セカンダリ AWS アカウントを追加する前に、以下があることを確認してください。

- プライマリアカウントの AWS DevOps エージェントコンソールへのアクセス
- セカンダリ AWS アカウントへの管理アクセス
- セカンダリアカウントでロールを作成する IAM アクセス許可

セカンダリ AWS アカウントの追加

以下のステップに加えて、 を使用してセカンダリアカウントthe section called “AWS DevOps エージェント CLI オンボーディングガイド”をプログラムで追加できます。

ステップ 1: セカンダリアカウント設定を開始する

1. AWS マネジメントコンソールにサインインし、 AWS DevOps エージェントコンソールに移動します。

2. エージェントスペースを選択する
3. 機能タブに移動する
4. クラウドセクションで、セカンダリソースサブセクションを見つけます。
5. 追加をクリックします

ステップ 2: ロール名を指定する

1. ロールの名前フィールドに、セカンダリアカウントで作成するロールの名前を入力します。
2. この名前に注意してください。セカンダリアカウントでロールを作成するときに再度使用します。
3. コンソールで提供されている信頼ポリシーをコピーし、スクラッチスペースに保存します。

ステップ 3: セカンダリアカウントにロールを作成する

1. 新しいブラウザタブを開き、セカンダリ AWS アカウントの IAM コンソールにサインインします。
2. IAM > ロール > ロールの作成に移動する
3. カスタム信頼ポリシーの選択
4. ステップ 2 からコピーした信頼ポリシーを貼り付ける
5. [次へ] をクリックします。

ステップ 4: AWS 管理ポリシーをアタッチする

1. アクセス許可ポリシーセクションで、AIOpsAssistantPolicy を検索します。
2. AIOpsAssistantPolicy 管理ポリシーの横にあるチェックボックスをオンにします。
3. [次へ] をクリックします。

ステップ 5: ロールに名前を付けて作成する

1. ロール名フィールドに、ステップ 2 で指定したのと同じロール名を入力します。
2. (オプション) ロールの目的を特定するのに役立つ説明を追加します。
3. 信頼ポリシーとアタッチされたアクセス許可を確認する

4. ロールの作成 をクリックします。

ステップ 6: インラインポリシーをアタッチする

1. IAM コンソールで、先ほど作成したロールを見つけて選択します。
2. アクセス許可タブに移動する
3. アクセス許可の追加 > インラインポリシーの作成 をクリックします。
4. JSON タブに切り替える
5. ステップ 2 で保存したポリシーを貼り付ける
6. IAM コンソールの JSON エディタにポリシーを貼り付ける
7. [次へ] をクリックします。
8. インラインポリシーの名前を指定します (DevOpsAgentInlinePolicy」など)。
9. ポリシーの作成 をクリックします。

ステップ 7: 設定を完了する

1. プライマリアカウントの AWS DevOps エージェントコンソールに戻る
2. 次へをクリックしてセカンダリアカウント設定を完了します
3. 接続ステータスがアクティブと表示されることを確認する

必要なポリシーについて

AWS DevOps Agent では、セカンダリアカウントのリソースにアクセスするために 3 つのポリシーコンポーネントが必要です。

- 信頼ポリシー – プライマリアカウントの AWS DevOps エージェントがセカンダリアカウントのロールを引き受けることを許可します。これにより、アカウント間の信頼関係が確立されます。
- AIOpsAssistantPolicy (AWS 管理ポリシー) – セカンダリアカウントのリソースを調査するために AWS DevOps Agent が必要とするコア読み取り専用アクセス許可を提供します。このポリシーはによって維持 AWS され、新機能が追加されると更新されます。
- インラインポリシー – エージェントスペース設定に固有の追加のアクセス許可を提供します。このポリシーは、エージェントスペースの設定に基づいて生成され、特定の統合または機能のアクセス許可が含まれる場合があります。

プライマリアカウントでは、 AWS DevOps エージェント IAM ロールがセカンダリアカウントで作成されたロールを引き受けることができる必要があります。

セカンダリアカウントの管理

- 接続されたアカウントの表示 – 機能タブで、セカンダリソースサブセクションは接続ステータスを持つすべての接続されたセカンダリアカウントを一覧表示します。
- IAM ロールの更新 – アクセス許可を変更する必要がある場合は、セカンダリアカウントのロールにアタッチされたインラインポリシーを更新します。変更は即時適用されます。
- セカンダリアカウントの削除 – セカンダリアカウントを切断するには、セカンダリソースリストでセカンダリアカウントを選択し、削除をクリックします。これにより、セカンダリアカウントの IAM ロールは削除されません。

テレメトリソースの接続

AWS DevOps エージェントには、テレメトリソースに接続する 3 つの方法があります。

組み込みの双方向統合

現在、 AWS DevOps Agent は、以下を可能にする 2 方向統合が組み込まれた Dynatrace ユーザーをサポートしています。

- トポロジーリソースマッピング - AWS DevOps Agent は、DevOps Agent がホストする Dynatrace MCP サーバーを介して利用可能なエンティティと関係で AWS DevOps Agent Space Topology を強化します。
- 自動調査トリガー - Dynatrace ワークフローは、Dynatrace 問題からインシデント解決調査をトリガーするように設定できます。
- テレメトリのイントロスペクション - AWS DevOps Agent は、 AWS DevOps Agent がホストする Dynatrace MCP サーバーを介して問題を調査する際に、Dynatrace テレメトリをイントロスペクションできます。
- ステータスの更新 - AWS DevOps Agent は、主要な調査結果、根本原因分析、生成された緩和計画を Dynatrace ユーザーインターフェイスに発行します。

双方向統合の詳細については、「」を参照してください。

- the section called “Dynatrace の接続”

組み込みの 1 方向統合

現在、 AWS DevOps エージェントは、 AWS CloudWatch、Datadog、Grafana、New Relic、Splunk ユーザーを組み込みの 1 方向統合でサポートしています。

セキュリティのベストプラクティス: 組み込みの 1 方向統合の認証情報を設定する場合は、API キーとトークンの範囲を読み取り専用アクセスに設定することをお勧めします。 AWS DevOps Agent はこれらの認証情報をテレメトリイントロスペクションにのみ使用し、テレメトリプロバイダーへの書き込みアクセスは必要ありません。

AWS CloudWatch 組み込みの 1 方向統合は、追加のセットアップを必要とせず、以下を有効にします。

- トポロジーリソースマッピング - AWS DevOps エージェントは、設定されたプライマリクラウドアカウントとセカンダリ AWS クラウドアカウントを介して利用可能なエンティティと関係で DevOps エージェントスペーストポロジーを強化します。
- テレメトリイントロスペクション - AWS DevOps Agent は、プライマリおよびセカンダリの AWS クラウドアカウント設定中に提供された IAM ロール (複数可) を介して問題を調査する際に、Introspect AWS CloudWatch テレメトリを使用できます。

Datadog、Grafana、New Relic、Splunk の組み込みの 1 方向統合では、以下を設定して有効にする必要があります。

- 自動調査トリガー - Datadog、Grafana、New Relic、Splunk イベントは、 AWS DevOps Agent ウェブフックを介して AWS DevOps Agent インシデント解決調査をトリガーするように設定できます。
- テレメトリイントロスペクション - AWS DevOps エージェントは、各プロバイダーのリモート MCP サーバーを介して問題を調査する際に、Datadog、Grafana、New Relic、Splunk テレメトリをイントロスペクションできます。

一方向統合の詳細については、以下を参照してください。

- the section called “DataDog の接続”
- the section called “Grafana の接続”
- the section called “New Relic の接続”
- the section called “Splunk の接続”

Bring-your-ownテレメトリソース

Prometheus メトリクスを含む他のテレメトリソースについては、ウェブフックと MCP サーバー統合の両方に対する AWS DevOps Agent のサポートを活用できます。

bring-your-own 統合の詳細については、以下を参照してください。

- the section called “Webhook による DevOps エージェントの呼び出し”
- the section called “MCP サーバーの接続”

Dynatrace の接続

組み込みの双方向統合

現在、 AWS DevOps Agent は、以下を可能にする 2 方向統合が組み込まれた Dynatrace ユーザーをサポートしています。

- トポロジーリソースマッピング - AWS DevOps Agent は、Dynatrace 環境から利用可能なエンティティと関係で DevOps Agent Space Topology を強化します。
- 自動調査トリガー - Dynatrace ワークフローは、Dynatrace 問題からインシデント解決調査をトリガーするように設定できます。
- テレメトリのイントロスペクション - AWS DevOps Agent は、 AWS DevOps Agent がホストする Dynatrace MCP サーバーを介して問題を調査する際に、Dynatrace テレメトリをイントロスペクションできます。
- ステータスの更新 - AWS DevOps Agent は、主要な調査結果、根本原因分析、生成された緩和計画を Dynatrace ユーザーインターフェイスに発行します。

オンボーディング

オンボーディングプロセス

Dynatrace オブザーバビリティシステムのオンボーディングには、次の 3 つの段階があります。

1. Connect - アカウントアクセス認証情報を設定し、必要なすべての環境を使用して Dynatrace への接続を確立します。
2. 有効 - 特定の Dynatrace 環境を持つ特定のエージェントスペースで Dynatrace をアクティブ化する

3. Dynatrace 環境を設定する - ワークフローとダッシュボードをダウンロードして Dynatrace にインポートし、ウェブフックの詳細を書き留めて、指定されたエージェントスペースで調査をトリガーします。

ステップ 1: 接続する

Dynatrace 環境への接続を確立する

設定

1. 機能プロバイダーページに移動する (サイドナビゲーションからアクセス可能)
2. Telemetry の「利用可能なプロバイダー」セクションで Dynatrace を検索し、Register をクリックします。
3. 詳細なアクセス許可を使用して、Dynatrace で OAuth クライアントを作成します。
   a. 「Dynatrace ドキュメント」を参照してください。
   b. 準備ができたら、次へを押します
   c. 複数の Dynatrace 環境とそれ以降のスコープを、DevOps エージェントスペースごとに特定の環境に接続できます。
4. OAuth クライアント設定から Dynatrace の詳細を入力します。
   - クライアント名
   - クライアント ID
   - クライアントシークレット
   - アカウント URN
5. [次へ] をクリックします。
6. 確認して追加する

ステップ 2: を有効にする

特定のエージェントスペースで Dynatrace をアクティブ化し、適切なスコープを設定する

設定

1. エージェントスペースページからエージェントスペースを選択し、詳細の表示を押します。
2. 機能タブを選択する
3. Telemetry セクションを見つけ、Add キーを押します。

4. Dynatrace のステータスが「登録済み」であることがわかります。add をクリックして、これをエージェントスペースに追加します。
5. Dynatrace 環境 ID - この DevOps エージェントスペースに関連付ける Dynatrace 環境 ID を指定します。
6. 1 つ以上の Dynatrace エンティティ IDs を入力します。これは、DevOps エージェントが最も重要なリソースを検出するのに役立ちます。例としては、サービスやアプリケーションなどがあります。不明な場合は、削除を押します。
7. 確認して保存を押します
8. Webhook URL と Webhook Secret をコピーします。これらの認証情報を Dynatrace に追加するには、Dynatrace のドキュメントを参照してください。

ステップ 3: Dynatrace 環境を設定する

Dynatrace のセットアップを完了するには、Dynatrace 環境で特定のセットアップ手順を実行する必要があります。Dynatrace ドキュメントの指示に従ってください。

サポートされているイベントスキーマ

AWS DevOps Agent は、ウェブフックを使用した Dynatrace からの 2 種類のイベントをサポートします。サポートされているイベントスキーマを以下に示します。

インシデントイベント

インシデントイベントは、調査をトリガーするために使用されます。イベントスキーマは次のとおりです。

```
{
    "event.id": string;
    "event.status": "ACTIVE" | "CLOSED";
    "event.status_transition": string;
    "event.description": string;
    "event.name": string;
    "event.category": "AVAILABILITY" | "ERROR" | "SLOWDOWN" | "RESOURCE_CONTENTION" |
 "CUSTOM_ALERT" | "MONITORING_UNAVAILABLE" | "INFO";
    "event.start"?: string;
    "affected_entity_ids"?: string[];
}
```

緩和イベント

緩和イベントは、次のステップの調査の緩和レポートの生成をトリガーするために使用されます。イベントスキーマは次のとおりです。

```
{
    "task_id": string;
    "task_version": number;
    "event.type": "mitigation_request";
}
```

削除

テレメトリソースは、エージェントスペースレベルとアカウントレベルで 2 つのレベルで接続されます。これを完全に削除するには、まずそれを使用するすべてのエージェントスペースから削除してから、登録を解除する必要があります。

ステップ 1: エージェントスペースから削除する

1. エージェントスペースページからエージェントスペースを選択し、詳細の表示を押します。
2. 機能タブを選択する
3. Telemetry セクションまで下にスクロールします。
4. Dynatrace を選択する
5. 削除を押す

ステップ 2: アカウントから登録解除する

1. 機能プロバイダーページに移動する (サイドナビゲーションからアクセス可能)
2. 現在登録されているセクションまでスクロールします。
3. エージェントスペース数がゼロであることを確認します (他のエージェントスペースで上記のステップ 1 を繰り返しない場合) 。
4. Dynatrace の横にある登録解除を押します

DataDog の接続

組み込みの 1 方向統合

現在、 AWS DevOps Agent は、組み込みの 1 方向統合で Datadog ユーザーをサポートし、以下を有効にします。

- 自動調査トリガー - Datadog イベントは、 AWS DevOps エージェントウェブフックを介して AWS DevOps エージェントインシデント解決調査をトリガーするように設定できます。
- テレメトリイントロスペクション - AWS DevOps Agent は、各プロバイダーのリモート MCP サーバーを介して問題を調査する際に、Datadog テレメトリをイントロスペクションできます。

オンボーディング

ステップ 1: 接続する

アカウントアクセス認証情報を使用して Datadog リモート MCP エンドポイントへの接続を確立する

設定

1. 機能プロバイダーページに移動する (サイドナビゲーションからアクセス可能)
2. テレメトリの「利用可能なプロバイダー」セクションで Datadog を検索し、登録をクリックします
3. Datadog MCP サーバーの詳細を入力します。
   - サーバー名 - 一意の識別子 (例: my-datadog-server)
   - エンドポイント URL - Datadog MCP サーバーエンドポイント。エンドポイント URL は、Datadog サイトによって異なります。以下の Datadog サイトエンドポイント表を参照してください。
   - 説明 - オプションのサーバーの説明
4. [次へ] をクリックします。
5. レビューして送信

Datadog サイトエンドポイント

MCP エンドポイント URL は、Datadog サイトによって異なります。サイトを識別するには、Datadog にログインしたときにブラウザの URL を確認するか、「Datadog サイトへのアクセス」を参照してください。

Datadog サイト	サイトドメイン	MCP エンドポイント URL
US1 (デフォルト)	`datadoghq.com`	`https://mcp.datadoghq.com/api/unstable/mcp-server/mcp`
US3	`us3.datadoghq.com`	`https://mcp.us3.datadoghq.com/api/unstable/mcp-server/mcp`
US5	`us5.datadoghq.com`	`https://mcp.us5.datadoghq.com/api/unstable/mcp-server/mcp`
EU1	`datadoghq.eu`	`https://mcp.datadoghq.eu/api/unstable/mcp-server/mcp`
AP1	`ap1.datadoghq.com`	`https://mcp.ap1.datadoghq.com/api/unstable/mcp-server/mcp`
AP2	`ap2.datadoghq.com`	`https://mcp.ap2.datadoghq.com/api/unstable/mcp-server/mcp`

Authorization

次の方法で OAuth 認可を完了します。

- Datadog OAuth ページでユーザーとして を承認する
- ログインしていない場合は、許可、ログイン、認可 をクリックします。

設定すると、Datadog はすべてのエージェントスペースで使用可能になります。

ステップ 2: を有効にする

特定のエージェントスペースで DataDog をアクティブ化し、適切なスコープを設定する

設定

1. エージェントスペースページからエージェントスペースを選択し、表示の詳細を押します (エージェントスペースをまだ作成していない場合は、「」を参照してくださいthe section called “エージェントスペースの作成”)。
2. 機能タブを選択する
3. Telemetry セクションまで下にスクロールします。
4. 追加を押す
5. Datadog を選択する
6. 次へ
7. 確認して Save キーを押します。
8. Webhook URL と API キーをコピーする

ステップ 3: ウェブフックを設定する

Webhook URL と API キーを使用して、Datadog を設定して、アラームなどから調査をトリガーするようにイベントを送信できます。

DevOps エージェントによって送信されるイベントを使用できるようにするには、ウェブフックに送信されるデータが以下で指定されたデータスキーマと一致していることを確認してください。このスキーマに一致しないイベントは、DevOps エージェントによって無視される場合があります。

メソッドとヘッダーを設定する

```
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      "Authorization": "Bearer <Token>",
    },
```

本文を JSON 文字列として送信します。

```
{
    eventType: 'incident';
    incidentId: string;
    action: 'created' | 'updated' | 'closed' | 'resolved';
    priority: "CRITICAL" | "HIGH" | "MEDIUM" | "LOW" | "MINIMAL";
    title: string;
    description?: string;
    timestamp?: string;
    service?: string;
    // The original event generated by service is attached here.
    data?: object;
}
```

Datadog https://docs.datadoghq.com/integrations/webhooks/ を使用してウェブフックを送信します (承認を選択せず、代わりにカスタムヘッダーオプションを使用することに注意してください)。

詳細: Datadog リモート MCP サーバー

削除

テレメトリソースは、エージェントスペースレベルとアカウントレベルで 2 つのレベルで接続されます。これを完全に削除するには、まず、それが使用されているすべてのエージェントスペースから削除してから、登録を解除する必要があります。

ステップ 1: エージェントスペースから削除する

1. エージェントスペースページからエージェントスペースを選択し、詳細の表示を押します。
2. 機能タブを選択する
3. Telemetry セクションまで下にスクロールします。
4. Datadog を選択する
5. 削除を押す

ステップ 2: アカウントから登録解除する

1. 機能プロバイダーページに移動する (サイドナビゲーションからアクセス可能)
2. 現在登録されているセクションにスクロールします。
3. エージェントスペース数がゼロであることを確認します (他のエージェントスペースで上記のステップ 1 を繰り返しない場合) 。
4. Datadog の横にある登録解除を押します

Grafana の接続

Grafana 統合により、 AWS DevOps Agent はインシデント調査中に Grafana インスタンスからメトリクス、ダッシュボード、アラートデータをクエリできます。この統合は、Grafana のアカウントレベルの登録と、個々のエージェントスペースへの接続という 2 つのステップのプロセスに従います。

セキュリティを向上させるために、Grafana 統合は読み取り専用ツールのみを有効にします。書き込みツールは無効になっており、有効にすることはできません。つまり、エージェントは Grafana インスタンスからデータをクエリおよび読み取ることができますが、ダッシュボード、アラート、注釈などの Grafana リソースを作成、変更、または削除することはできません。詳細については、「Security in AWS DevOps Agent」を参照してください。

Grafana の要件

Grafana を接続する前に、以下を確認してください。

- Grafana バージョン 9.0 以降。一部の機能、特にデータソース関連のオペレーションは、API エンドポイントがないため、以前のバージョンでは正しく動作しない場合があります。
- HTTPS 経由でアクセス可能な Grafana インスタンス。パブリックネットワークエンドポイントとプライベートネットワークエンドポイントの両方がサポートされています。プライベートネットワーク接続では、Grafana インスタンスをパブリックインターネットアクセスなしで VPC 内でホストできます。詳細については、「the section called “プライベートにホストされたツールへの接続”」を参照してください。
- 適切な読み取りアクセス許可を持つアクセストークンを持つ Grafana サービスアカウント

Grafana の登録 (アカウントレベル)

Grafana は AWS アカウントレベルで登録され、そのアカウントのすべてのエージェントスペース間で共有されます。

ステップ 1: Grafana を設定する

1. AWS マネジメントコンソールにサインインする
2. AWS DevOps エージェントコンソールに移動する
3. 機能プロバイダーページに移動する (サイドナビゲーションからアクセス可能)
4. Telemetry の「利用可能なプロバイダー」セクションで Grafana を検索し、登録をクリックします
5. Grafana の設定ページで、次の情報を入力します。
   - サービス名 (必須) – 英数字、ハイフン、アンダースコアのみを使用して、Grafana サーバーのわかりやすい名前を入力します。例えば、`my-grafana-server`。
   - Grafana URL (必須) – Grafana インスタンスの完全な HTTPS URL を入力します。例えば、`https://myinstance.grafana.net`。
   - サービスアカウントアクセストークン (必須) – Grafana サービスアカウントアクセストークンを入力します。トークンは通常、 で始まります`glsa_`。サービスアカウントトークンを作成するには、Grafana インスタンスに移動し、管理 > サービスアカウントに移動し、ビューワーロールを使用してサービスアカウントを作成し、トークンを生成します。
   - 説明 (オプション) – サーバーの目的を特定するのに役立つ説明を追加します。例えば、`Production Grafana server for monitoring`。
6. (オプション) 組織の目的で登録に AWS タグを追加します。
7. [次へ] をクリックします。

ステップ 2: Grafana 登録を確認して送信する

1. Grafana 設定の詳細をすべて確認する
2. 送信をクリックして登録を完了します
3. 登録が成功すると、Grafana は機能プロバイダーページの現在登録されているセクションに表示されます。

エージェントスペースへの Grafana の追加

アカウントレベルで Grafana を登録したら、個々のエージェントスペースに接続できます。

1. AWS DevOps エージェントコンソールで、エージェントスペースを選択します。
2. 機能タブに移動する
3. テレメトリセクションで、追加 をクリックします。
4. 利用可能なプロバイダーのリストから Grafana を選択する
5. 保存 をクリックします。

Grafana アラートウェブフックの設定

Grafana コンタクトポイントを介してウェブフックを送信することで、アラートが発生したときに AWS DevOps エージェント調査を自動的にトリガーするように Grafana を設定できます。ウェブフック認証方法と認証情報管理の詳細については、「」を参照してくださいthe section called “Webhook による DevOps エージェントの呼び出し”。

ステップ 1: カスタム通知テンプレートを作成する

Grafana インスタンスで、アラート > コンタクトポイント > 通知テンプレートに移動し、次の内容の新しいテンプレートを作成します。

```
{{ define "devops-agent-payload" }}
{
  "eventType": "incident",
  "incidentId": "{{ (index .Alerts 0).Labels.alertname }}-{{ (index .Alerts
 0).Fingerprint }}",
  "action": "{{ if eq .Status "resolved" }}resolved{{ else }}created{{ end }}",
  "priority": "{{ if eq .Status "resolved" }}MEDIUM{{ else }}HIGH{{ end }}",
  "title": "{{ (index .Alerts 0).Labels.alertname }}",
  "description": "{{ (index .Alerts 0).Annotations.summary }}",
  "service": "{{ if (index .Alerts 0).Labels.job }}{{ (index .Alerts 0).Labels.job }}
{{ else }}grafana{{ end }}",
  "timestamp": "{{ (index .Alerts 0).StartsAt }}",
  "data": {
    "metadata": {
      {{ range $k, $v := (index .Alerts 0).Labels }}
      "{{ $k }}": "{{ $v }}",
      {{ end }}
      "_source": "grafana"
```

```
    }
  }
}
{{ end }}
```

このテンプレートは、Grafana アラートを AWS DevOps Agent が期待するウェブフックペイロード構造にフォーマットします。アラートラベル、注釈、ステータスを適切なフィールドにマッピングし、すべてのアラートラベルをメタデータとして含めます。

> 注: このテンプレートは、グループ内の最初のアラートのみを処理します。Grafana は、デフォルトで複数の発射アラートを 1 つの通知にグループ化します。各アラートが個別に送信されるようにするには、 でグループ化するように通知ポリシーを設定しますalertname。さらに、このテンプレートはラベル値または注釈の特殊な JSON 文字をエスケープしません。アラートラベルとsummary注釈に二重引用符や改行などの文字が含まれていないことを確認すると、無効な JSON が生成されます。

ステップ 2: ウェブフックコンタクトポイントを作成する

1. Grafana でアラート > コンタクトポイントに移動し、コンタクトポイントの追加をクリックします
2. 統合タイプとして Webhook を選択する
3. AWS DevOps Agent ウェブフックエンドポイントへの URL の設定
4. オプションのウェブフック設定で、ウェブフックタイプに基づいて認証ヘッダーを設定します。詳細については、「Webhook 認証方法」を参照してください。
5. カスタムテンプレートを使用するように Message フィールドを設定します。 `{{ template "devops-agent-payload" . }}`
6. コンタクトポイントの保存 をクリックします。

ステップ 3: コンタクトポイントを通知ポリシーに割り当てる

1. アラート > 通知ポリシーに移動する
2. 既存のポリシーを編集するか、新しいポリシーを作成します。
3. 作成したウェブフックコンタクトポイントにコンタクトポイントを設定する
4. ポリシーの保存 をクリックします。

一致するアラートが発生すると、Grafana はフォーマットされたペイロードを AWS DevOps エージェントに送信し、これにより自動的に調査が開始されます。

制限事項

- ClickHouse データソースツール – ClickHouse データソースツールは現在サポートされていません。
- プロアクティブインシデント防止 – は現在 Grafana ツールを使用しthe section called “プロアクティブインシデント防止”ません。サポートは今後のリリースが予定されています。

Amazon Managed Grafana に関する考慮事項

Amazon Managed Grafana (AMG) を使用している場合は、次の制限に注意してください。

- Webhook コンタクトポイントはサポートされていません – AMG は現在、アラート設定で Webhook コンタクトポイントをサポートしていません。AMG を使用してアラートウェブフックを AWS DevOps エージェントに直接送信することはできません。詳細については、「Amazon Managed Grafana でのコンタクトポイントのアラート」を参照してください。
- サービスアカウントトークンの有効期限 – AMG サービスアカウントトークンの最大有効期限は 30 日です。トークンをローテーションし、有効期限が切れる前に AWS DevOps Agent で Grafana 登録を更新する必要があります。認証情報を更新する方法については、「Grafana 接続の管理」を参照してください。AMG トークンの制限の詳細については、「Amazon Managed Grafana のサービスアカウント」を参照してください。

Grafana 接続の管理

- 認証情報の更新 – サービスアカウントトークンの有効期限が切れるか、更新する必要がある場合は、機能プロバイダーページから Grafana を登録解除し、新しいトークンに再登録します。
- 接続されたインスタンスの表示 – AWS DevOps エージェントコンソールで、エージェントスペースを選択し、機能タブに移動して、接続されたテレメトリソースを表示します。
- Grafana の削除 – エージェントスペースから Grafana を切断するには、テレメトリセクションでそれを選択し、削除をクリックします。登録を完全に削除するには、まずすべてのエージェントスペースから削除してから、機能プロバイダーページから登録を解除します。

New Relic の接続

組み込みの 1 方向統合

現在、 AWS DevOps Agent は、組み込みの 1 方向統合で New Relic ユーザーをサポートし、以下を有効にします。

- 自動調査トリガー - New Relic イベントは、 AWS DevOps エージェントウェブフックを介して AWS DevOps エージェントインシデント解決調査をトリガーするように設定できます。
- テレメトリのイントロスペクション - AWS DevOps Agent は、各プロバイダーのリモート MCP サーバーを介して問題を調査する際に、New Relic テレメトリをイントロスペクションできます。

オンボーディング

ステップ 1: 接続する

アカウントアクセス認証情報を使用して New Relic リモート MCP エンドポイントへの接続を確立する

New Relic MCP ツールを有効にするには、New Relic でフルプラットフォームユーザー (ベーシック/コアではない) を使用してください。

設定

1. 機能プロバイダーページに移動する (サイドナビゲーションからアクセス可能)
2. Telemetry の「利用可能なプロバイダー」セクションで New Relic を検索し、登録をクリックします
3. 手順に従って New Relic API キーを取得します。
4. New Relic MCP サーバー API キーの詳細を入力します。
   - アカウント ID: 上記で取得した New Relic アカウント ID を入力します。
   - API キー: 上記の API キーを入力します。
   - New Relic アカウントの場所に基づいて、米国または欧州のリージョンを選択します。
5. 追加をクリックします。

ステップ 2: を有効にする

特定のエージェントスペースで New Relic をアクティブ化し、適切なスコープを設定する

設定

1. エージェントスペースページからエージェントスペースを選択し、表示の詳細を押します (エージェントスペースをまだ作成していない場合は、「」を参照してくださいthe section called “エージェントスペースの作成”)。
2. 機能タブを選択する
3. Telemetry セクションまで下にスクロールします。
4. 追加を押す
5. New Relic を選択する
6. 次へ
7. 確認して保存を押します
8. Webhook URL と API キーのコピー

ステップ 3: ウェブフックを設定する

Webhook URL と API キーを使用して、New Relic を設定して、アラームなどから調査をトリガーするようにイベントを送信できます。ウェブフックの設定の詳細については、「変更追跡ウェブフック」を参照してください。

DevOps エージェントによって送信されるイベントを使用できるようにするには、ウェブフックに送信されるデータが以下に指定されたデータスキーマと一致していることを確認してください。このスキーマに一致しないイベントは、DevOps エージェントによって無視される場合があります。

メソッドとヘッダーを設定する

```
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      "Authorization": "Bearer <Token>",
    },
```

本文を JSON 文字列として送信します。

```
{
    eventType: 'incident';
    incidentId: string;
    action: 'created' | 'updated' | 'closed' | 'resolved';
```

```
    priority: "CRITICAL" | "HIGH" | "MEDIUM" | "LOW" | "MINIMAL";
    title: string;
    description?: string;
    timestamp?: string;
    service?: string;
    // The original event generated by service is attached here.
    data?: object;
}
```

New Relic https://newrelic.com/instant-observability/webhook-notifications を使用してウェブフックを送信します。認可タイプにベアラートークンを選択するか、認可を選択せずに をカスタムヘッダーAuthorization: Bearer <Token>として追加できます。

詳細については、https://docs.newrelic.com/docs/agentic-ai/mcp/overview/ を参照してください。

削除

テレメトリソースは、エージェントスペースレベルとアカウントレベルで 2 つのレベルで接続されます。これを完全に削除するには、まずそれを使用するすべてのエージェントスペースから削除してから、登録を解除する必要があります。

ステップ 1: エージェントスペースから削除する

1. エージェントスペースページからエージェントスペースを選択し、詳細の表示を押します。
2. 機能タブを選択する
3. Telemetry セクションまで下にスクロールします。
4. New Relic を選択する
5. 削除を押す

ステップ 2: アカウントから登録解除する

1. 機能プロバイダーページに移動する (サイドナビゲーションからアクセス可能)
2. 現在登録されているセクションにスクロールします。
3. エージェントスペース数がゼロであることを確認します (他のエージェントスペースで上記のステップ 1 を繰り返しない場合) 。
4. New Relic の横にある登録解除を押します

Splunk の接続

組み込みの 1 方向統合

現在、 AWS DevOps Agent は、組み込みの 1 方向統合で Splunk ユーザーをサポートし、以下を有効にします。

- 自動調査トリガー - Splunk イベントは、 AWS DevOps Agent ウェブフックを介して AWS DevOps Agent インシデント解決調査をトリガーするように設定できます。
- テレメトリイントロスペクション - AWS DevOps Agent は、各プロバイダーのリモート MCP サーバーを介して問題を調査する際に、Splunk テレメトリをイントロスペクションできます。

前提条件

Splunk API トークンの取得

Splunk を接続するには、MCP URL とトークンが必要です。

Splunk 管理者のステップ

Splunk 管理者は、次の手順を実行する必要があります。

- REST API アクセスを有効にする
- デプロイでトークン認証を有効にします。
- 新しいロール「mcp_user」を作成すると、新しいロールには機能は必要ありません。
- MCP サーバーの使用が許可されているデプロイのすべてのユーザーにロール 'mcp_user' を割り当てます。
- ユーザー自身にトークンを作成するアクセス許可がない場合は、対象者を「mcp」として承認されたユーザーのトークンを作成し、適切な有効期限を設定します。

Splunk ユーザーステップ

Splunk ユーザーは、次のステップを実行する必要があります。

- アクセス許可がある場合は、Splunk 管理者から適切なトークンを取得するか、自分でトークンを作成します。トークンの対象者は「mcp」である必要があります。

オンボーディング

ステップ 1: 接続する

アカウントアクセス認証情報を使用して Splunk リモート MCP エンドポイントへの接続を確立する

設定

1. 機能プロバイダーページに移動する (サイドナビゲーションからアクセス可能)
2. Telemetry の「利用可能なプロバイダー」セクションで Splunk を検索し、「登録」をクリックします。
3. Splunk MCP サーバーの詳細を入力します。
   - サーバー名 - 一意の識別子 (my-splunk-server など)
   - エンドポイント URL - Splunk MCP サーバーエンドポイント:

```
https://<YOUR_SPLUNK_DEPLOYMENT_NAME>.api.scs.splunk.com/
<YOUR_SPLUNK_DEPLOYMENT_NAME>/mcp/v1/
```

- 説明 - オプションのサーバーの説明
- トークン名 - 認証用のベアラートークンの名前。 `my-splunk-token`
- トークン値 認証のベアラートークン値

ステップ 2: を有効にする

特定のエージェントスペースで Splunk をアクティブ化し、適切なスコープを設定する

設定

1. エージェントスペースページからエージェントスペースを選択し、表示の詳細を押します (エージェントスペースをまだ作成していない場合は、「」を参照してくださいthe section called “エージェントスペースの作成”)。
2. 機能タブを選択する
3. Telemetry セクションまで下にスクロールします。
4. 追加を押す
5. Splunk を選択する
6. 次へ

7. 確認して Save キーを押します。

8. Webhook URL と API キーをコピーする

ステップ 3: ウェブフックを設定する

Webhook URL と API キーを使用して、 アラームなどから調査をトリガーするイベントを送信するように Splunk を設定できます。

DevOps エージェントによって送信されるイベントを使用できるようにするには、ウェブフックに送信されるデータが以下で指定されたデータスキーマと一致していることを確認してください。このスキーマに一致しないイベントは、DevOps エージェントによって無視される場合があります。

メソッドとヘッダーを設定する

```
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      "Authorization": "Bearer <Token>",
    },
```

本文を JSON 文字列として送信します。

```
{
    eventType: 'incident';
    incidentId: string;
    action: 'created' | 'updated' | 'closed' | 'resolved';
    priority: "CRITICAL" | "HIGH" | "MEDIUM" | "LOW" | "MINIMAL";
    title: string;
    description?: string;
    timestamp?: string;
    service?: string;
    // The original event generated by service is attached here.
    data?: object;
}
```

Splunk https://help.splunk.com/en/splunk-enterprise/alert-and-respond/alerting-manual/9.4/configure-alert-actions/use-a-webhook-alert-action でウェブフックを送信する (承認なしで選択し、代わりにカスタムヘッダーオプションを使用することに注意してください)

詳細はこちら:

- Splunk の MCP サーバードキュメント: https://help.splunk.com/en/splunk-cloud-platform/mcp-server-for-splunk-platform/about-mcp-server-for-splunk-platform
- Splunk Cloud Platform REST API のアクセス要件と制限: https://docs.splunk.com/Documentation/SplunkCloud/latest/RESTTUT/RESTandCloud
- Splunk Cloud Platform で認証トークンを管理する: https://help.splunk.com/en/splunk-cloud-platform/administer/manage-users-and-security/9.3.2411/authenticate-into-the-splunk-platform-with-tokens/manage-or-delete-authentication-tokens
- Splunk Web を使用したロールの作成と管理: https://docs.splunk.com/Documentation/SplunkCloud/latest/Security/Addandeditroles

削除

テレメトリソースは、エージェントスペースレベルとアカウントレベルで 2 つのレベルで接続されます。完全に削除するには、最初にそれを使用するすべてのエージェントスペースから削除してから、登録を解除する必要があります。

ステップ 1: エージェントスペースから削除する

1. エージェントスペースページからエージェントスペースを選択し、詳細の表示を押します。
2. 機能タブを選択する
3. Telemetry セクションまで下にスクロールします。
4. Splunk を選択する
5. 削除を押す

ステップ 2: アカウントから登録解除する

1. 機能プロバイダーページに移動する (サイドナビゲーションからアクセス可能)
2. 現在登録されているセクションにスクロールします。
3. エージェントスペース数がゼロであることを確認します (他のエージェントスペースで上記のステップ 1 を繰り返しない場合) 。
4. Splunk の横にある登録解除を押します

チケット発行とチャットへの接続

AWS DevOps エージェントは、チームの既存のコミュニケーションチャネルに参加することで、チームのメンバーとして機能するように設計されています。DevOps エージェントを ServiceNow や PagerDuty などのチケット発行およびアラームシステムに接続して、インシデントチケットから調査を自動的に開始し、既存のワークフロー内のインシデント対応を加速して、意図した復旧時間 (MTTR) を短縮できます。DevOps エージェントを Slack などのチームコラボレーションシステムに接続して、チャットチャネルで DevOps エージェントからアクティビティの概要を受信することもできます。

チケット発行とチャット統合の接続については、以下を参照してください。

- the section called “PagerDuty の接続”
- the section called “ServiceNow の接続”
- the section called “Slack の接続”

PagerDuty の接続

PagerDuty の統合により、 AWS DevOps Agent はインシデント調査と自動対応中に PagerDuty アカウントからインシデントデータ、オンコールスケジュール、サービス情報にアクセスして更新できます。この統合では、OAuth 2.0 を使用して安全な認証を行います。

> **Important**
>
> AWS DevOps エージェントは、新しい PagerDuty OAuth 2.0 (スコープ付き OAuth) のみをサポートします。リダイレクト uri を使用するレガシー PagerDuty OAuth はサポートされていません。

PagerDuty の要件

PagerDuty を接続する前に、以下を確認してください。

- OAuth クライアント ID とクライアントシークレットを持つ PagerDuty アカウント
- PagerDuty アカウントのサブドメイン (たとえば、PagerDuty URL が の場合`https://your-company.pagerduty.com`、サブドメインは `your-company`)

PagerDuty の登録

PagerDuty は AWS アカウントレベルで登録され、そのアカウントのすべてのエージェントスペース間で共有されます。

ステップ 1: PagerDuty でアクセスを設定する

1. AWS マネジメントコンソールにサインインする
2. AWS DevOps エージェントコンソールに移動する
3. 機能プロバイダーページに移動する (サイドナビゲーションからアクセス可能)
4. 「コミュニケーション」の「利用可能なプロバイダー」セクションで PagerDuty を検索し、「登録」をクリックします。
5. PagerDuty の「アクセスの設定」ページのガイド付きセットアップに従います。

サービスリージョンとサブドメインを確認します。

- アカウントスコープ – PagerDuty リージョン (米国または欧州) を選択し、PagerDuty サブドメインを入力します。たとえば、PagerDuty URL が の場合`https://your-company.pagerduty.com`、 と入力します`your-company`。

PagerDuty で新しいアプリを作成します。

- 別のブラウザタブで PagerDuty にログインし、統合 > アプリ登録に移動します。
- OAuth 2.0 スコープド OAuth を使用して新しいアプリケーションを作成する
- 「アクセス許可」で、最低限必要なスコープ `incidents.read`、`incidents.write`、および を付与します。 `services.read`
- イベント統合を有効にして、 AWS DevOps Agent と PagerDuty 間の双方向通信を許可する

OAuth 認証情報を設定します。

- アクセス許可の範囲 – 最低限必要な範囲は、`incidents.read`、`incidents.write`、 です。`services.read`
- クライアント名 – OAuth クライアントのわかりやすい名前を入力します。
- クライアント ID – PagerDuty アプリ登録から OAuth クライアント ID を入力します。
- クライアントシークレット – PagerDuty アプリ登録から OAuth クライアントシークレットを入力します。

ステップ 2: PagerDuty 登録を確認して送信する

1. PagerDuty 設定の詳細をすべて確認する
2. 送信をクリックして登録を完了します
3. 登録が成功すると、PagerDuty は機能プロバイダーページの現在登録されているセクションに表示されます。

エージェントスペースへの PagerDuty の追加

PagerDuty をアカウントレベルで登録したら、個々のエージェントスペースに接続できます。

1. AWS DevOps エージェントコンソールで、エージェントスペースを選択します。
2. 機能タブに移動する
3. 「コミュニケーション」セクションで、「追加」をクリックします。
4. 利用可能なプロバイダーのリストから PagerDuty を選択する
5. 保存 をクリックします。

PagerDuty 接続の管理

- 認証情報の更新 – OAuth 認証情報を更新する必要がある場合は、機能プロバイダーページから PagerDuty を登録解除し、新しい認証情報で再登録します。
- 接続の表示 – AWS DevOps エージェントコンソールで、エージェントスペースを選択し、機能タブに移動して、接続された通信統合を表示します。
- PagerDuty の削除 – PagerDuty をエージェントスペースから切断するには、コミュニケーションセクションでそれを選択し、削除をクリックします。登録を完全に削除するには、まずすべてのエージェントスペースから削除してから、機能プロバイダーページから登録を解除します。

Webhook のサポート

AWS DevOps エージェントは PagerDuty V3 ウェブフックのみをサポートします。以前のウェブフックバージョンはサポートされていません。

PagerDuty V3 ウェブフックサブスクリプションの詳細については、PagerDuty デベロッパードキュメントの「Webhooks Overview」を参照してください。

ServiceNow の接続

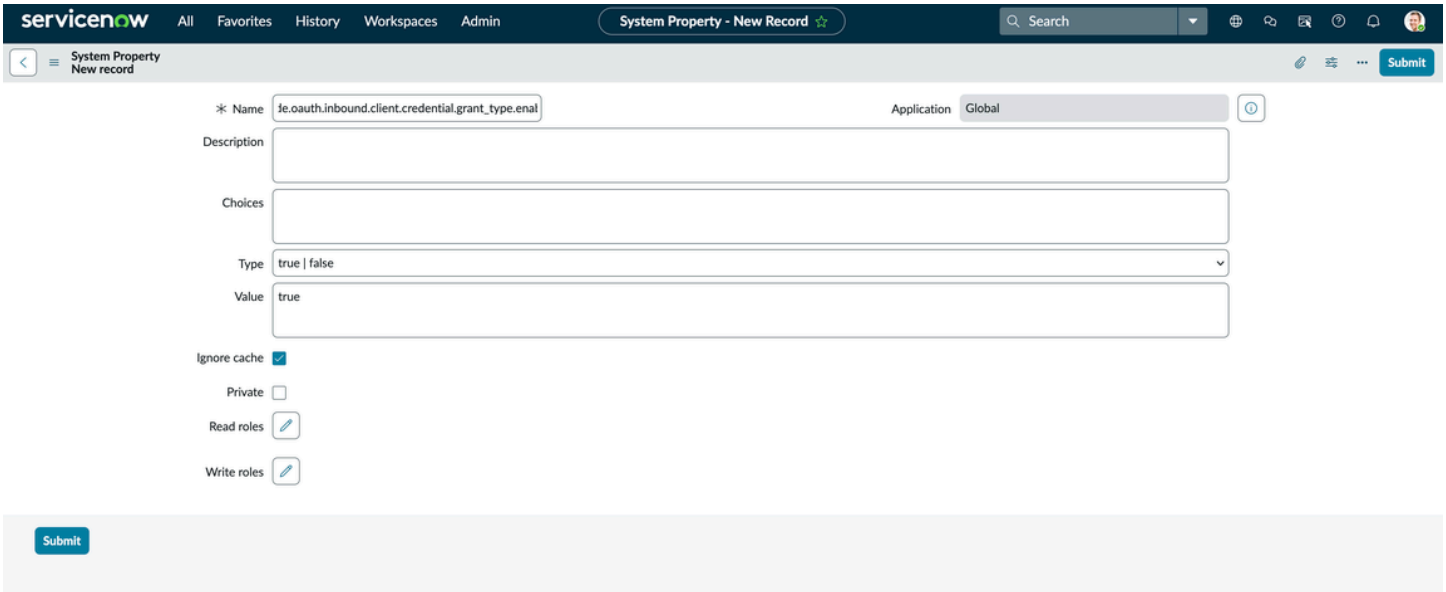
このチュートリアルでは、ServiceNow インスタンスを AWS DevOps エージェントに接続して、チケットの作成時にインシデント対応調査を自動的に開始し、主要な検出結果を元のチケットに投稿できるようにします。また、特定のチケットのみを DevOps エージェントスペースに送信するように ServiceNow インスタンスを設定する方法と、複数の DevOps エージェントスペース間でチケットルーティングを調整する方法の例も含まれています。

初期セットアップ

最初のステップは、 AWS DevOps が ServiceNow インスタンスにアクセスするために使用できる OAuth アプリケーションクライアントを ServiceNow に作成することです。

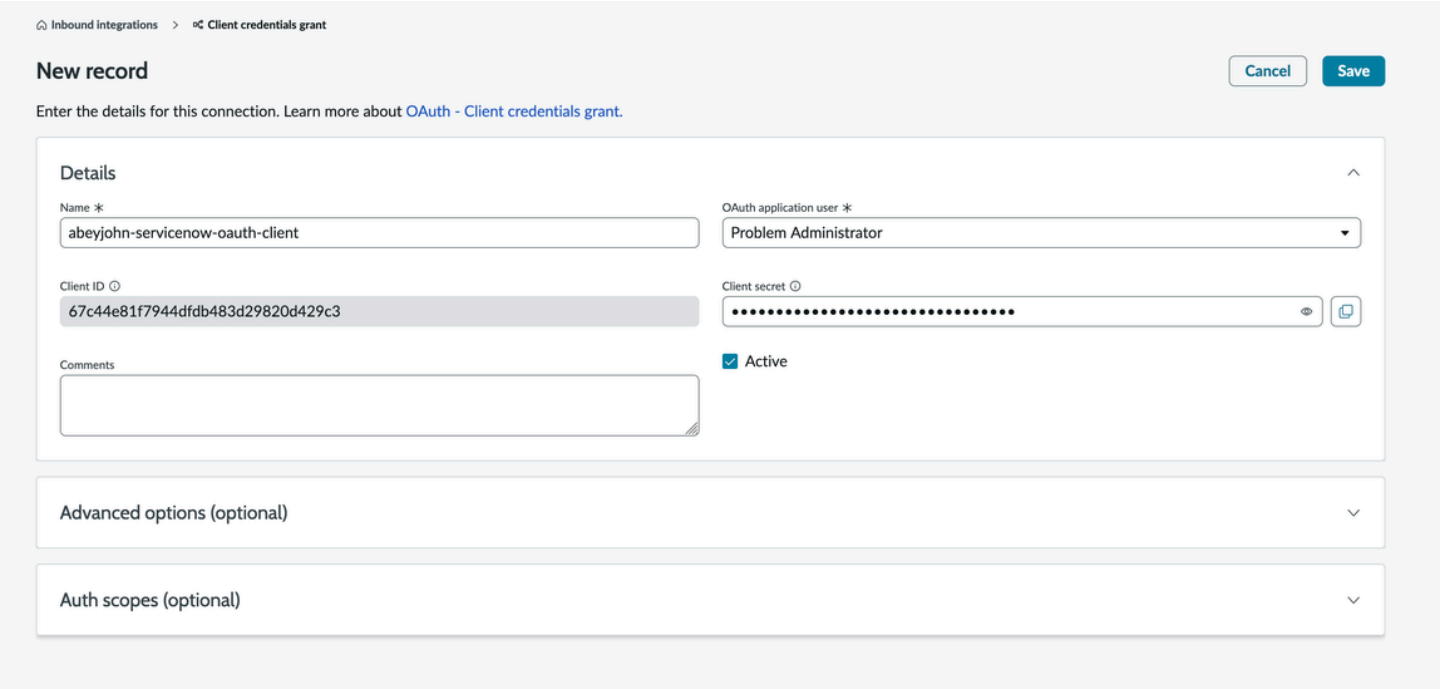
ServiceNow OAuth アプリケーションクライアントを作成する

1. インスタンスのクライアント認証情報システムプロパティを有効にする
   a. フィルター検索ボックス`sys_properties.list`で検索し、Enter キーを押します (オプションは表示されませんが、Enter キーを押すと機能します) 。
   b. 新規を選択する
   c. 名前を `glide.oauth.inbound.client.credential.grant_type.enabled`、値を true に追加し、型を true | false



1. フィルター検索ボックスから System OAuth > Application Registry に移動します。

2. 「新規」 >「新しいインバウンド統合エクスペリエンス」 >「新しい統合」 >OAuth - クライアント認証情報付与」を選択します。
3. 名前を選択し、OAuth アプリケーションユーザーを「問題管理者」に設定し、「保存」をクリックします。

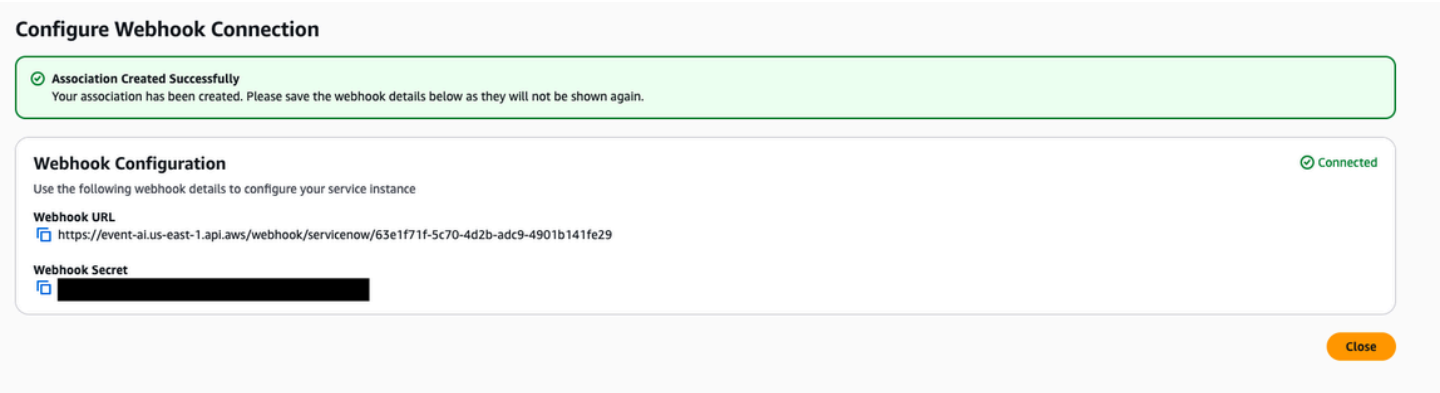


ServiceNow OAuth クライアントを AWS DevOps エージェントに接続する

1. このプロセスは 2 つの場所で開始できます。まず、Capability Providers ページに移動し、Communication で ServiceNow を見つけ、Register をクリックします。または、作成した DevOps エージェントスペースを選択して、機能 → 通信 → 追加 → ServiceNow に移動し、登録をクリックします。
2. 次に、先ほど作成した OAuth アプリケーションクライアントを使用して、DevOps Agent が ServiceNow インスタンスにアクセスすることを許可します。

- 次のステップに従い、ウェブフックに関する結果情報を保存します。

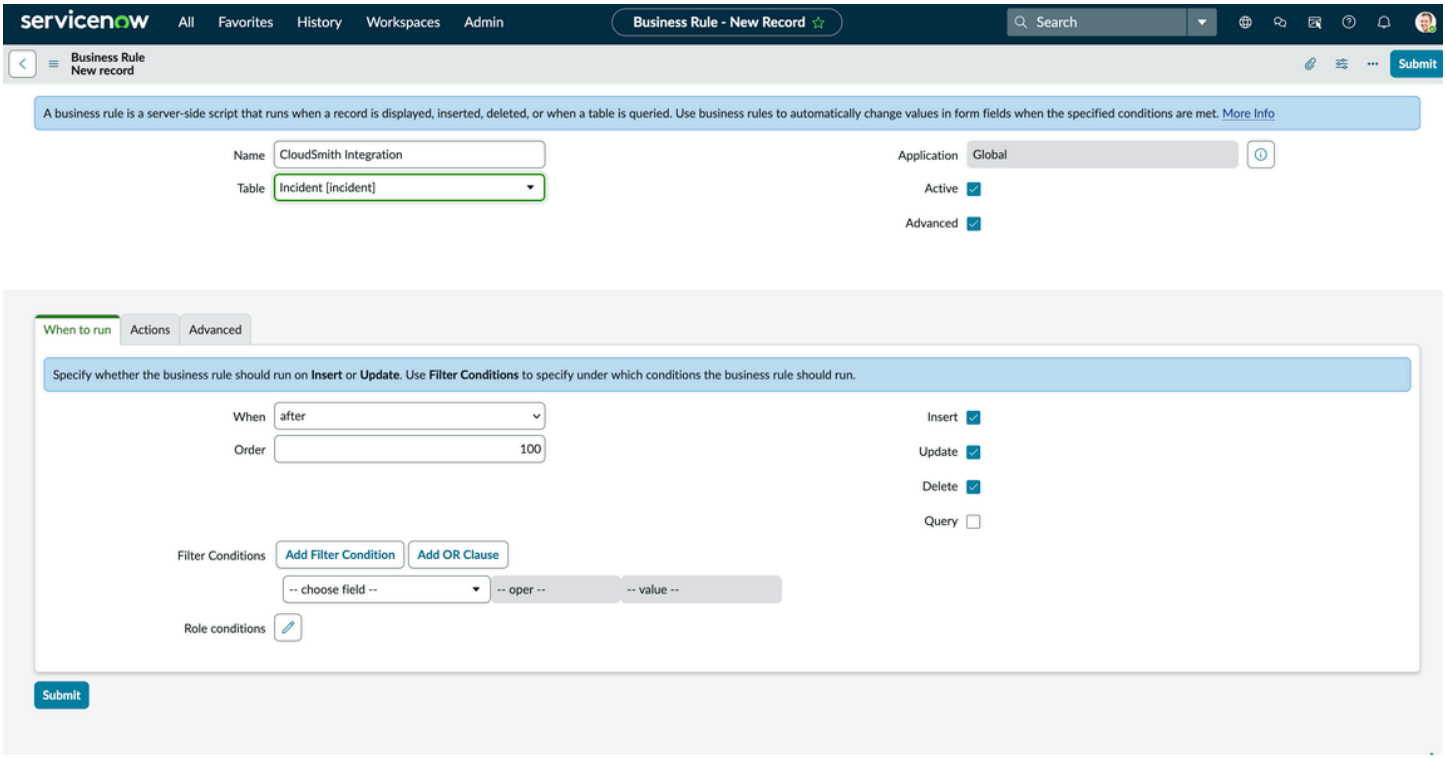
> **Important**
>
> この情報は再度表示されません



ServiceNow ビジネスルールを設定する

接続を確立したら、ServiceNow でビジネスルールを設定して、DevOps エージェントスペース (複数可) にチケットを送信する必要があります。

1. アクティビティサブスクリプション → 管理 → ビジネスルールに移動し、新規をクリックします。
2. 「Table」フィールドを「Incident [incident]」に設定し、「Advanced」チェックボックスをオンにして、挿入、更新、削除後にルールを実行するように設定します。



1. 「詳細」タブに移動し、次のウェブフックスクリプトを追加し、ウェブフックシークレットと URL を指定された場所に挿入して、送信をクリックします。

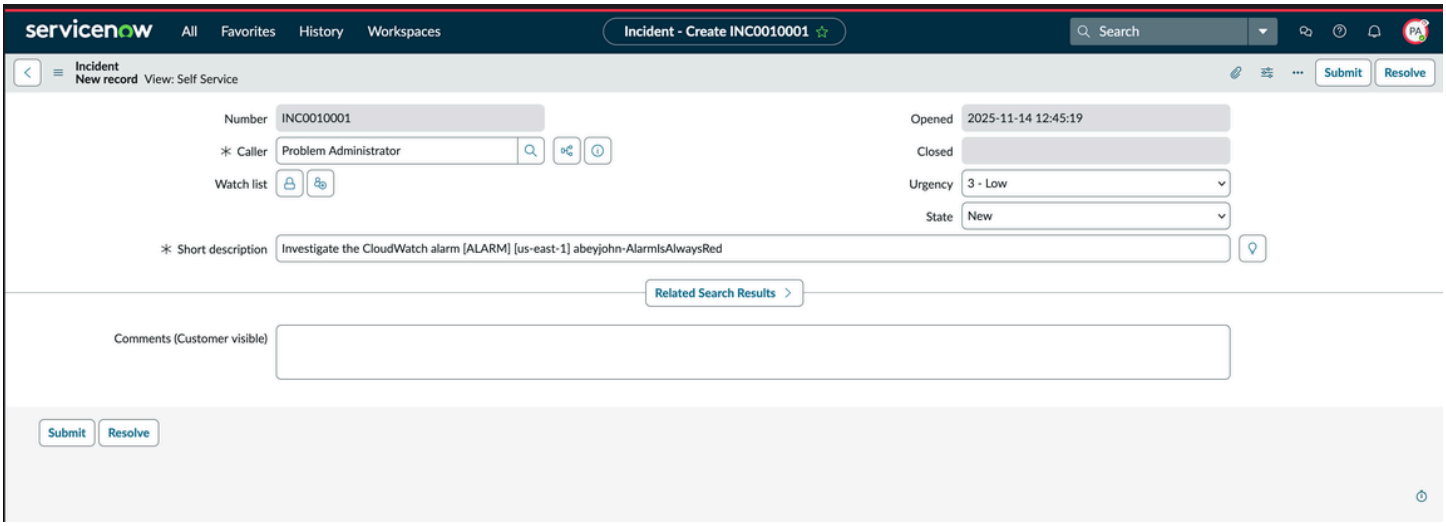
```
(function executeRule(current, previous /*null when async*/ ) {

    var WEBHOOK_CONFIG = {
        webhookSecret: GlideStringUtil.base64Encode('<<< INSERT WEBHOOK SECRET HERE
 >>>'),
        webhookUrl: '<<< INSERT WEBHOOK URL HERE >>>'
    };

    function generateHMACSignature(payloadString, secret) {
        try {
            var mac = new GlideCertificateEncryption();
            var signature = mac.generateMac(secret, "HmacSHA256", payloadString);
            return signature;
        } catch (e) {
            gs.error('HMAC generation failed: ' + e);
            return null;
        }
    }
```

```
    function callWebhook(payload, config) {
        try {
            var timestamp = new Date().toISOString();
            var payloadString = JSON.stringify(payload);
            var payloadWithTimestamp =`${timestamp}:${payloadString}`;

            var signature = generateHMACSignature(payloadWithTimestamp,
config.webhookSecret);

            if (!signature) {
                gs.error('Failed to generate signature');
                return false;
            }

            gs.info('Generated signature: ' + signature);

            var request = new sn_ws.RESTMessageV2();
            request.setEndpoint(config.webhookUrl);
            request.setHttpMethod('POST');

            request.setRequestHeader('Content-Type', 'application/json');
            request.setRequestHeader('x-amzn-event-signature', signature);
            request.setRequestHeader('x-amzn-event-timestamp', timestamp);

            request.setRequestBody(payloadString);

            var response = request.execute();
            var httpStatus = response.getStatusCode();
            var responseBody = response.getBody();

            if (httpStatus >= 200 && httpStatus < 300) {
                gs.info('Webhook sent successfully. Status: ' + httpStatus);
                return true;
            } else {
                gs.error('Webhook failed. Status: ' + httpStatus + ', Response: ' +
responseBody);
                return false;
            }

        } catch (ex) {
            gs.error('Error sending webhook: ' + ex.getMessage());
            return false;
        }
    }
```

```
    function createReference(field) {
        if (!field || field.nil()) {
            return null;
        }

        return {
            link: field.getLink(true),
            value: field.toString()
        };
    }

    function getStringValue(field) {
        if (!field || field.nil()) {
            return null;
        }
        return field.toString();
    }

    function getIntValue(field) {
        if (!field || field.nil()) {
            return null;
        }
        var val = parseInt(field.toString());
        return isNaN(val) ? null : val;
    }

    var eventType = (current.operation() == 'insert') ? "create" : "update";

    var incidentEvent = {
        eventType: eventType.toString(),
        sysId: current.sys_id.toString(),
        priority: getStringValue(current.priority),
        impact: getStringValue(current.impact),
        active: getStringValue(current.active),
        urgency: getStringValue(current.urgency),
        description: getStringValue(current.description),
        shortDescription: getStringValue(current.short_description),
        parent: getStringValue(current.parent),
        incidentState: getStringValue(current.incident_state),
        severity: getStringValue(current.severity),
        problem: createReference(current.problem),
        additionalContext: {}
    };
```

```
    incidentEvent.additionalContext = {
        number: current.number.toString(),
        opened_at: getStringValue(current.opened_at),
        opened_by: current.opened_by.nil() ? null :
 current.opened_by.getDisplayValue(),
        assigned_to: current.assigned_to.nil() ? null :
 current.assigned_to.getDisplayValue(),
        category: getStringValue(current.category),
        subcategory: getStringValue(current.subcategory),
        knowledge: getStringValue(current.knowledge),
        made_sla: getStringValue(current.made_sla),
        major_incident: getStringValue(current.major_incident)
    };

    for (var key in incidentEvent.additionalContext) {
        if (incidentEvent.additionalContext[key] === null) {
            delete incidentEvent.additionalContext[key];
        }
    }

    gs.info(JSON.stringify(incidentEvent, null, 2)); // Pretty print for logging only

    if (WEBHOOK_CONFIG.webhookUrl && WEBHOOK_CONFIG.webhookSecret) {
        callWebhook(incidentEvent, WEBHOOK_CONFIG);
    } else {
        gs.info('Webhook not configured.');
    }

})(current, previous);
```

機能プロバイダーページから ServiceNow 接続を登録する場合は、ServiceNow インシデントチケットを調査する DevOps エージェントスペースに移動し、機能 → 通信を選択し、機能プロバイダーページで登録した ServiceNow インスタンスを登録する必要があります。これで、すべてを設定する必要があり、発信者が「問題管理者」に設定されているすべてのインシデント (AWS DevOps OAuth クライアントに付与したアクセス許可を模倣するため) は、設定された DevOps エージェントスペースでインシデント対応調査をトリガーします。これをテストするには、ServiceNow で新しいインシデントを作成し、インシデントの発信者フィールドを「問題管理者」に設定します。



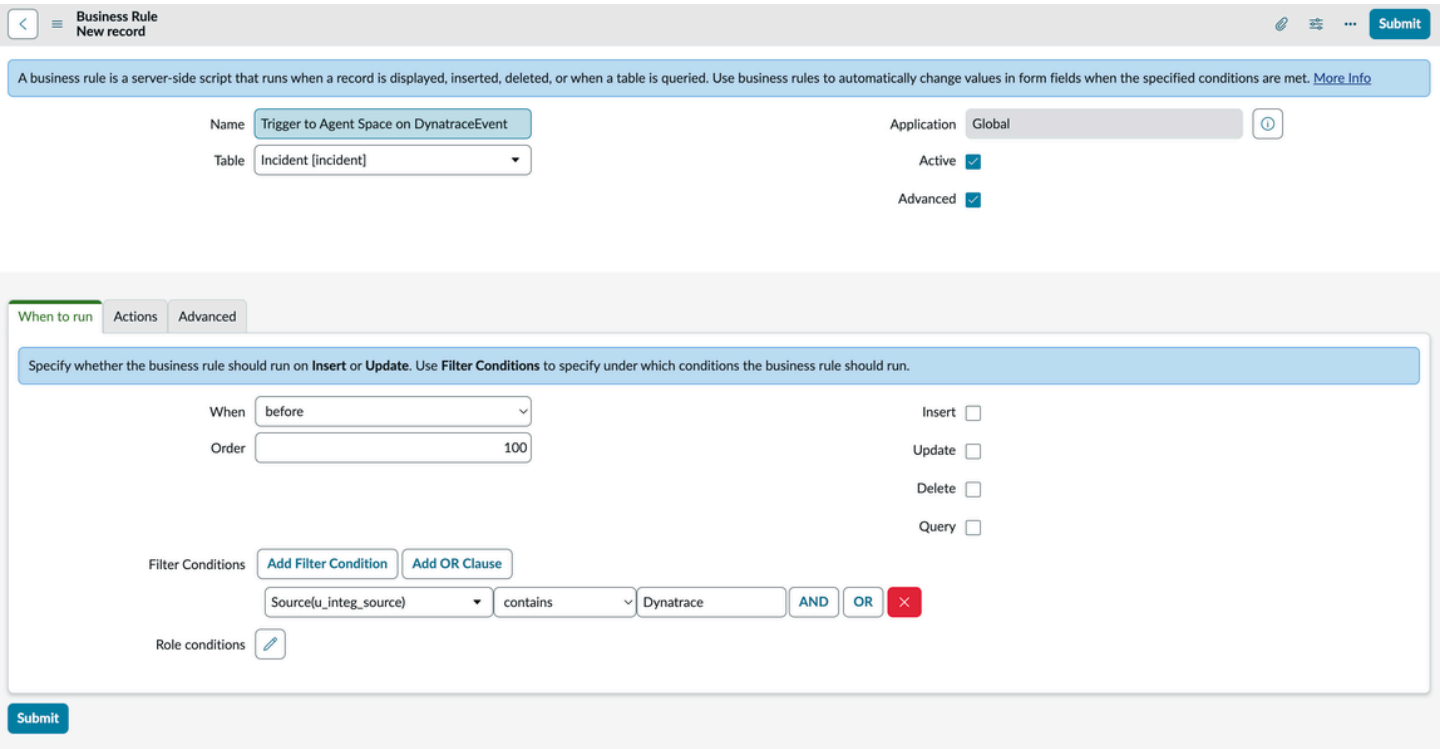
ServiceNow チケットの更新

トリガーされたすべてのインシデント対応調査中、DevOps エージェントは主要な検出結果、根本原因分析、緩和計画の更新を元のチケットに提供します。エージェントの検出結果はインシデントのコメントに投稿され、現在、タイプ `finding`、、`causeinvestigation_summary`、`mitigation_summary`、および調査ステータスの更新 (例: `AWS DevOps Agent started/finished its investigation`) のエージェントレコードのみを投稿します。

チケットルーティングとオーケストレーションの例

シナリオ: DevOps エージェントスペースに送信されるインシデントのフィルタリング

これは簡単なシナリオですが、インシデントソースを追跡するために ServiceNow でフィールドを作成するには、ServiceNow でいくつかの設定が必要です。この例では、SNOW フォームビルダーを使用して新しいソース (u_source) フィールドを作成します。これにより、インシデントソースを追跡し、それを使用して特定のソースから DevOps エージェントスペースにリクエストをルーティングできます。ルーティングは、Service Now ビジネスルールを作成し、「When」トリガーと「Filter Conditions」を設定するタブで実行します。この例では、フィルター条件を次のように設定します。

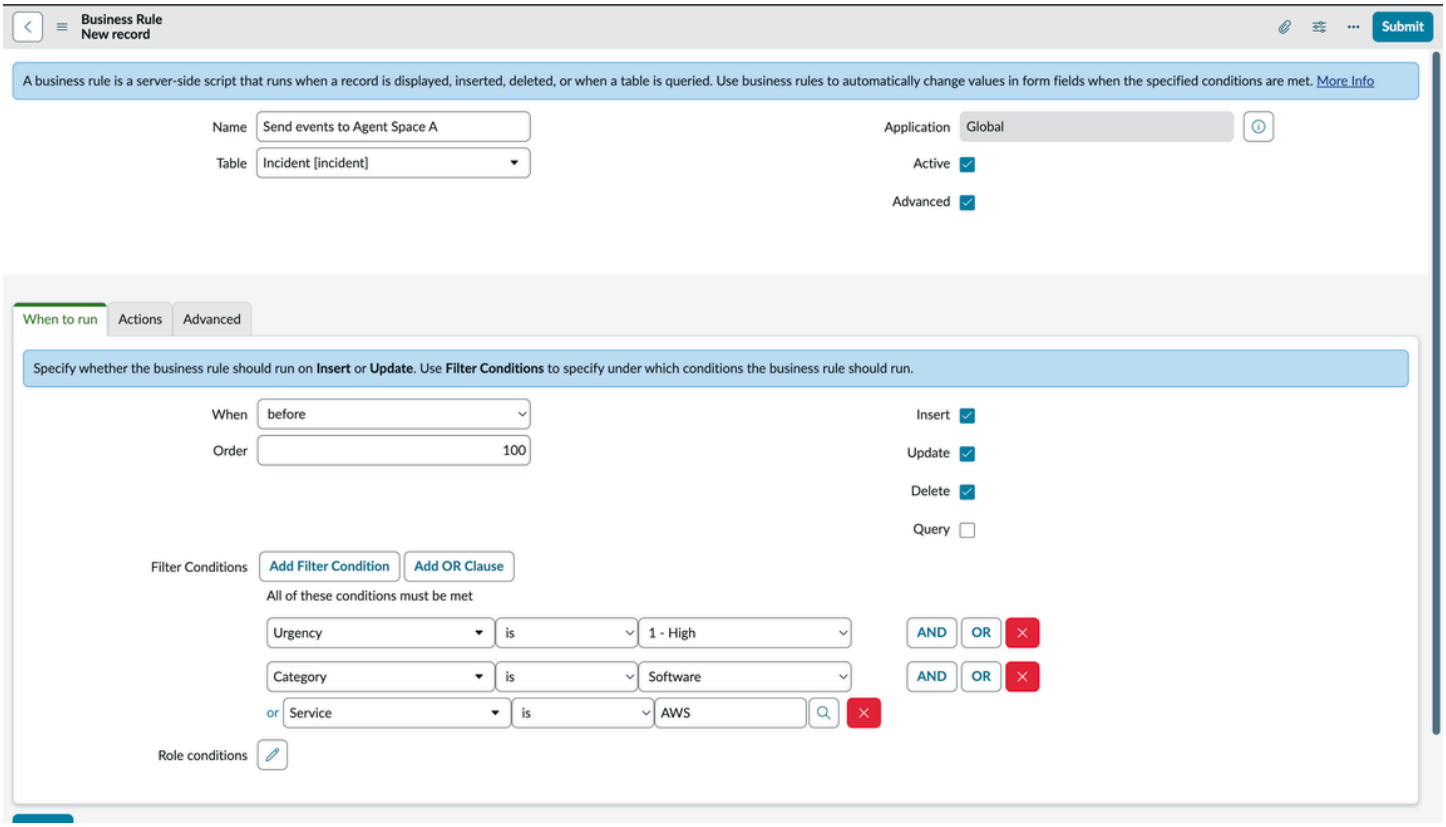


シナリオ: 複数の DevOps エージェントスペースにインシデントをルーティングする

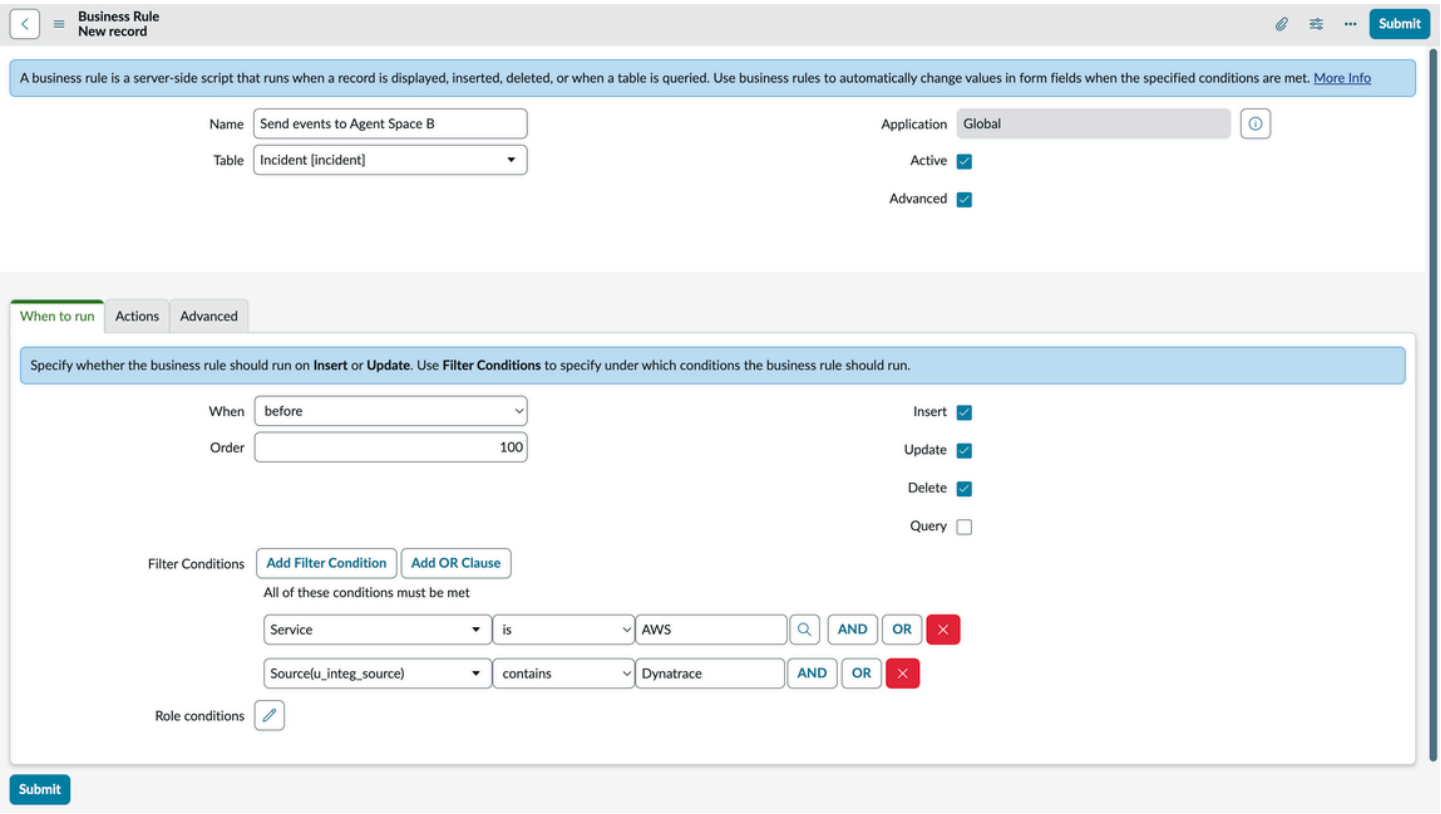
この例では、緊急度が 、カテゴリが 1、またはサービスが の場合に DevOps エージェントスペース B で調査をトリガーしAWS、サービスが Software AWS、ソースが の場合に DevOps エージェントスペース A で調査をトリガーする方法を示しますDynatrace。

このシナリオは 2 つの方法で実現できます。ウェブフックスクリプト自体を更新して、このビジネスロジックを含めることができます。このシナリオでは、ServiceNow ビジネスルールを使用してこれを実現する方法を示し、透明性を高め、デバッグを簡素化します。ルーティングは、2 つの Service Now ビジネスルールを作成することで実現されます。

- ServiceNow for DevOps エージェントスペース A でビジネスルールを作成し、条件ビルダーを使用して条件を作成し、指定した条件に基づいてのみイベントを送信します。



- 次に、ServiceNow for AgentSpace B で別のビジネスルールを作成します。このビジネスルールは、サービスが AWS でソースが Dynatrace の場合にのみトリガーされます。



これで、指定された条件に一致する新しいインシデントを作成すると、DevOps エージェントスペース A または DevOps エージェントスペース B の調査がトリガーされ、インシデントルーティングをきめ細かく制御できるようになります。

Slack の接続

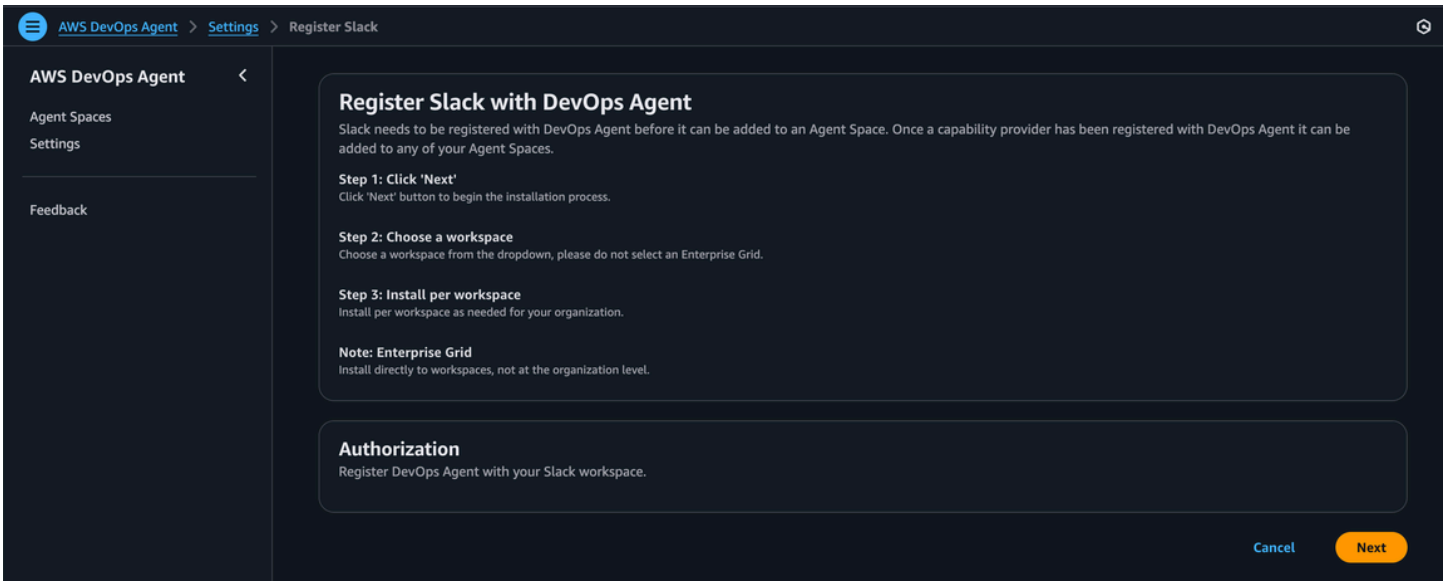
インシデント対応調査の主要な検出結果、根本原因分析、生成された緩和計画を使用して、選択した Slack チャネルを更新するように AWS DevOps エージェントを設定できます。

[開始する前に]

エージェントスペースに追加する前に、Slack を DevOps エージェントに登録する必要があります。AWS DevOps エージェントを Slack と統合するには、次の要件を満たす必要があります。

- サードパーティーアプリケーションをインストールして認可できる Slack ワークスペースにアクセスできる
- AWS DevOps Agent が通知を送信する Slack チャネルを特定している

Slack と AWS DevOps エージェントの統合を登録する



1. AWS DevOps エージェントコンソールの Capability Providers ページから、「Communication」の「Available providers」セクションで Slack を見つけ、「Register」をクリックします。
2. 登録ボタンを選択します。
3. ワークスペースの AWS DevOps エージェントアプリケーションを承認するには、Slack にリダイレクトされます。
4. Slack 認可ページで、組織レベルではなくワークスペースに直接 をインストールします。
5. ドロップダウンからワークスペースを選択します。エンタープライズグリッドを選択しないでください。
6. 組織の必要に応じてワークスペースごとに をインストールします。
7. リクエストされたスコープを確認し、統合を許可するをクリックします。
8. 認可後、 AWS DevOps エージェントコンソールに戻ります。

Slack を DevOps エージェントスペースに関連付ける (複数可)

DevOps エージェントスペースに Slack を登録したら、それを DevOps エージェントスペース (複数可) に関連付けることができます。

1. 設定した AgentSpace の Capabilities タブから、Communications > Slack に移動します。
2. Slack の追加を選択する
3. チャネル ID を入力する

4. Create を選択して Slack 設定を完了します。

> **Note**
>
> エージェントのボットユーザーは、メッセージを投稿する前にプライベートチャネルに追加する必要があります。

> **Important**
>
> Slack アプリをアンインストールすると、Slack アプリを再インストールできなくなる可能性があります。Slack アプリをアンインストールしないでください。

Webhook による DevOps エージェントの呼び出し

Webhook を使用すると、外部システムが AWS DevOps エージェント調査を自動的にトリガーできます。これにより、インシデントが発生したときに HTTP リクエストを送信できるチケットシステム、モニタリングツール、およびその他のプラットフォームとの統合が可能になります。

前提条件

ウェブフックアクセスを設定する前に、以下があることを確認してください。

- AWS DevOps エージェントで設定されたエージェントスペース
- AWS DevOps エージェントコンソールへのアクセス
- ウェブフックリクエストを送信する外部システム

ウェブフックタイプ

AWS DevOps エージェントは、次のタイプのウェブフックをサポートしています。

- 統合固有のウェブフック – Dynatrace、Splunk、Datadog、New Relic、ServiceNow、Slack などのサードパーティー統合を設定すると自動的に生成されます。これらのウェブフックは特定の統合に関連付けられ、統合タイプによって決定される認証方法を使用します。

- 汎用ウェブフック – 特定の統合の対象ではないソースから調査をトリガーするために手動で作成できます。汎用ウェブフックは現在 HMAC 認証を使用しています (ベアラートークンは現在利用できません)。
- Grafana アラートウェブフック – Grafana は、ウェブフックのコンタクトポイントを通じてアラート通知を AWS DevOps エージェントに直接送信できます。カスタム通知テンプレートを含むセットアップ手順については、「Grafana の接続」を参照してください。

Webhook 認証方法

ウェブフックの認証方法は、関連付けられている統合によって異なります。

HMAC 認証 – 以下によって使用されます。

- Dynatrace 統合ウェブフック
- 一般的なウェブフック (特定のサードパーティー統合にリンクされていない)

ベアラートークン認証 – 以下によって使用されます。

- Splunk 統合ウェブフック
- Datadog 統合ウェブフック
- New Relic 統合ウェブフック
- ServiceNow 統合ウェブフック
- Slack 統合ウェブフック

ウェブフックアクセスの設定

ステップ 1: ウェブフック設定に移動する

1. AWS マネジメントコンソールにサインインし、 AWS DevOps エージェントコンソールに移動します。
2. エージェントスペースを選択する
3. 機能タブに移動する
4. Webhook セクションで、Configure をクリックします。

ステップ 2: ウェブフック認証情報を生成する

統合固有のウェブフックの場合:

サードパーティー統合の設定を完了すると、ウェブフックが自動的に生成されます。ウェブフックエンドポイント URL と認証情報は、統合セットアッププロセスの最後に提供されます。

汎用ウェブフックの場合:

1. ウェブフックの生成 をクリックします。
2. システムは HMAC キーペアを生成します
3. 生成されたキーとシークレットを安全に保存する - 再度取得することはできません
4. 提供されたウェブフックエンドポイント URL をコピーする

ステップ 3: 外部システムを設定する

ウェブフックエンドポイント URL と認証情報を使用して、 AWS DevOps エージェントにリクエストを送信するように外部システムを設定します。特定の設定手順は、外部システムによって異なります。

ウェブフック認証情報の管理

認証情報の削除 – ウェブフック認証情報を削除するには、ウェブフック設定セクションに移動し、削除をクリックします。認証情報を削除すると、ウェブフックエンドポイントは新しい認証情報を生成するまでリクエストを受け入れなくなります。

認証情報の再生成 – 新しい認証情報を生成するには、まず既存の認証情報を削除してから、新しいキーペアまたはトークンを生成します。

ウェブフックの使用

Webhook リクエスト形式

調査をトリガーするには、外部システムがウェブフックエンドポイント URL に HTTP POST リクエストを送信する必要があります。

バージョン 1 (HMAC 認証):

ヘッダー。

- `Content-Type: application/json`

- x-amzn-event-signature: <HMAC signature>
- x-amzn-event-timestamp: <+%Y-%m-%dT%H:%M:%S.000Z>

HMAC 署名は、SHA-256 を使用してシークレットキーでリクエスト本文に署名することで生成されます。

バージョン 2 (ベアラートークン認証):

ヘッダー。

- Content-Type: application/json
- Authorization: Bearer <your-token>

リクエスト本文:

リクエスト本文には、インシデントに関する情報を含める必要があります。

```
json

{
  "title": "Incident title",
  "severity": "high",
  "affectedResources": ["resource-id-1", "resource-id-2"],
  "timestamp": "2025-11-23T18:00:00Z",
  "description": "Detailed incident description",
  "data": {
    "metadata": {
        "region": "us-east-1",
        "environment": "production"
    }
  }
}
```

コードの例

バージョン 1 (HMAC 認証) - JavaScript:

```
const crypto = require('crypto');

// Webhook configuration
const webhookUrl = 'https://your-webhook-endpoint.amazonaws.com/invoke';
```

```
const webhookSecret = 'your-webhook-secret-key';

// Incident data
const incidentData = {
    eventType: 'incident',
    incidentId: 'incident-123',
    action: 'created',
    priority: "HIGH",
    title: 'High CPU usage on production server',
    description: 'High CPU usage on production server host ABC in AWS account 1234
 region us-east-1',
    timestamp: new Date().toISOString(),
    service: 'MyTestService',
    data: {
      metadata: {
        region: 'us-east-1',
        environment: 'production'
      }
    }
};

// Convert data to JSON string
const payload = JSON.stringify(incidentData);
const timestamp = new Date().toISOString();
const hmac = crypto.createHmac("sha256", webhookSecret);
hmac.update(`${timestamp}:${payload}`, "utf8");
const signature = hmac.digest("base64");

// Send the request
fetch(webhookUrl, {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json',
    'x-amzn-event-timestamp': timestamp,
    'x-amzn-event-signature': signature
  },
  body: payload
})
.then(res => {
  console.log(`Status Code: ${res.status}`);
  return res.text();
})
.then(data => {
  console.log('Response:', data);
```

```
})
.catch(error => {
  console.error('Error:', error);
});
```

バージョン 1 (HMAC 認証) - cURL:

```
#!/bin/bash

# Configuration
WEBHOOK_URL="https://event-ai.us-east-1.api.aws/webhook/generic/YOUR_WEBHOOK_ID"
SECRET="YOUR_WEBHOOK_SECRET"

# Create payload
TIMESTAMP=$(date -u +%Y-%m-%dT%H:%M:%S.000Z)
INCIDENT_ID="test-alert-$(date +%s)"

PAYLOAD=$(cat <<EOF
{
"eventType": "incident",
"incidentId": "$INCIDENT_ID",
"action": "created",
"priority": "HIGH",
"title": "Test Alert",
"description": "Test alert description",
"service": "TestService",
"timestamp": "$TIMESTAMP"
}
EOF
)

# Generate HMAC signature
SIGNATURE=$(echo -n "${TIMESTAMP}:${PAYLOAD}" | openssl dgst -sha256 -hmac "$SECRET" -
binary | base64)

# Send webhook
curl -X POST "$WEBHOOK_URL" \
-H "Content-Type: application/json" \
-H "x-amzn-event-timestamp: $TIMESTAMP" \
-H "x-amzn-event-signature: $SIGNATURE" \
-d "$PAYLOAD"
```

バージョン 2 (ベアラートークン認証) - JavaScript:

```
function sendEventToWebhook(webhookUrl, secret) {
  const timestamp = new Date().toISOString();

  const payload = {
    eventType: 'incident',
    incidentId: 'incident-123',
    action: 'created',
    priority: "HIGH",
    title: 'Test Alert',
    description: 'Test description',
    timestamp: timestamp,
    service: 'TestService',
    data: {}
  };

  fetch(webhookUrl, {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      "x-amzn-event-timestamp": timestamp,
      "Authorization": `Bearer ${secret}`,  // Fixed: template literal
    },
    body: JSON.stringify(payload),
  });
}
```

バージョン 2 (ベアラートークン認証) - cURL:

```
#!/bin/bash

# Configuration
WEBHOOK_URL="https://event-ai.us-east-1.api.aws/webhook/generic/YOUR_WEBHOOK_ID"
SECRET="YOUR_WEBHOOK_SECRET"

# Create payload
TIMESTAMP=$(date -u +%Y-%m-%dT%H:%M:%S.000Z)
INCIDENT_ID="test-alert-$(date +%s)"

PAYLOAD=$(cat <<EOF
{
"eventType": "incident",
"incidentId": "$INCIDENT_ID",
"action": "created",
```

```
"priority": "HIGH",
"title": "Test Alert",
"description": "Test alert description",
"service": "TestService",
"timestamp": "$TIMESTAMP"
}
EOF
)

# Send webhook
curl -X POST "$WEBHOOK_URL" \
-H "Content-Type: application/json" \
-H "x-amzn-event-timestamp: $TIMESTAMP" \
-H "Authorization: Bearer $SECRET" \
-d "$PAYLOAD"
```

ウェブフックのトラブルシューティング

200 を受信しない場合

200 とウェブフックが受信したメッセージは、認証に合格し、システムが確認および処理するためにメッセージがキューに入れられたことを示します。200 ではなく 4xx を取得する場合、認証またはヘッダーに問題がある可能性があります。認証のデバッグに役立つ curl オプションを使用して手動で送信してみてください。

200 を受信しても調査が開始されない場合

原因として、ペイロードの変形が考えられます。

1. タイムスタンプとインシデント ID の両方が更新され、一意であることを確認します。重複するメッセージは重複排除されます。
2. メッセージが有効な JSON であることを確認します。
3. 形式が正しいことを確認する

200 を受け取り、調査が直ちにキャンセルされた場合

ほとんどの場合、その月の上限に達しています。必要に応じて AWS 、連絡先に連絡してレート制限の変更を依頼してください。

関連トピック

- the section called “エージェントスペースの作成”
- the section called “DevOps エージェントウェブアプリとは”
- the section called “DevOps エージェント IAM アクセス許可”

AWS DevOps エージェントと Amazon EventBridge の統合

調査および緩和ライフサイクル中に発生するイベントを使用して、 AWS DevOps エージェントをイベント駆動型アプリケーションと統合できます。 AWS DevOps エージェントは、調査または緩和の状態が変化したときにイベントを Amazon EventBridge に送信します。その後、これらのイベントに基づいてアクションを実行する EventBridge ルールを作成できます。

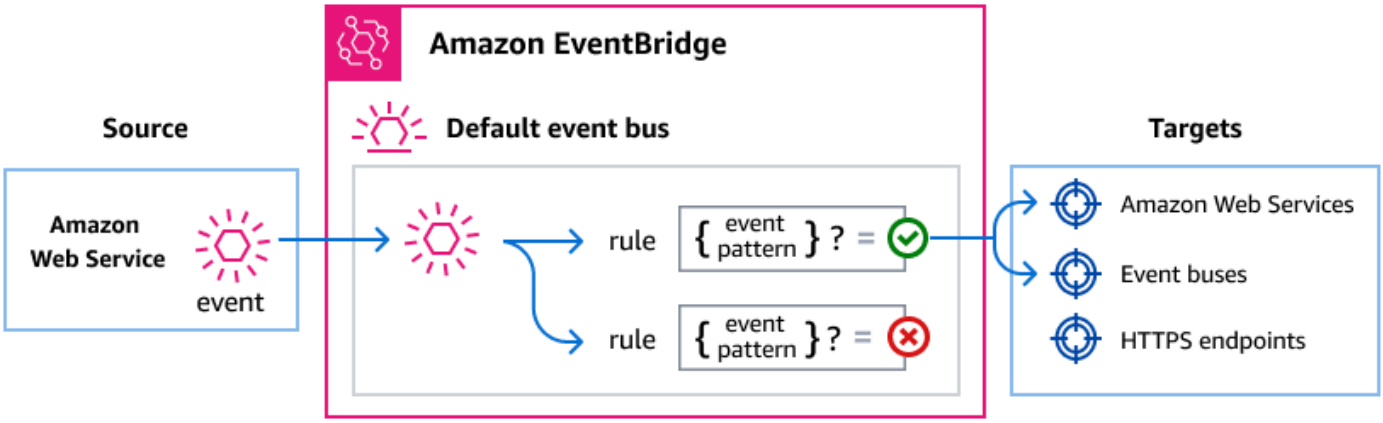
たとえば、次のアクションを実行するルールを作成できます。

- 調査が完了したら、 AWS Lambda 関数を呼び出して調査結果を処理します。
- 調査が失敗またはタイムアウトしたときに Amazon SNS 通知を送信します。
- 新しい調査の作成時にチケットシステムを更新します。
- 緩和アクションが完了したら、 AWS Step Functions ワークフローを開始します。

EventBridge が AWS DevOps エージェントイベントをルーティングする方法

AWS DevOps Agent は EventBridge のデフォルトイベントバスにイベントを送信します。その後、EventBridge は作成したルールに対してイベントを評価します。イベントがルールのイベントパターンと一致すると、EventBridge は指定されたターゲットにイベントを送信します。

次の図は、EventBridge が AWS DevOps エージェントイベントをルーティングする方法を示しています。



1. AWS DevOps Agent は、調査または緩和ライフサイクルの状態が変更されたときに EventBridge のデフォルトイベントバスにイベントを送信します。
2. EventBridge は、作成したルールに対してイベントを評価します。
3. イベントがルールのイベントパターンと一致する場合、EventBridge はルールで指定されたターゲットにイベントを送信します。

AWS DevOps エージェントイベント

AWS DevOps エージェントは、以下のイベントを EventBridge に送信します。すべてのイベントはソース を使用しますaws.aidevops。

サポートされている調査イベント

detail-type	説明
`Investigation Created`	エージェントスペースに調査が作成されました。
`Investigation Priority Updated`	調査の優先度が変更されました。
`Investigation In Progress`	調査がアクティブな分析を開始しました。
`Investigation Completed`	調査は検出結果で正常に終了しました。

detail-type	説明
`Investigation Failed`	調査でエラーが発生しました。完了できませんでした。
`Investigation Timed Out`	調査が最大許容期間を超えました。
`Investigation Cancelled`	調査は完了前にキャンセルされました。
`Investigation Pending Triage`	調査は、アクティブな分析が開始される前にトリアージを待っています。
`Investigation Linked`	調査が関連するインシデントまたはチケットにリンクされました。

サポートされている緩和イベント

detail-type	説明
`Mitigation In Progress`	緩和アクションが開始されました。
`Mitigation Completed`	緩和アクションが正常に終了しました。
`Mitigation Failed`	緩和アクションでエラーが発生しました。完了できませんでした。
`Mitigation Timed Out`	緩和アクションが最大許容期間を超えました。
`Mitigation Cancelled`	緩和アクションは完了前にキャンセルされました。

詳細なフィールドの説明とイベント例については、「」を参照してくださいthe section called “AWS DevOps エージェントイベントの詳細リファレンス”。

AWS DevOps エージェントイベントに一致するイベントパターンの作成

EventBridge ルールは、イベントパターンを使用してイベントを選択し、ターゲットにルーティングします。イベントパターンは、処理するイベントの構造と一致します。イベントパターンを作成して、イベントフィールドに基づいて AWS DevOps エージェントイベントをフィルタリングします。

次の例は、一般的なユースケースのイベントパターンを示しています。

すべての AWS DevOps エージェントイベントを照合する

次のイベントパターンは、 AWS DevOps エージェントからのすべてのイベントに一致します。

```
{
  "source": ["aws.aidevops"]
}
```

調査イベントのみの一致

次のイベントパターンでは、プレフィックス一致を使用して、調査ライフサイクルイベントのみを選択します。

```
{
  "source": ["aws.aidevops"],
  "detail-type": [{"prefix": "Investigation"}]
}
```

完了イベントと失敗イベントのみを一致させる

次のイベントパターンは、完了したまたは失敗した調査と緩和策のイベントと一致します。

```
{
  "source": ["aws.aidevops"],
  "detail-type": [
    "Investigation Completed",
    "Investigation Failed",
    "Mitigation Completed",
    "Mitigation Failed"
  ]
}
```

特定のエージェントスペースのイベントを照合する

次のイベントパターンは、特定のエージェントスペースからのイベントに一致します。

```
{
  "source": ["aws.aidevops"],
  "detail": {
    "metadata": {
      "agent_space_id": ["your-agent-space-id"]
    }
  }
}
```

イベントパターンの詳細については、Amazon EventBridge ユーザーガイドの「Amazon EventBridge イベントパターン」を参照してください。

Amazon EventBridge アクセス許可

AWS DevOps Agent ではEventBridge にイベントを配信するための追加のアクセス許可は必要ありません。イベントはデフォルトのイベントバスに自動的に送信されます。

EventBridge ルール用に設定したターゲットによっては、特定のアクセス許可を追加する必要がある場合があります。ターゲットに必要なアクセス許可の詳細については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge のリソースベースのポリシーEventBridge」を参照してください。

追加の EventBridge リソース

EventBridge の概念と設定の詳細については、Amazon EventBridge ユーザーガイドの以下のトピックを参照してください。

- EventBridge イベントバス
- EventBridge イベント
- EventBridge イベントパターン
- EventBridge ルール
- EventBridge ターゲット

AWS DevOps エージェントイベントの詳細リファレンス

AWS サービスからのイベントには、、、`source`、、`detail-type`、`account`、`region`などの一般的なメタデータフィールドがありますtime。これらのイベントには、サービスに固有のデータを含

むdetailフィールドも含まれます。 AWS DevOps エージェントイベントの場合、 sourceは常にaws.aidevopsであり、 は特定のイベントdetail-typeを識別します。

調査イベント

次のdetail-type値は調査イベントを識別します。

- Investigation Created
- Investigation Priority Updated
- Investigation In Progress
- Investigation Completed
- Investigation Failed
- Investigation Timed Out
- Investigation Cancelled
- Investigation Pending Triage
- Investigation Linked

source および detail-typeフィールドには、 AWS DevOps エージェントイベントの特定の値が含まれているため、以下が含まれます。すべてのイベントに含まれる他のメタデータフィールドの定義については、「Amazon EventBridge イベントリファレンス」の「イベントの構造」を参照してください。

以下は、調査イベントの JSON 構造です。

```
{
  . . .,
  "detail-type" : "string",
  "source" : "aws.aidevops",
  . . .,
  "detail" : {
    "version" : "string",
    "metadata" : {
      "agent_space_id" : "string",
      "task_id" : "string",
      "execution_id" : "string"
    },
    "data" : {
      "task_type" : "string",
```

```
      "priority" : "string",
      "status" : "string",
      "created_at" : "string",
      "updated_at" : "string",
      "summary_record_id" : "string"
    }
  }
}
```

detail-type イベントのタイプを識別します。調査イベントの場合、これは前述のイベント名の 1 つです。

source イベントを生成したサービスを識別します。 AWS DevOps エージェントイベントの場合、この値は ですaws.aidevops。

detail イベント固有のデータを含む JSON オブジェクト。detail オブジェクトには、次のフィールドが含まれます。

- version (文字列) – イベント詳細のスキーマバージョン。現在 1.0.0。
- metadata.agent_space_id (文字列) – イベントが発生したエージェントスペースの一意の識別子。
- metadata.task_id (文字列) – タスクの一意の識別子。
- metadata.execution_id (文字列) – 実行実行の一意の識別子。実行が調査に割り当てられたときに表示されます。
- data.task_type (文字列) – タスクのタイプ。値: INVESTIGATION。
- data.priority (文字列) – 優先度。値: CRITICAL、HIGH、MEDIUM、LOW、MINIMAL。
- data.status (文字列) – 現在のステータス。値: PENDING_START、IN_PROGRESS、COMPLETED、、FAILED、TIMED_OUT、CANCELLEDPENDING_TRIAGE、LINKED
- data.created_at (文字列) – タスクが作成されたときの ISO 8601 タイムスタンプ。
- data.updated_at (文字列) – タスクが最後に更新されたときの ISO 8601 タイムスタンプ。
- data.summary_record_id (文字列) – 調査結果を含むサマリーレコードの識別子。完了した調査の概要が生成された場合に含まれます。この識別子を使用してレコードタイプが のジャーナルレコードを検索することで、 AWS DevOps Agent API を通じて概要コンテンツを取得できますinvestigation_summary_md。

例: 調査完了イベント

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789015",
  "detail-type": "Investigation Completed",
  "source": "aws.aidevops",
  "account": "123456789012",
  "time": "2026-03-12T18:10:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:aidevops:us-
east-1:123456789012:agentspace/8f6187a7-0388-4926-8217-3a0fe32f757c"
  ],
  "detail": {
    "version": "1.0.0",
    "metadata": {
      "agent_space_id": "8f6187a7-0388-4926-8217-3a0fe32f757c",
      "task_id": "a1b2c3d4-5678-90ab-cdef-example11111",
      "execution_id": "b2c3d4e5-6789-01ab-cdef-example22222"
    },
    "data": {
      "task_type": "INVESTIGATION",
      "priority": "CRITICAL",
      "status": "COMPLETED",
      "created_at": "2026-03-12T18:00:00Z",
      "updated_at": "2026-03-12T18:10:00Z",
      "summary_record_id": "d4e5f6g7-6789-01ab-cdef-example44444"
    }
  }
}
```

例: 調査失敗イベント

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789016",
  "detail-type": "Investigation Failed",
  "source": "aws.aidevops",
  "account": "123456789012",
  "time": "2026-03-12T18:10:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:aidevops:us-
east-1:123456789012:agentspace/8f6187a7-0388-4926-8217-3a0fe32f757c"
```

```
  ],
  "detail": {
    "version": "1.0.0",
    "metadata": {
      "agent_space_id": "8f6187a7-0388-4926-8217-3a0fe32f757c",
      "task_id": "a1b2c3d4-5678-90ab-cdef-example11111",
      "execution_id": "b2c3d4e5-6789-01ab-cdef-example22222"
    },
    "data": {
      "task_type": "INVESTIGATION",
      "priority": "CRITICAL",
      "status": "FAILED",
      "created_at": "2026-03-12T18:00:00Z",
      "updated_at": "2026-03-12T18:10:00Z"
    }
  }
}
```

緩和イベント

次のdetail-type値は緩和イベントを識別します。

- Mitigation In Progress
- Mitigation Completed
- Mitigation Failed
- Mitigation Timed Out
- Mitigation Cancelled

source および detail-typeフィールドには、 AWS DevOps エージェントイベントの特定の値が含まれているため、以下が含まれます。すべてのイベントに含まれる他のメタデータフィールドの定義については、「Amazon EventBridge イベントリファレンス」の「イベントの構造」を参照してください。

以下は、緩和イベントの JSON 構造です。

```
{
  . . .,
  "detail-type" : "string",
  "source" : "aws.aidevops",
  . . .,
```

```
  "detail" : {
    "version" : "string",
    "metadata" : {
      "agent_space_id" : "string",
      "task_id" : "string",
      "execution_id" : "string"
    },
    "data" : {
      "task_type" : "string",
      "priority" : "string",
      "status" : "string",
      "created_at" : "string",
      "updated_at" : "string",
      "summary_record_id" : "string"
    }
  }
}
```

detail-type イベントのタイプを識別します。緩和イベントの場合、これは前述のイベント名の 1 つです。

source イベントを生成したサービスを識別します。 AWS DevOps エージェントイベントの場合、この値は ですaws.aidevops。

detail イベント固有のデータを含む JSON オブジェクト。detail オブジェクトには、次のフィールドが含まれます。

- version (文字列) – イベント詳細のスキーマバージョン。現在 1.0.0。
- metadata.agent_space_id (文字列) – イベントが発生したエージェントスペースの一意の識別子。
- metadata.task_id (文字列) – タスクの一意の識別子。
- metadata.execution_id (文字列) – 実行実行の一意の識別子。実行が緩和に割り当てられたときに表示されます。
- data.task_type (文字列) – タスクのタイプ。値: INVESTIGATION。
- data.priority (文字列) – 優先度。値: CRITICAL、HIGH、MEDIUM、LOW、MINIMAL。
- data.status (文字列) – 現在のステータス。値: IN_PROGRESS、COMPLETED、FAILED、TIMED_OUT、CANCELLED。
- data.created_at (文字列) – タスクが作成されたときの ISO 8601 タイムスタンプ。

- data.updated_at (文字列) – タスクが最後に更新されたときの ISO 8601 タイムスタンプ。
- data.summary_record_id (文字列) – 緩和結果を含むサマリーレコードの識別子。完了した緩和策の概要が生成された場合に含まれます。この識別子を使用してレコードタイプが のジャーナルレコードを検索することで、 AWS DevOps Agent API を通じて概要コンテンツを取得できますmitigation_summary_md。

例: 緩和完了イベント

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-12345678901c",
  "detail-type": "Mitigation Completed",
  "source": "aws.aidevops",
  "account": "123456789012",
  "time": "2026-03-12T18:20:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:aidevops:us-
east-1:123456789012:agentspace/8f6187a7-0388-4926-8217-3a0fe32f757c"
  ],
  "detail": {
    "version": "1.0.0",
    "metadata": {
      "agent_space_id": "8f6187a7-0388-4926-8217-3a0fe32f757c",
      "task_id": "a1b2c3d4-5678-90ab-cdef-example11111",
      "execution_id": "c3d4e5f6-7890-12ab-cdef-example33333"
    },
    "data": {
      "task_type": "INVESTIGATION",
      "priority": "CRITICAL",
      "status": "COMPLETED",
      "created_at": "2026-03-12T18:00:00Z",
      "updated_at": "2026-03-12T18:20:00Z",
      "summary_record_id": "e5f6g7h8-7890-12ab-cdef-example55555"
    }
  }
}
```

例: 緩和失敗イベント

```
{
```

```
  "version": "0",
  "id": "12345678-1234-1234-1234-12345678901d",
  "detail-type": "Mitigation Failed",
  "source": "aws.aidevops",
  "account": "123456789012",
  "time": "2026-03-12T18:20:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:aidevops:us-
east-1:123456789012:agentspace/8f6187a7-0388-4926-8217-3a0fe32f757c"
  ],
  "detail": {
    "version": "1.0.0",
    "metadata": {
      "agent_space_id": "8f6187a7-0388-4926-8217-3a0fe32f757c",
      "task_id": "a1b2c3d4-5678-90ab-cdef-example11111",
      "execution_id": "c3d4e5f6-7890-12ab-cdef-example33333"
    },
    "data": {
      "task_type": "INVESTIGATION",
      "priority": "CRITICAL",
      "status": "FAILED",
      "created_at": "2026-03-12T18:00:00Z",
      "updated_at": "2026-03-12T18:20:00Z"
    }
  }
}
```

販売されたログとメトリクス

販売された Amazon CloudWatch メトリクスとログを使用して、エージェントスペースとサービスオペレーションをモニタリングできます。このトピックでは、 AWS DevOps Agent がアカウントに自動的に発行する CloudWatch メトリクスと、任意の送信先に配信するように設定できる販売ログについて説明します。

提供された CloudWatch メトリクス

AWS DevOps エージェントは、アカウントの Amazon CloudWatch にメトリクスを自動的に発行します。これらのメトリクスは、設定なしで使用できます。これらを使用して、使用状況のモニタリング、運用アクティビティの追跡、アラームの作成を行うことができます。

サービスにリンクされたロール

このサービスのアカウントで Amazon CloudWatch メトリクスを公開するために、 AWS DevOps エージェントはサービスにリンクされたロール AWSServiceRoleForAIDevOps サービスにリンクされたロールを自動的に作成します。API を呼び出す IAM ロールに適切なアクセス許可がない場合、リソースの作成は InvalidParameterException で失敗します。

> **Important**
>
> 2026 年 3 月 13 日より前に AgentSpace を作成したお客様は、AWSServiceRoleForAIDevOps サービスリンクロールを手動で作成して、 AWS DevOps エージェントの CloudWatch メトリクスを自分のアカウントで公開する必要があります。

サービスにリンクされたロールを手動で作成する (既存のお客様向け)

次のいずれかを行います。

- IAM コンソールで、DevOps エージェントサービスの下に AWSServiceRoleForAIDevOps ロールを作成します。 AWS DevOps
- AWS CLI から、次のコマンドを実行します。

```
aws iam create-service-linked-role --aws-service-name aidevops.amazonaws.com
```

名前空間

すべてのメトリクスは `AWS/AIDevOps`名前空間の下に発行されます。

ディメンション

すべてのメトリクスには、次のディメンションが含まれます。

ディメンション	説明
`AgentSpaceUUID`	エージェントスペースの一意の識別子。アカウント内のすべてのエージェントスペースのメトリクスを集計するには、CloudWatch の数式を

ディメンション	説明
	使用するか、ディメンションフィルターを省略します。

メトリクスのリファレンス

メトリクス名	説明	Unit	発行頻度	有用な統計
ConsumedChatRequests	エージェントスペースが消費したチャットリクエストの数。アカウントの合計数を取得するには、すべてのAgentSpaceUUID ディメンションで SUM 統計を使用します。	カウント	5 分ごと	合計、平均
ConsumedInvestigationTime	エージェントスペースでの調査の実行にかかった時間。	秒	5 分ごと	合計、平均、最大
ConsumedEvaluationTime	エージェントスペースでの評価の実行にかかった時間。	秒	5 分ごと	合計、平均、最大
TopologyCompletionCount	トポロジ処理の完了数。 AWS DevOps エージェントは、オンボーディン	カウント	イベント駆動型 (各完了時に発行)	Sum、SampleCount

メトリクス名	説明	Unit	発行頻度	有用な統計
	グ中の最初の作成、手動更新、またはスケジュールされた毎日の更新のいずれからでも、トポロジの処理が完了すると、このメトリクスを出力します。			

CloudWatch コンソールでのメトリクスの表示

1. CloudWatch コンソールを開きます。
2. ナビゲーションペインで、[Metrics] (メトリクス)、[All metrics] (すべてのメトリクス) の順に選択します。
3. AWS/AIDevOps 名前空間を選択します。
4. エージェントスペースのメトリクスを表示するには、AgentSpace で を選択します。

> **Note**
>
> これらのメトリクスに CloudWatch アラームを作成して、使用量がしきい値を超えたときに通知を受け取ることができます。たとえば、 でアラームを作成してConsumedChatRequests、チャットリクエストの消費量をモニタリングします。

前提条件

ログ配信を設定する前に、以下があることを確認してください。

- AWS DevOps エージェントコンソールにアクセスできるアクティブな AWS アカウント
- CloudWatch Logs 配信 APIs のアクセス許可を持つ IAM プリンシパル
- (オプション) Amazon S3 バケットまたは Amazon Data Firehose 配信ストリームをログ送信先として使用する予定の場合

提供されるログ

AWS DevOps Agent は、エージェントスペースとサービス登録が処理するイベントを可視化する販売ログをサポートします。提供されたログは、Amazon CloudWatch Logs インフラストラクチャを使用して、任意の宛先にログを配信します。

販売ログを使用するには、配信先を設定する必要があります。次の送信先がサポートされています。

- Amazon CloudWatch Logs – アカウントのロググループ
- Amazon S3 – アカウントの S3 バケット
- Amazon Data Firehose – アカウントの Firehose 配信ストリーム

サポートされているログのタイプ

1 つのログタイプがサポートされています: `APPLICATION_LOGS`。このログタイプは、サービスが発行するすべての運用イベントを対象としています。

ログイベントタイプ

次の表は、 AWS DevOps Agent がログに記録するイベントをまとめたものです。

[Event] (イベント)	説明	ログレベル
受信したエージェントのインバウンドイベント	エージェントは統合ソースによってトリガーされ、インバウンドイベント (PagerDuty インシデントイベントなど) を受け取ります。	情報
エージェントインバウンドイベントがドロップされました	インバウンドイベントは、エージェントが処理する前にドロップされました。ログには理由 (不正な形式のデータなど) が含まれます。	TBD
エージェントアウトバウンド通信の失敗	サードパーティー統合へのアウトバウンド通信に失敗しました。ログには、タスク ID と	TBD

[Event] (イベント)	説明	ログレベル
	送信先識別子 (認証エラーなど) が含まれます。	
キューに入れられたトポロジの作成	トポロジ作成ジョブが処理のためにキューに入れられました。	情報
トポロジの作成が開始されました	トポロジ作成ジョブの処理が開始されました。	情報
トポロジの作成が完了しました	トポロジ作成ジョブの処理が完了しました。このイベントは、最初の作成、更新、および毎日の更新に適用されます。	情報
リソース検出に失敗しました	トポロジの作成中にリソース検出でエラーが発生しました。	エラー
サービス登録に失敗しました	サービス登録で回復不可能な障害が発生する	エラー
Webhook の検証が失敗する	Devops エージェントによって受信されたウェブフックが予想されるスキーマと一致しない場合	エラー
関連付けの検証ステータスの更新	エージェントスペースの関連付け (通常、プライマリ/セカンダリアカウント) の場合、検証ステータスは有効から無効に変わり、その逆も同様です (たとえば、サービスでは想定できない不正な形式のロールが原因など) 。	エラー/INFO

アクセス許可

AWS DevOps エージェントは、CloudWatch 販売ログ (V2 アクセス許可) を使用してログを配信します。ログ配信を設定するには、配信を設定する IAM ロールに次のアクセス許可が必要です。

- `aidevops:AllowVendedLogDeliveryForResource` – エージェントスペースリソースのログ配信を許可するために必要です。
- CloudWatch Logs 配信 APIs のアクセス許可 (`logs:PutDeliverySource`、`logs:PutDeliveryDestinationlogs:CreateDelivery`、、および関連するオペレーション) 。
- 選択した配信先に固有のアクセス許可。

各送信先タイプに必要な完全な IAM ポリシーについては、Amazon CloudWatch Logs ユーザーガイドの以下のトピックを参照してください。

- CloudWatch Logs に送信されたログ
- Amazon S3 に送信されたログ
- Firehose に送信されたログ

ログ配信の設定 (コンソール)

AWS DevOps Agent は、ログ配信を設定するために AWS マネジメントコンソールに 2 つの場所を提供します。

- サービス登録設定ページ – サービスレベルのイベントのログ配信を設定します。これらのログは、サービス ARN (`arn:aws:aidevops:<region>:<account-id>:service/<account-id>`) をリソースとして使用します。
- エージェントスペースページ – 個々のエージェントスペースに固有のイベントのログ配信を設定します。これらのログは、エージェントスペース ARN (`arn:aws:aidevops:<region>:<account-id>:agentspace/<agent-space-id>`) をリソースとして使用します。

サービス登録のログ配信を設定するには

1. AWS マネジメントコンソールで AWS DevOps エージェントコンソールを開きます。
2. ナビゲーションペインで [設定] を選択します。

3. Capability Providers > Logs タブで、Configure を選択します。
4. 送信先タイプで、次のいずれかを選択します。
5. CloudWatch Logs – ロググループを選択または作成します。
6. Amazon S3 – S3 バケット ARN を入力します。
7. Amazon Data Firehose – Firehose 配信ストリームを選択または作成します。
8. 追加設定 – オプションで、次のオプションを指定できます。
   a. [フィールド選択] で、送信先に配信するログのフィールド名を選択します。アクセスログフィールドと、リアルタイムアクセスログフィールドのサブセットを選択できます。
   b. (Amazon S3 のみ) [パーティショニング] で、ログファイルデータをパーティション分割するパスを指定します。
   c. (Amazon S3 のみ) [Hive 互換ファイル形式] でチェックボックスをオンにして、Hive 互換 S3 パスを使用できます。これにより、Hive 互換ツールへの新しいデータのロードを簡素化できます。
   d. [出力形式] で、希望する形式を指定します。
   e. [フィールド区切り文字] で、ログフィールドを区切る方法を指定します。
9. [保存] を選択します。
10配信ステータスがアクティブになっていることを確認します。

エージェントスペースのログ配信を設定するには

1. AWS マネジメントコンソールで AWS DevOps エージェントコンソールを開きます。
2. 設定するエージェントスペースを選択します。
3. 設定 タブで、設定 を選択します。
4. 送信先タイプで、次のいずれかを選択します。
5. CloudWatch Logs – ロググループを選択または作成します。
6. Amazon S3 – S3 バケット ARN を入力します。
7. Amazon Data Firehose – Firehose 配信ストリームを選択または作成します。
8. 追加設定 – *オプション * では、次のオプションを指定できます。
   a. [フィールド選択] で、送信先に配信するログのフィールド名を選択します。アクセスログフィールドと、リアルタイムアクセスログフィールドのサブセットを選択できます。
   b. (Amazon S3 のみ) [パーティショニング] で、ログファイルデータをパーティション分割するパスを指定します。

c. (Amazon S3 のみ) [Hive 互換ファイル形式] でチェックボックスをオンにして、Hive 互換 S3 パスを使用できます。これにより、Hive 互換ツールへの新しいデータのロードを簡素化できます。

d. [出力形式] で、希望する形式を指定します。

e. [フィールド区切り文字] で、ログフィールドを区切る方法を指定します。

9. [保存] を選択します。

10配信ステータスがアクティブになっていることを確認します。

ログ配信の設定 (CloudWatch API)

CloudWatch Logs API を使用して、プログラムでログ配信を設定することもできます。動作しているログ配信は、次の 3 つの要素で構成されます。

- DeliverySource – ログを生成する AWS DevOps エージェントスペースリソースを表します。
- DeliveryDestination – ログが書き込まれる送信先を表します。
- 配信 – 配信ソースを配信先に接続します。

ステップ 1: 配信ソースを作成する

PutDeliverySource オペレーションを使用して配信ソースを作成します。 AWS DevOps エージェントスペースリソースの ARN を渡し、 をログタイプAPPLICATION_LOGSとして指定します。

次の例では、エージェントスペースの配信ソースを作成します。

```
{
    "name": "my-agent-space-delivery-source",
    "resourceArn": "arn:aws:aidevops:us-east-1:123456789012:agentspace/my-agent-space-
id",
    "logType": "APPLICATION_LOGS"
}
```

次の例では、 サービスの配信ソースを作成します。

```
{
    "name": "my-service-delivery-source",
    "resourceArn": "arn:aws:aidevops:us-east-1:123456789012:service",
    "logType": "APPLICATION_LOGS"
}
```

ステップ 2: 配信先を作成する

PutDeliveryDestination オペレーションを使用して、ログの保存先を設定します。Amazon CloudWatch Logs、Amazon S3、または Amazon Data Firehose を選択できます。

次の例では、CloudWatch Logs の送信先を作成します。

```
{
    "name": "my-cwl-destination",
    "deliveryDestinationConfiguration": {
        "destinationResourceArn": "arn:aws:logs:us-east-1:123456789012:log-group:/aws/
aidevops/my-agent-space"
    },
    "outputFormat": "json"
}
```

次の例では、Amazon S3 送信先を作成します。

```
{
    "name": "my-s3-destination",
    "deliveryDestinationConfiguration": {
        "destinationResourceArn": "arn:aws:s3:::my-aidevops-logs-bucket"
    },
    "outputFormat": "json"
}
```

次の例では、Amazon Data Firehose の送信先を作成します。

```
{
    "name": "my-firehose-destination",
    "deliveryDestinationConfiguration": {
        "destinationResourceArn": "arn:aws:firehose:us-
east-1:123456789012:deliverystream/my-aidevops-log-stream"
    },
    "outputFormat": "json"
}
```

> **Note**
>
> クロスアカウントでログを配信する場合は、送信先アカウントで PutDeliveryDestinationPolicy を使用して配信を承認する必要があります。

CloudFormation を使用する場合は、以下を使用できます。

- Delivery
- DeliveryDestination
- DeliverySource

ResourceArn は AgentSpaceArn であり、APPLICATION_LOGS はサポートされているログタイプとして LogType にする必要があります。

ステップ 3: 配信を作成する

CreateDelivery オペレーションを使用して、配信ソースを配信先にリンクします。

```
{
    "deliverySourceName": "my-agent-space-delivery-source",
    "deliveryDestinationArn": "arn:aws:logs:us-east-1:123456789012:delivery-
destination:my-cwl-destination"
}
```

AWS CloudFormation

次のリソースで AWS CloudFormation を使用してログ配信を設定することもできます。

- AWS::Logs::DeliverySource
- AWS::Logs::DeliveryDestination
- AWS::Logs::Delivery

ResourceArn を AWS DevOps エージェントスペースまたはサービス ARN に設定し、 LogType を に設定しますAPPLICATION_LOGS。

ログスキーマのリファレンス

AWS DevOps エージェントは、すべてのイベントタイプで共有ログスキーマを使用します。すべてのログイベントがすべてのフィールドを使用するわけではありません。

次の表に、ログスキーマのフィールドを示します。

フィールド	タイプ	説明
event_timestamp	Long	イベントが発生した時刻の Unix タイムスタンプ
resource_arn	String	イベントを生成したリソースの ARN
optional_account_id	String	AWS ログに関連付けられたアカウント ID。
optional_level	String	ログレベル: INFO、WARN、ERROR
optional_agent_space_id	String	エージェントスペースの識別子。
optional_association_id	String	ログの関連付け識別子。
optional_status	String	トポロジオペレーションのステータス。
optional_webhook_id	String	Webhook 識別子。
optional_mcp_endpoint_url	String	MCP サーバーエンドポイント URL
optional_service_type	String	サービスのタイプ: DYNATRACE 、DATADOG、、GITHUBSLACK、SERVICENOW 。
optional_service_endpoint_url	String	サードパーティー統合のエンドポイント URL。
optional_service_id	String	ソースの識別子。

フィールド	タイプ	説明
request_id	String	AWS CloudTrail またはサポートチケットと関連付けるためのリクエスト識別子。
optional_operation	String	実行されたオペレーションの名前。
optional_task_type	String	エージェントバックログタスクタイプ: `INVESTIGATION` または `EVALUATION`
optional_task_id	String	エージェントバックログタスク IDAgentバックログタスク識別子。
optional_reference	String	エージェントタスクからのリファレンス (Jira チケットなど) 。
optional_error_type	String	エラータイプ
optional_error_message	String	オペレーションが失敗した場合のエラーの説明。
optional_details	文字列 (JSON)	オペレーションパラメータと結果を含むサービス固有のイベントペイロード。

ログ配信の管理と無効化

ログ配信は、 AWS マネジメントコンソールの AWS DevOps エージェントコンソールから、または CloudWatch Logs API を使用して、いつでも変更または削除できます。

ログ配信の管理 (コンソール)

1. AWS マネジメントコンソールで AWS DevOps エージェントコンソールを開きます。

2. 設定ページ (サービスレベルのログの場合) または特定のエージェントスペースページ (エージェントスペースレベルのログの場合) に移動します。
3. 設定タブ (エージェントスペースレベルのログの場合) または機能プロバイダー > ログタブ (サービスレベルのログの場合) で、変更する配信を選択します。
4. 必要に応じて設定を更新し、保存を選択します。

> 注: 既存の配信の送信先タイプを変更することはできません。送信先タイプを変更するには、現在の配信を削除し、新しい配信を作成します。

ログ配信を無効にする (コンソール)

1. AWS マネジメントコンソールで AWS DevOps エージェントコンソールを開きます。
2. 設定ページ (サービスレベルのログの場合) または特定のエージェントスペースページ (エージェントスペースレベルのログの場合) に移動します。
3. 設定タブ (エージェントスペースレベルのログの場合) または機能プロバイダー > ログタブ (サービスレベルのログの場合) で、削除する配信を選択します。
4. 削除を選択して確認します。

ログ配信を無効にする (API)

API を使用してログ配信を削除するには、次の順序でリソースを削除します。

1. DeleteDelivery を使用して配信を削除します。
2. DeleteDeliverySource を使用して配信ソースを削除します。
3. (オプション) 配信先が不要になった場合は、DeleteDeliveryDestination を使用して配信先を削除します。

> **Important**
>
> ログを生成するエージェントスペースリソースを削除した後 (エージェントスペースを削除した後など) 、ログ配信リソースを削除する必要があります。これらのリソースを削除しない場合、孤立した配信設定が残る可能性があります。

料金

AWS DevOps エージェントは、提供されたログの有効化には課金しません。ただし、選択したログ配信先によっては、配信、取り込み、ストレージ、またはアクセスに対して料金が発生する場合があります。料金の詳細については、「Amazon CloudWatch 料金表」の「ログ」タブの「Vended Logs」を参照してください。 Amazon CloudWatch

送信先固有の料金については、以下を参照してください。

- Amazon CloudWatch Logs 料金
- Amazon S3 料金
- Amazon Data Firehose 料金

プライベートにホストされたツールへの接続

プライベート接続の概要

AWS DevOps エージェントは、カスタムモデルコンテキストプロトコル (MCP) ツールやその他の統合を使用して拡張できます。これにより、エージェントはプライベートパッケージレジストリ、セルフホスト型オブザーバビリティプラットフォーム、内部ドキュメント APIs「」を参照AWS DevOps Agent の機能の設定)。これらのサービスは、多くの場合、パブリックインターネットアクセスが制限されているか、パブリックインターネットアクセスがない Amazon Virtual Private Cloud (Amazon VPC) 内で実行されます。つまり、 AWS DevOps Agent はデフォルトでそれらにアクセスできません。

AWS DevOps Agent のプライベート接続を使用すると、パブリックインターネットに公開することなく、VPC で実行されているサービスにエージェントスペースを安全に接続できます。プライベート接続は、MCP サーバー、セルフホスト型の Grafana または Splunk インスタンス、GitHub Enterprise Server や GitLab Self-Managed などのソース管理システムなど、プライベートエンドポイントに到達する必要がある統合と連携します。

> **Note**
>
> プライベートにホストされたツールが VPC 内から AWS DevOps エージェントにアウトバウンドリクエストを行う場合、このトラフィックは VPC エンドポイントを使用して保護し、 AWS ネットワーク内に留まることもできます。たとえば、これはウェブフックイベントを介して DevOps エージェントをトリガーするツールで使用できます (「」を参照the

section called “Webhook による DevOps エージェントの呼び出し”)。詳細については、「the section called “VPC エンドポイント (AWS PrivateLink)”」を参照してください。

プライベート接続の仕組み

プライベート接続は、 AWS DevOps Agent と VPC 内のターゲットリソースとの間に安全なネットワークパスを作成します。 AWS DevOps Agent は、Amazon VPC Lattice を使用して、この安全なプライベート接続パスを確立します。VPC Lattice は、基盤となるネットワークインフラストラクチャを管理することなく、VPCs、アカウント、コンピューティングタイプ間のアプリケーション間の通信を接続、保護、モニタリングできるアプリケーションネットワークサービスです。

プライベート接続を作成すると、以下が発生します。

- ターゲットサービスへのネットワーク接続を持つ VPC、サブネット、および (オプションで) セキュリティグループを指定します。
- AWS DevOps Agent は、サービスマネージドリソースゲートウェイを作成し、指定したサブネットに Elastic Network Interface (ENIs) をプロビジョニングします。
- エージェントはリソースゲートウェイを使用して、プライベートネットワークパス経由でターゲットサービスの IP アドレスまたは DNS 名にトラフィックをルーティングします。

リソースゲートウェイは AWS DevOps エージェントによって完全に管理され、 アカウント (という名前) の読み取り専用リソースとして表示されますaidevops-{your-private-connection-name}。設定または保守する必要はありません。VPC で作成されるリソースはENIs のみです。これらの ENIs はプライベートトラフィックのエントリポイントとして機能し、 サービスによって完全に管理されます。インターネットからのインバウンド接続は受け付けず、独自のセキュリティグループを介してトラフィックを完全に制御できます。

セキュリティ

プライベート接続は、複数のセキュリティレイヤーで設計されています。

- パブリックインターネットへの露出なし — AWS DevOps Agent とターゲットサービス間のすべてのトラフィックは AWS ネットワーク上にとどまります。サービスにパブリック IP アドレスやインターネットゲートウェイは必要ありません。
- サービスマネージドリソースゲートウェイ – サービスマネージドリソースゲートウェイは、アカウント内で読み取り専用です。 AWS DevOps エージェントのみが使用でき、他のサービスやプリ

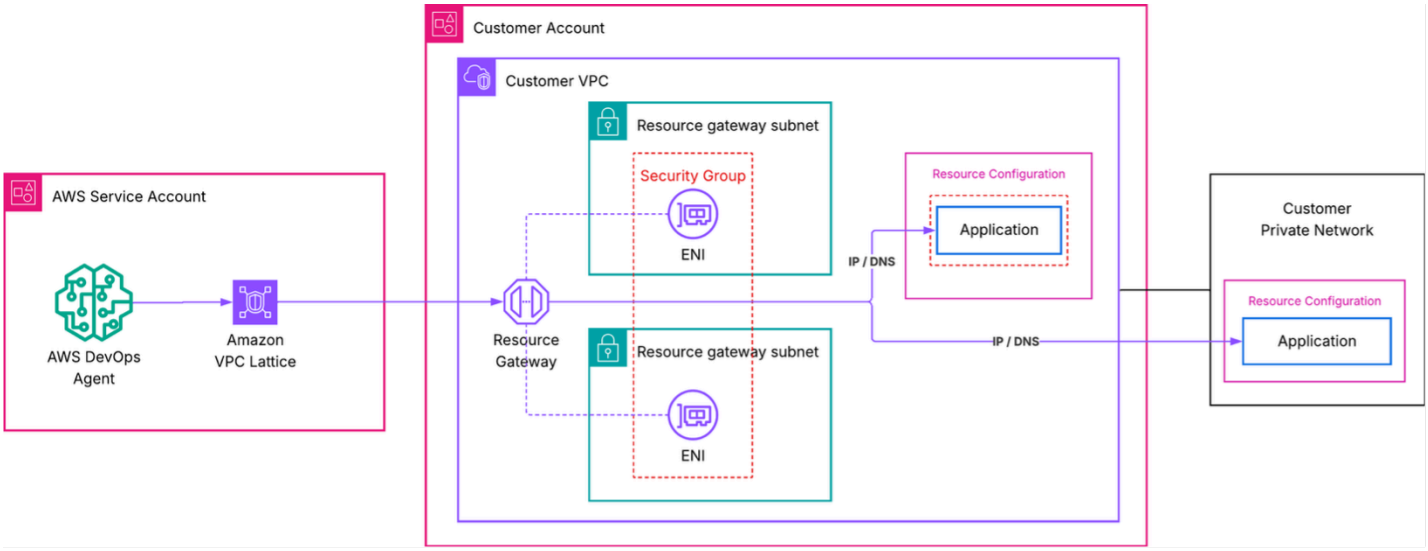
ンシパルがそのエージェントを介してトラフィックをルーティングすることはできません。これは、すべての VPC Lattice API コールを記録する AWS CloudTrail ログで確認できます。

- セキュリティグループ、ルール – 所有および管理しているセキュリティグループを通じてENIs へのインバウンドトラフィックとアウトバウンドトラフィックを制御します。セキュリティグループを指定しない場合、 AWS DevOps Agent は定義したポートを対象とするデフォルトのセキュリティグループを作成します。
- 最小特権のサービスにリンクされたロール – AWS DevOps エージェントは、サービスにリンクされたロールを使用して、必要な VPC Lattice と Amazon EC2 リソースのみを作成します。このロールは、 でタグ付けされたリソースに限定AWSAIDevOpsManagedされており、アカウントの他のリソースにアクセスすることはできません。

> **Note**
>
> 組織に VPC Lattice API アクションを制限するサービスコントロールポリシー (SCPs) がある場合、サービスマネージドリソースゲートウェイはサービスにリンクされたロールを介して作成されます。SCPsサービスにリンクされたロールに必要なアクションを許可していることを確認します。

アーキテクチャ

次の図は、プライベート接続のネットワークパスを示しています。



このアーキテクチャの詳細は以下のとおりです。

- AWS DevOps エージェントは、ターゲットサービスへのリクエストを開始します。
- Amazon VPC Lattice は、VPC のサービスマネージドリソースゲートウェイを介してリクエストをルーティングします。独自の VPC Lattice リソースを使用した高度なセットアップについては、「既存の VPC Lattice リソースを使用した高度なセットアップ」を参照してください。
- VPC の ENI はトラフィックを受信し、ターゲットサービスの IP アドレスまたは DNS 名に転送します。
- セキュリティグループは、ENIsを介して許可されるトラフィックを管理します。
- ターゲットサービスの観点から、リクエストは VPC 内の ENIs のプライベート IP アドレスから送信されます。

プライベート接続を作成する

プライベート接続は、 AWS マネジメントコンソールまたは CLI AWS を使用して作成できます。

> **Note**
>
> VPC Lattice では、次のアベイラビリティーゾーンはサポートされていません: `use1-az3`、`usw1-az2`、`apne1-az3`、、`apne2-az2`、`euc1-az2euw1-az4cac1-az3`、、`ilc1-az2`。

前提条件

プライベート接続を作成する前に、以下があることを確認します。

- アクティブなエージェントスペース – アカウント内に既存のエージェントスペースが必要です。アカウントをお持ちでない場合は、「AWS DevOps エージェントの開始方法」を参照してください。
- プライベートにアクセス可能なターゲットサービス – MCP サーバー、オブザーバビリティプラットフォーム、またはその他のサービスは、リソースゲートウェイがデプロイされている VPC の既知のプライベート IP アドレスまたは DNS 名で到達可能である必要があります。サービスは、リソースゲートウェイサブネットからルーティング可能である限り、同じ VPC、ピア接続された VPC、またはオンプレミスで実行できます。サービスは、接続の作成時に指定したポートで、最小 TLS バージョン 1.2 の HTTPS トラフィックを提供する必要があります。
- VPC 内のサブネット – ENIs が作成されるサブネットを 1 ~ 20 個特定します。高可用性を実現するには、複数のアベイラビリティーゾーンのサブネットを選択することをお勧めします。これらの

サブネットには、ターゲットサービスへのネットワーク接続が必要です。VPC Lattice では、アベイラビリティーゾーンごとに 1 つのサブネットを使用できます。

- (オプション) セキュリティグループ – 特定のルールでトラフィックを制御する場合は、ENI にアタッチするセキュリティグループ IDs を最大 5 つまで準備します。 ENIs セキュリティグループを省略すると、 AWS DevOps Agent はデフォルトのセキュリティグループを作成します。

プライベート接続はアカウントレベルのリソースです。プライベート接続を作成したら、同じホストに到達する必要がある複数の統合とエージェントスペースで再利用できます。

コンソールを使用してプライベート接続を作成する

1. AWS DevOps エージェントコンソールを開きます。
2. ナビゲーションペインで、機能プロバイダーを選択し、プライベート接続を選択します。
3. [新しい接続を作成する] を選択します。
4. Name には、 など、接続のわかりやすい名前を入力しますmy-mcp-tool-connection。
5. VPC の場合は、リソースゲートウェイ ENIs がデプロイされる VPC を選択します。
6. サブネットの場合は、1 つ以上のサブネット (最大 20) を選択します。少なくとも 2 つのアベイラビリティーゾーンでサブネットを選択することをお勧めします。
7. IP アドレスタイプで、ターゲットサービスの IP アドレスのタイプ (IPv4、IPv6、または) を選択しますDualStack。
8. (オプション) IPv4 アドレスの数で、IP アドレスタイプに IPv4 またはデュアルスタックを選択した場合は、リソースゲートウェイの ENI ごとに IPv4 アドレスの数を入力できます。デフォルトは、ENI あたり 16 個の IPv4 アドレスです。
9. (オプション) セキュリティグループで、既存のセキュリティグループ (最大 5) を選択して、ターゲットサービスに到達できるトラフィックを制限します。選択しない場合、デフォルトのセキュリティグループが作成されます。
10. (オプション) ポート範囲には、ターゲットアプリケーションがリッスンする TCP ポートを指定します (例: 443または 8080-8090)。最大 11 個のポート範囲を指定できます。
11. ホストアドレスには、ターゲットサービスの IP アドレスまたは DNS 名 (例: mcp.internal.example.comまたは) を入力します10.0.1.50。サービスは、選択した VPC から到達可能である必要があります。DNS 名を選択する場合は、選択した VPC から解決可能である必要があります。
12. (オプション) 証明書パブリックキーで、指定したホストアドレスがプライベート認証機関によって発行された TLS 証明書を使用している場合は、証明書の PEM エンコードされたパブリック

キーを入力します。これにより、 AWS DevOps Agent はターゲットサービスへの TLS 接続を信頼できます。

13[接続を作成] を選択します。

接続ステータスが進行中の作成に変わります。このプロセスには最大 10 分かかる場合があります。ステータスがアクティブに変わると、ネットワークパスの準備が整います。

ステータスが「作成」に変わった場合は、以下を確認します。

- 指定したサブネットには使用可能な IP アドレスがあります。
- アカウントが VPC Lattice サービスクォータに達していません。
- 制限付きの IAM ポリシーは、サービスにリンクされたロールによるリソースの作成を妨げません。

> **Note**
>
> これらのステップは、機能プロバイダーの登録`Create a new private connection`時に を選択して実行することもできます。詳細については、「機能プロバイダーとのプライベート接続を使用する」を参照してください。

CLI AWS を使用してプライベート接続を作成する

次のコマンドを実行して、プライベート接続を作成します。プレースホルダーの値を独自の値に置き換えます。

```
aws devops-agent create-private-connection \
    --name my-mcp-tool-connection \
    --mode '{
        "serviceManaged": {
            "hostAddress": "mcp.internal.example.com",
            "vpcId": "vpc-0123456789abcdef0",
            "subnetIds": [
                "subnet-0123456789abcdef0",
                "subnet-0123456789abcdef1"
            ],
            "securityGroupIds": [
                "sg-0123456789abcdef0"
```

```
            ],
            "portRanges": ["443"]
        }
    }'
```

レスポンスには、接続名と のステータスが含まれますCREATE_IN_PROGRESS。

```
{
    "name": "my-mcp-tool-connection",
    "status": "CREATE_IN_PROGRESS",
    "resourceGatewayId": "rgw-0123456789abcdef0",
    "hostAddress": "mcp.internal.example.com",
    "vpcId": "vpc-0123456789abcdef0"
}
```

接続ステータスを確認するには、 describe-private-connection コマンドを使用します。

```
aws devops-agent describe-private-connection \
    --name my-mcp-tool-connection
```

ステータスが になるとACTIVE、プライベート接続を使用する準備が整います。

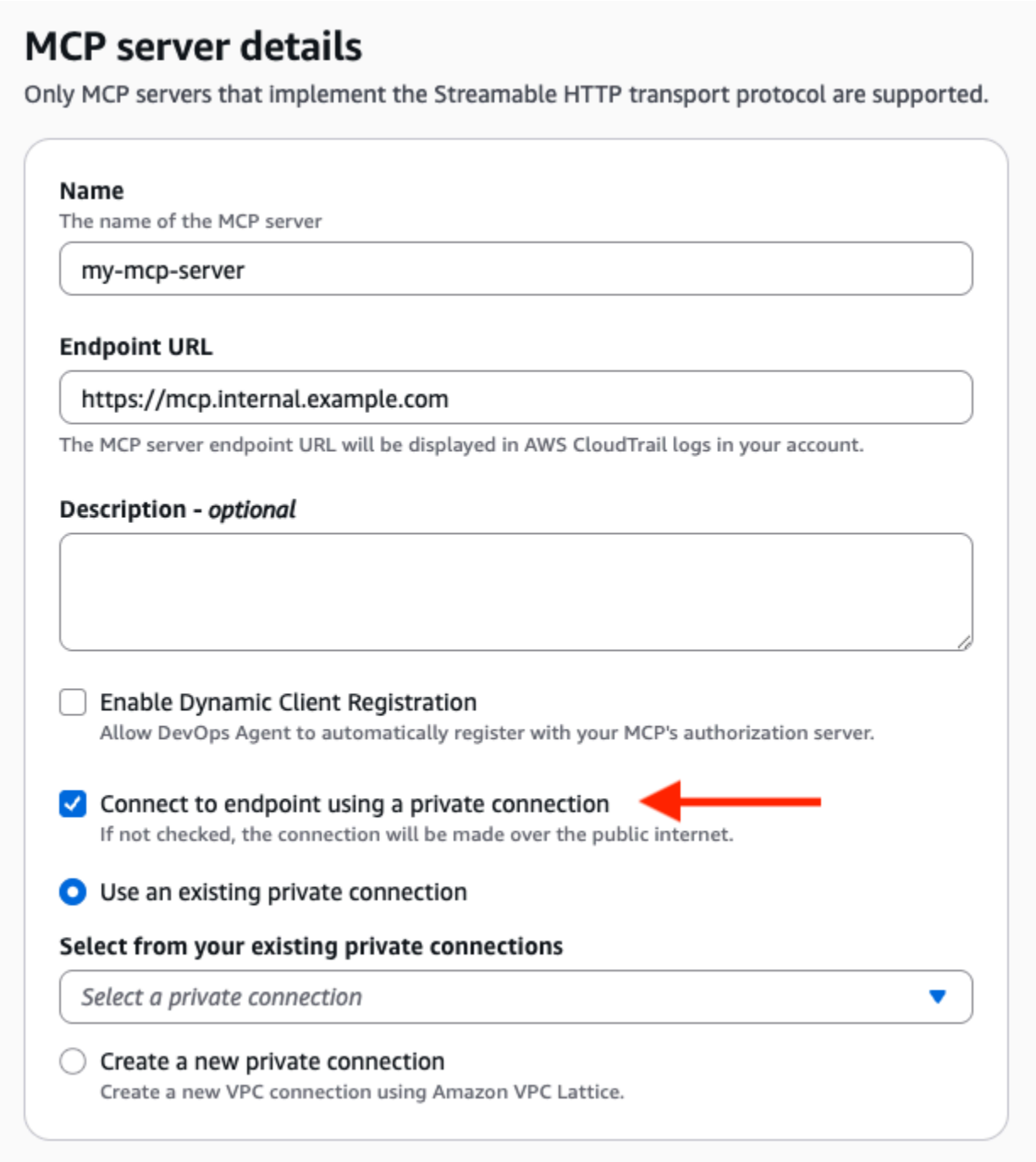
機能プロバイダーとのプライベート接続を使用する

プライベート接続を使用するには、機能プロバイダーの登録中にプライベート接続にリンクできます。プライベート接続で使用できるサポート対象の機能には、GitHub、、GitLabMCP Server、などがありますGrafana。このステップは、 AWS マネジメントコンソールまたは CLI AWS を使用して実行できます。

> **Note**
>
> 機能プロバイダーを登録すると、 AWS DevOps Agent はエンドポイントに到達可能で応答していることを検証します。登録を完了する前に、ターゲットサービスが実行されていて、接続を受け入れていることを確認します。

コンソールを使用して機能プロバイダーとのプライベート接続を使用する

AWS DevOps エージェントコンソールでは、「プライベート接続を使用してエンドポイントに接続する」オプションを選択して、登録中にプライベート接続を機能にリンクできます。



1. AWS DevOps エージェントコンソールを開き、エージェントスペースに移動します。
2. 「機能プロバイダー」セクションで、「登録」を選択します。
3. プライベート接続で使用する機能タイプの登録を選択します。
4. 登録の詳細ビューで、プライベート接続を使用して接続するエンドポイント URL を入力します (例: `https://mcp.internal.example.com`)。
5. プライベート接続を使用してエンドポイントに接続するを選択します。

6. 接続するエンドポイント URL に対応する既存のプライベート接続を選択するか、新しいプライベート接続を作成するを選択します。
7. 機能プロバイダーの登録プロセスを完了します。

AWS CLI を使用して機能プロバイダーとのプライベート接続を使用する

引`private-connection-name`数を含めることで、プライベート接続に機能を登録できます。以下は、`my-mcp-tool-connection`プライベート接続を使用して API キー認可で MCP サーバーを登録する例です。プレースホルダーの値を独自の値に置き換えます。

```
aws devops-agent register-service \
    --service mcpserver \
    --private-connection-name my-mcp-tool-connection \
    --service-details '{
        "mcpserver": {
            "name": "my-mcp-tool",
            "endpoint": "https://mcp.internal.example.com",
            "authorizationConfig": {
                "apiKey": {
                    "apiKeyName": "api-key",
                    "apiKeyValue": "secret-value",
                    "apiKeyHeader": "x-api-key"
                }
            }
        }
    }' \
    --region us-east-1
```

プライベート接続を検証する

プライベート接続がアクティブ状態になり、機能プロバイダーによって利用されたら、 AWS DevOps Agent がターゲットサービスに到達できることを確認します。

1. AWS DevOps エージェントコンソールを開き、エージェントスペースに移動します。
2. 新しいチャットセッションを開始します。
3. プライベート接続でバックアップされた統合を使用するコマンドを呼び出します。たとえば、MCP ツールが内部ナレッジベースへのアクセスを提供する場合は、そのナレッジベースを必要とする質問をエージェントに依頼します。
4. エージェントがプライベートサービスから結果を返すことを確認します。

接続が失敗した場合は、以下を確認してください。

- VPC Lattice 制限 - リソースゲートウェイやその他の VPC Lattice クォータ制限に達していないことを確認します。
- セキュリティグループルール – ENIs にアタッチされたセキュリティグループが、サービスがリッスンするポートでアウトバウンドトラフィックを許可していることを確認します。また、サービスのセキュリティグループがターゲットポートでインバウンドトラフィックを許可していることを確認します。VPC CIDR 範囲内の VPC Lattice データプレーン IPs からトラフィックが到着します。セキュリティグループ参照 (ENI セキュリティグループをソースとして許可) を使用するか、VPC CIDR からのインバウンドを許可できます。
- サブネット接続 — 選択したサブネットがサービスにトラフィックをルーティングできることを確認します。サービスが別のサブネットで実行されている場合は、ルートテーブルがそれらの間のトラフィックを許可していることを確認します。
- サービスの可用性 – サービスが実行されており、予想されるポートで接続を受け入れていることを確認します。
- サポートされていないアベイラビリティーゾーン - サブネットがサポートされているアベイラビリティーゾーンにあることを確認します。を実行して`aws ec2 describe-subnets --subnet-ids <your-subnet-ids> --query 'Subnets[*].[SubnetId,AvailabilityZoneId]'`、上記のサポートされていないアベイラビリティーゾーンと照合します。

プライベート接続を削除する

未使用のプライベート接続は、 AWS マネジメントコンソールまたは CLI AWS を使用して削除できます。

コンソールを使用してプライベート接続を削除する

1. AWS DevOps エージェントコンソールを開きます。
2. ナビゲーションペインで、機能プロバイダーを選択し、プライベート接続を選択します。
3. 削除するプライベート接続のアクションメニューを選択し、削除を選択します。

プライベート接続のステータスは「接続の削除」と表示され、 AWS DevOps Agent は VPC からマネージドリソースゲートウェイと ENI を削除します。 ENIs 削除が完了すると、接続はプライベート接続のリストに表示されなくなります。

CLI AWS を使用してプライベート接続を削除する

```
aws devops-agent delete-private-connection \
    --name my-mcp-tool-connection
```

レスポンスは ステータスを返しますDELETE_IN_PROGRESS。 AWS DevOps Agent は VPC からマネージドリソースゲートウェイと ENIs を削除します。削除が完了すると、接続はプライベート接続のリストに表示されなくなります。

既存の VPC Lattice リソースを使用した高度なセットアップ

組織がすでに Amazon VPC Lattice を使用していて、独自のリソース設定を管理している場合は、セルフマネージドモードでプライベート接続を作成できます。 AWS DevOps エージェントがリソースゲートウェイを作成する代わりに、ターゲットサービスを指す既存のリソース設定の Amazon リソースネーム (ARN) を指定します。

このアプローチは、次の場合に便利です。

- リソースゲートウェイとリソース設定ライフサイクルを完全に制御したい。
- 複数の AWS アカウントまたはサービス間でリソース設定を共有する必要があります。
- 詳細なトラフィックモニタリングには、VPC Lattice アクセスログが必要です。
- hub-and-spokeのネットワークアーキテクチャを実行します。

AWS CLI を使用してセルフマネージドプライベート接続を作成するには:

```
aws devops-agent create-private-connection \
    --name my-advanced-connection \
    --mode '{
        "selfManaged": {
            "resourceConfigurationId": "arn:aws:vpc-lattice:us-
east-1:123456789012:resourceconfiguration/rcfg-0123456789abcdef0"
        }
    }'
```

VPC Lattice リソースゲートウェイとリソース設定の設定の詳細については、「Amazon VPC Lattice ユーザーガイド」を参照してください。

関連トピック

- the section called “VPC エンドポイント (AWS PrivateLink)”
- the section called “MCP サーバーの接続”
- AWS DevOps Agent の機能の設定
- AWS DevOps エージェントセキュリティ
- the section called “DevOps エージェント IAM アクセス許可”

AWS DevOps エージェントセキュリティ

このドキュメントでは、 AWS DevOps Agent のセキュリティ上の考慮事項、データ保護、アクセスコントロール、コンプライアンス機能について説明します。この情報を使用して、 AWS DevOps Agent がセキュリティおよびコンプライアンス要件を満たすように設計されている方法を理解します。

マルチレイヤーセキュリティ

AWS DevOps Agent は、複数のレイヤーにセキュリティを実装します。より広範なアクセス許可がエージェントの IAM ロールに付与されている場合でも、エージェントは独自の内部アクセスコントロールを適用してアクションの範囲を制限します。たとえば、顧客が完全な Amazon S3 アクセス IAM ポリシーをエージェントの IAM ロールに追加すると、 AWS DevOps Agent は、トラブルシューティングの目的で`AWSLogs`プレフィックスの後のログのみが読み取られるようにします。

AWS DevOps Agent の IAM アクセス許可を設定し、複数のレイヤーにセキュリティを実装するときは、最小特権の原則に従うことをお勧めします。多層防御により、単一の設定ミスによって環境のセキュリティが損なわれることはありません。

エージェントスペース

エージェントスペースは、 AWS DevOps エージェントの主要なセキュリティ境界として機能します。各エージェントスペース:

- 独自の設定とアクセス許可で独立して動作します
- エージェントがアクセスできる AWS アカウントとリソースを定義します。
- サードパーティープラットフォームへの接続を確立します

エージェントスペースは、セキュリティを確保し、さまざまな環境やチーム間の意図しないアクセスを防ぐために、厳密な分離を維持します。

リージョン処理とデータフロー

AWS DevOps エージェントは、リージョンの処理機能を使用してグローバルに運用されています。エージェントは、設定されたエージェントスペース内のアクセス権が付与されたすべての AWS アカ

ウントの AWS リージョンから運用データを取得します。このマルチリージョンのクロスアカウントデータ収集は、推論処理の地理的境界を尊重しながら、包括的なインシデント分析を実現します。

Amazon Bedrock の使用とクロスリージョン推論

AWS DevOps エージェントは、推論リクエストを処理するために、地理的な最適なリージョンを自動的に選択します。これにより、利用可能なコンピューティングリソース、モデルの可用性を最大化し、最高のカスタマーエクスペリエンスを実現します。データは、エージェントスペースが作成されたリージョンにのみ保存されますが、次のリストで説明するように、入力プロンプトと出力結果がそのリージョン外で処理される場合があります。すべてのデータは Amazon の安全なネットワーク経由で暗号化されて送信されます。

AWS DevOps Agent は、次のように、推論リクエストをリクエスト元の地理的領域内の利用可能なコンピューティングリソースに安全にルーティングします。

- 欧州連合を起点とする推論リクエストは、欧州連合内で処理されます。
- 米国を起点とする推論リクエストは、米国内で処理されます。
- オーストラリアを起点とする推論リクエストは、オーストラリア内で処理されます。
- 日本国内からの推論リクエストは、日本国内で処理されます。
- 推論リクエストがリストされていないエリアで発生した場合、デフォルトで米国内で処理されます。
- DevOps エージェントと Bedrock は、顧客コンテンツを特定のリージョンに制限するサービスコントロールポリシー (SCPsまたは Control Tower の顧客ポリシーの影響を受けません。
- Bedrock は、パフォーマンスと可用性を最適化するために、地域内の発信元リージョン以外のリージョンを使用してステートレス推論を実行する場合があります。

ID とアクセス管理

認証方法

AWS DevOps エージェントには、 AWS DevOps エージェントスペースウェブアプリにログインするための 2 つの認証方法が用意されています。

- AWS アイデンティティセンターの統合 – プライマリ認証方法では、HTTP 専用 Cookie を使用したセッションベースの認証で OAuth 2.0 を使用します。 AWS アイデンティティセンターは、Okta、Ping Identity、Microsoft Entra ID などのプロバイダーを含む標準の OIDC および SAML

プロトコルを通じて外部 ID プロバイダーとフェデレーションできます。このメソッドは、ID プロバイダーを介した多要素認証をサポートします。 AWS Identity Center はデフォルトで最大 12 時間のセッション期間に設定され、希望する期間に設定できます。

- IAM 認証リンク – 別の方法では、既存の AWS マネジメントコンソールセッションから派生した JWT ベースのトークンを使用して、 AWS マネジメントコンソールからウェブアプリケーションに直接アクセスできます。このオプションは、完全な Identity Center 統合を実装する前に AWS DevOps エージェントを評価する場合や、Identity Center ベースの認証を通じて AWS DevOps Agent ウェブアプリケーションにアクセスできなくなった場合に管理アクセスを取得する場合に便利です。セッションは 10 分に制限されています。

IAM ロール

AWS DevOps エージェントは IAM ロールを使用してアクセス許可を定義します。

- プライマリアカウントロール – エージェントスペースを作成する AWS アカウントのリソースへのアクセスと、セカンダリアカウントロールへのアクセスをエージェントに許可します。
- セカンダリアカウントロール – エージェントスペースに接続された追加の AWS アカウントのリソースへのアクセス権をエージェントに付与します。
- ウェブアプリケーションロール – ウェブアプリケーションの AWS DevOps エージェント調査データと検出結果へのアクセス権をユーザーに付与します。

これらのロールは、最小特権の原則に従って設定し、調査に必要な読み取り専用アクセス許可のみを付与する必要があります。

データ保護

データ暗号化

AWS DevOps エージェントは、すべての顧客データを暗号化します。

- 保管時の暗号化 – すべてのデータは AWSマネージドキーで暗号化されます。
- 転送中の暗号化 – 取得されたすべてのログ、メトリクス、ナレッジ項目、チケットメタデータ、およびその他のデータは、エージェントのプライベートネットワーク内および外部ネットワークへの転送中に暗号化されます。

データストレージと保持

データはエージェントスペースが作成されたリージョンに保存されますが、上記の Amazon Bedrock の使用に関するセクションで説明されているように、推論処理はお客様の地域内で発生する可能性があります。

個人を特定できる情報 (PII)

AWS DevOps エージェントは、調査、レコメンデーション評価、またはチャットレスポンス中に収集されたデータを要約する際に PII 情報をフィルタリングしません。PII データは、オブザーバビリティログに保存する前に編集することをお勧めします。

エージェントジャーナルと監査ログ記録

エージェントジャーナル

Incident Investigation and Prevention 機能はいずれも、次のような詳細なジャーナルを維持します。

- すべての推論ステップと実行されたアクションをログに記録する
- エージェントの意思決定プロセスに完全な透明性を持たせる
- 一度記録するとエージェントは変更できず、プロンプトインジェクションなどの攻撃が重要なアクションを非表示にしないようにします。
- 調査ページからすべてのチャットメッセージを含める

AWS CloudTrail 統合

すべての AWS DevOps エージェント API コールは、ホスティング AWS アカウント内の AWS CloudTrail によって自動的にキャプチャされます。CloudTrail によって収集されたデータを使用して、以下の情報を判断できます。

- エージェントに対して行われたリクエスト
- リクエストが行われた IP アドレス
- リクエストを行ったユーザー
- リクエストが行われた時刻

プロンプトインジェクション保護

プロンプトインジェクション攻撃は、攻撃者が外部データに悪意のある指示を埋め込むときに発生します。 ウェブページやドキュメント、 生成 AI システムが後で処理します。 AWS DevOps Agent は、通常のオペレーションの一環として多くのデータソースをネイティブに消費します。 ログ、 リソースタグ、 およびその他の運用データ。 AWS DevOps Agent は、以下の保護手段を通じてプロンプトインジェクション攻撃から保護します。 ただし、接続されているすべてのデータソースと、それらのデータソースへのユーザーアクセスが信頼されていることを確認することが重要です。詳細については、「 責任共有モデル」セクションを参照してください。

プロンプトインジェクションの保護:

- 書き込み機能の制限 – エージェントが利用できるツールは、チケットの開封とサポートケースを除き、リソースを変更することはできません。これにより、悪意のある指示によってインフラストラクチャやアプリケーションが変更されるのを防ぐことができます。
- アカウント境界の適用 – AWS DevOps エージェントは、プライマリアカウントと接続されたセカンダリ AWS アカウントのエージェントに割り当てられたロールによって許可された境界内でのみ動作します。エージェントは、設定された範囲外のリソースにアクセスしたり変更したりすることはできません。
- AI 安全保護 – AWS DevOps エージェントは AI 安全レベル 3 (ASL-3) 保護を備えたモデルを使用します。これらの保護には、エージェントの動作に影響を与える前にプロンプトインジェクション攻撃を検出して防止する分類子が含まれます。
- イミュータブルな監査証跡 – エージェントジャーナルは、すべての推論ステップと実行されたアクションを記録します。一度記録すると、エージェントはジャーナルエントリを変更できないため、プロンプトインジェクション攻撃によって悪意のあるアクションが非表示になります。

AWS DevOps Agent はプロンプトインジェクション攻撃に対して複数の保護レイヤーを提供しますが、特定の設定はリスクを高める可能性があります。

- カスタム MCP サーバーツール – bring-your-own MCP 機能を使用すると、エージェントにカスタムツールを導入できるため、プロンプトインジェクションの機会が増える可能性があります。カスタムツールにはネイティブの AWS DevOps エージェントツールと同じセキュリティコントロールがない可能性があり、悪意のある指示はこれらのツールを意図しない方法で活用する可能性があります。詳細については、「 責任共有モデル」セクションを参照してください。
- 許可されたユーザー攻撃 – AWS アカウント境界内または接続されたツール内で操作する権限を持つユーザーは、エージェントに対する攻撃を試みる可能性が高くなります。これらのユーザーは、

ログやリソースタグなど、エージェントが消費するデータソースを変更できるため、エージェントが処理する悪意のある指示を簡単に埋め込むことができます。

これらのリスクを軽減するには:

1. エージェントスペースにデプロイする前に、カスタム MCP サーバーを慎重に確認してテストします。
   a. 読み取り専用アクションの実行のみが許可されていることを確認します。
   b. MCP サーバーによってアクセスされる外部ツールのユーザーが信頼されたエンティティであることを確認します。これは、これらのツールユーザーと AWS DevOps エージェントの間に確立された暗黙的な信頼関係が、DevOps エージェントとやり取りする AWS DevOps エージェントに依存しているためです。
2. エージェントにデータを提供するシステムへのアクセス権をユーザーに付与するときは、最小特権の原則を適用します。
3. エージェントスペースに接続されている MCP サーバーを定期的に監査する
4. 許可リストに登録された URLs から取得したコンテンツは、エージェントの動作を操作しようとする可能性があるため、許可リストには信頼できるソースのみを含めます。

統合セキュリティ

AWS DevOps Agent は複数の統合タイプをサポートし、それぞれに独自のセキュリティモデルがあります。

- ネイティブ双方向統合 – エージェントにデータを送信し、エージェントから更新を受信できる組み込み統合。これにより、ベンダーの認証方法が使用されます。
- MCP サーバー – OAuth 2.0 認証フローと API キーを使用して外部システムと安全に通信するリモートモデルコンテキストプロトコルサーバー。
- Webhook トリガー – チケットやオブザーバビリティシステムなどのリモートサービスからの調査トリガー。ウェブフックは、セキュリティにハッシュベースのメッセージ認証コード (HMAC) を使用します。
- アウトバウンド通信 – Slack やチケットシステムなどの統合はエージェントから更新を受け取りますが、双方向通信はまだサポートされていません。

登録プロバイダー

一部の外部ツールはアカウントレベルで認証され、アカウント内のすべてのエージェントスペース間で共有されます。これらのツールを登録すると、アカウントレベルで 1 回認証され、各エージェントスペースはその登録された接続内の特定のリソースに接続できます。

以下のツールは、アカウントレベルの登録を使用します。

- GitHub – 認証に OAuth フローを使用します。アカウントレベルで GitHub を登録すると、各エージェントスペースは GitHub 組織内の特定のリポジトリに接続できます。
- Dynatrace – OAuth トークン認証を使用します。アカウントレベルで Dynatrace を登録した後、各エージェントスペースは特定の Dynatrace 環境またはモニタリング設定に接続できます。
- Slack – OAuth トークン認証を使用します。アカウントレベルで Slack を登録した後、各エージェントスペースは特定の Slack チャネルチャネルに接続できます。
- Datadog – OAuth フローで MCP を認証に使用します。アカウントレベルで Datadog を登録すると、各エージェントスペースは特定の Datadog モニタリングリソースに接続できます。
- New Relic – API キー認証を使用します。アカウントレベルで New Relic を登録すると、各エージェントスペースは特定の New Relic モニタリング設定に接続できます。
- Splunk – ベアラートークン認証を使用します。アカウントレベルで Splunk を登録すると、各エージェントスペースは特定の Splunk データソースに接続できます。
- GitLab – アクセストークン認証を使用します。アカウントレベルで GitLab を登録すると、各エージェントスペースは特定の GitLab リポジトリに接続できます。
- ServiceNow – OAuth クライアントキー/トークン認証を使用します。アカウントレベルで ServiceNow を登録すると、各エージェントスペースは特定の ServiceNow インスタンスまたはチケットキューに接続できます。
- 一般公開アクセス可能なリモート MCP サーバー – OAuth フローを認証に使用します。アカウントレベルでリモート MCP サーバーを登録すると、各エージェントスペースはそのサーバーによって公開されている特定のリソースに接続できます。

ネットワーク接続

AWS DevOps Agent は、サードパーティーのシステムとリモート MCP サーバーに接続して、調査やその他のオペレーションを実行します。

AWS DevOps エージェントからシステムへのインバウンドトラフィック

AWS DevOps Agent は、サードパーティーシステムとリモート MCP サーバーへのアウトバウンド接続を開始し、インフラストラクチャへのインバウンドトラフィックとして到着します。このトラフィックを保護する方法は、ツールのホスト方法によって異なります。

- プライベートにホストされたツール – ツールが AWS VPC 内から到達可能な場合は、 AWS DevOps Agent のプライベート接続を使用して、トラフィックを AWS ネットワークに隔離し、パブリックインターネットから切り離すことができます。詳細については、「the section called “プライベートにホストされたツールへの接続”」を参照してください。
- パブリックにホストされたツール – ツールがパブリックインターネット経由で到達可能であり、IP 許可リストまたはファイアウォールルールを使用する場合は、次の AWS DevOps エージェントソース IP アドレスからのインバウンドトラフィックを許可する必要があります。
  - アジアパシフィック (シドニー) (ap-southeast-2)
    - `13.237.95.197`
    - `13.238.84.102`
  - アジアパシフィック (東京) (ap-northeast-1)
    - `13.192.12.233`
    - `35.74.181.230`
    - `57.183.50.158`
  - ヨーロッパ (フランクフルト) (eu-central-1)
    - `18.158.110.140`
    - `52.57.96.160`
    - `52.59.55.56`
  - 欧州 (アイルランド) (eu-west-1)
    - `34.251.85.24`
    - `52.30.157.157`
    - `52.51.192.222`
  - 米国東部 (バージニア北部) (us-east-1)
    - `34.228.181.128`
    - `44.219.176.187`
    - `54.226.244.221`
  - 米国西部 (オレゴン) (us-west-2)

- 34.212.16.133
- 52.89.67.212
- 54.187.135.61

VPC から AWS DevOps エージェントへのアウトバウンドトラフィック

AWS VPC から AWS DevOps エージェントへのアウトバウンドトラフィック (の使用などthe section called “Webhook による DevOps エージェントの呼び出し”) では、VPC エンドポイントを使用して、このネットワークトラフィックを AWS ネットワークに分離したままにできます。詳細については、「the section called “VPC エンドポイント (AWS PrivateLink)”」を参照してください。

責任共有モデル

AWS 責任

AWS は以下について責任を負います。

- エージェントによって取得されたデータのセキュリティを維持する
- エージェントで使用できるネイティブツールの保護
- AWS DevOps エージェントを実行するインフラストラクチャの保護

お客様の責任

お客様は以下について責任を負います。

- エージェントスペースへのユーザーアクセスの管理
- 悪意のあるプロンプトインジェクションを試みるために使用される可能性のあるログ、CloudTrail イベント、チケットなどのサービスやリソースなど、エージェントに入力を提供する外部システムの信頼されたユーザーへのアクセスを制限します。
- 接続されたすべてのデータソースに、プロンプトインジェクション攻撃の試みに使用される可能性が低い信頼できるデータがあることを確認します。
- bring-your-own MCP サーバー統合を安全に運用する
- エージェントに割り当てられた IAM ロールが適切にスコープ設定されていることを確認する
- オブザーバビリティログやその他のエージェントデータソースに保存する前に PII データを編集する

- Bringbring-your-own MCP サーバーなど、接続されたデータソースに読み取り専用アクセス許可のみを付与する推奨プラクティスに従う

データ使用量

AWS は、モデルをトレーニングしたり製品を改善したりするために、エージェントデータ、チャットメッセージ、または統合データソースからのデータを使用しません。 AWS DevOps エージェントスペースは、顧客の製品内フィードバックを使用してエージェントの応答と調査を改善しますが、サービス自体の改善には使用 AWS しません。

コンプライアンス

プレビューでは、 AWS DevOps Agent は SOC 2、PCI-DSS、ISO 27001、または FedRAMP などの標準に準拠していません。 は、後でどのコンプライアンス証明書が利用可能になるか AWS を発表します。

DevOps エージェント IAM アクセス許可

AWS DevOps エージェントは、サービス固有の AWS Identity and Access Management (IAM) アクションを使用して、その機能へのアクセスを制御します。これらのアクションにより、ユーザーが AWS DevOps エージェントコンソールと Operator Web App 内で実行できる操作が決まります。これは、エージェント自体がリソースの調査に使用する AWS サービス API アクセス許可とは異なります。

エージェントアクセスの制限の詳細については、「AWS アカウントのエージェントアクセスの制限」を参照してください。

エージェントスペース管理アクション

これらのアクションは、エージェントスペースの設定と管理へのアクセスを制御します。

- aidevops:GetAgentSpace – ユーザーは、設定、ステータス、関連するアカウントなど、エージェントスペースの詳細を表示できます。ユーザーは、 AWS マネジメントコンソールでエージェントスペースにアクセスするには、このアクセス許可が必要です。
- aidevops:GetAssociation – IAM ロールの設定や接続ステータスなど、特定のアカウントの関連付けに関する詳細をユーザーが表示できるようにします。

- aidevops:ListAssociations – ユーザーは、プライマリ AWS アカウントとセカンダリアカウントの両方を含む、エージェントスペースに設定されたすべてのアカウント関連付けを一覧表示できます。

調査アクションと実行アクション

これらのアクションは、インシデント調査機能へのアクセスを制御します。

- aidevops:ListExecutions – ユーザーは、タスクに関連する調査、緩和策、評価、チャット会話について、ID、ステータスなどの実行メタデータを表示できます。
- aidevops:ListJournalRecords – 調査、緩和、評価、チャット会話中に相談されたエージェントの推論ステップ、実行されたアクション、データソースを示す詳細なログへのアクセスをユーザーに許可します。これは、エージェントが結論にどのように達したかを理解するのに役立ちます。

チャット管理アクション

チャットを使用するには、次の IAM アクセス許可が必要です。

- aidevops:ListChats – ユーザーがチャット会話履歴を一覧表示してアクセスできるようにします。
- aidevops:CreateChat – ユーザーが新しいチャット会話を作成できるようにします。
- aidevops:SendMessage – ユーザーがクエリを送信し、ストリーミングレスポンスを受信できるようにします。

トポロジおよび検出アクション

これらのアクションは、アプリケーションリソースマッピング機能へのアクセスを制御します。

- aidevops:DiscoverTopology – ユーザーがエージェントスペースのトポロジ検出とマッピングをトリガーできるようにします。このアクションは、 AWS アカウントをスキャンし、アプリケーションリソーストポロジを構築するプロセスを開始します。

防止アクションとレコメンデーションアクション

これらのアクションは、防止機能へのアクセスを制御します。

- aidevops:ListGoals – ユーザーは、最近のインシデントパターンに基づいて、エージェントが取り組んでいる防止目標と目的を表示できます。
- aidevops:ListRecommendations – ユーザーは、優先度やカテゴリなど、防止機能によって生成されたすべての推奨事項を表示できます。
- aidevops:GetRecommendation – 特定のレコメンデーションに関する詳細情報の表示をユーザーに許可します。これには、防止したインシデントや実装ガイダンスが含まれます。

バックログタスク管理アクション

これらのアクションは、推奨事項をバックログタスクとして管理する機能を制御します。

- aidevops:CreateBacklogTask – ユーザーがインシデント調査または防止評価タスクを作成できるようにします。
- aidevops:UpdateBacklogTask – ユーザーが緩和計画を承認したり、アクティブな調査または評価をキャンセルしたりできます。
- aidevops:GetBacklogTask – ユーザーが特定のタスクに関する詳細を取得できるようにします。
- aidevops:ListBacklogTasks – ユーザーは、タスクタイプ、ステータス、優先度、または作成時間でフィルタリングされたエージェントスペースのタスクを一覧表示できます。

ナレッジ管理アクション

これらのアクションは、エージェントが調査中に使用できるカスタムナレッジを追加および管理する能力を制御します。

- aidevops:CreateKnowledgeItem – エージェントが参照するスキル、トラブルシューティングガイド、アプリケーション固有の情報などのカスタムナレッジ項目をユーザーが追加できるようにします。
- aidevops:ListKnowledgeItems – ユーザーがエージェントスペースに設定されたすべてのナレッジ項目を表示できるようにします。
- aidevops:GetKnowledgeItem – ユーザーが特定のナレッジ項目の詳細を取得できるようにします。
- aidevops:UpdateKnowledgeItem – ユーザーが既存のナレッジ項目を変更して情報を最新の状態に保つことができます。
- aidevops:DeleteKnowledgeItem – 関連性がなくなったナレッジ項目の削除をユーザーに許可します。

AWS 統合アクションのサポート

これらのアクションは、 AWS サポートケースとの統合を制御します。

- aidevops:InitiateChatForCase – ユーザーが調査から直接 AWS Support とのチャットセッションを開始し、インシデントに関するコンテキストを自動的に提供できるようにします。
- aidevops:EndChatForCase – ユーザーがアクティブな AWS サポートケースチャットセッションを終了できるようにします。
- aidevops:DescribeSupportLevel – ユーザーがアカウントの AWS サポートプランレベルをチェックして、利用可能なサポートオプションを決定できるようにします。

使用状況とモニタリングアクション

これらのアクションは、使用状況情報へのアクセスを制御します。

- aidevops:GetAccountUsage – ユーザーは、調査時間、防止評価時間、チャットリクエストの AWS DevOps エージェント月間クォータ、および当月の使用状況を表示できます。

一般的な IAM ポリシーの例

管理者ポリシー

このポリシーは、すべての AWS DevOps エージェント機能へのフルアクセスを許可します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "aidevops:*",
      "Resource": "*"
    }
  ]
}
```

オペレーターポリシー

このポリシーは、管理機能なしで調査および防止機能へのアクセスを許可します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "aidevops:GetAgentSpace",
        "aidevops:InvokeAgent",
        "aidevops:ListExecutions",
        "aidevops:ListJournalRecords",
        "aidevops:ListAssociations",
        "aidevops:GetAssociation",
        "aidevops:DiscoverTopology",
        "aidevops:ListRecommendations",
        "aidevops:GetRecommendation",
        "aidevops:CreateBacklogTask",
        "aidevops:UpdateBacklogTask",
        "aidevops:GetBacklogTask",
        "aidevops:ListBacklogTasks",
        "aidevops:ListKnowledgeItems",
        "aidevops:GetKnowledgeItem",
        "aidevops:InitiateChatForCase",
        "aidevops:EndChatForCase",
        "aidevops:ListChats",
        "aidevops:CreateChat",
        "aidevops:SendMessage",
        "aidevops:ListGoals",
        "aidevops:CreateKnowledgeItem",
        "aidevops:UpdateKnowledgeItem",
        "aidevops:DescribeSupportLevel",
        "aidevops:ListPendingMessages"
      ],
      "Resource": "*"
    }
  ]
}
```

読み取り専用ポリシー

このポリシーは、調査とレコメンデーションへのビューのみのアクセスを許可します。

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "aidevops:GetAgentSpace",
        "aidevops:ListAssociations",
        "aidevops:GetAssociation",
        "aidevops:ListExecutions",
        "aidevops:ListJournalRecords",
        "aidevops:ListRecommendations",
        "aidevops:GetRecommendation",
        "aidevops:ListBacklogTasks",
        "aidevops:GetBacklogTask",
        "aidevops:ListKnowledgeItems",
        "aidevops:GetKnowledgeItem",
        "aidevops:GetAccountUsage"
      ],
      "Resource": "*"
    }
  ]
}
```

AWS DevOps Agent のサービスにリンクされたロールの使用

AWS DevOps エージェントは、 AWS Identity and Access Management (IAM) サービスにリンクされたロールを使用します。サービスにリンクされたロールは、 AWS DevOps エージェントに直接リンクされた一意のタイプの IAM ロールです。サービスにリンクされたロールは、 AWS DevOps Agent によって事前定義されており、サービスがユーザーに代わって他の AWS サービスを呼び出すために必要なすべてのアクセス許可が含まれています。

サービスにリンクされたロールのアクセス許可

`AWSServiceRoleForAIDevOps` サービスにリンクされたロールは、`aidevops.amazonaws.com` サービスプリンシパルを信頼してロールを引き受けます。

ロールは、次のアクセス許可`AWSServiceRoleForAIDevOpsPolicy`を持つ 管理ポリシーを使用します。

- `cloudwatch:PutMetricData` – 使用状況メトリクスを `AWS/AIDevOps` CloudWatch 名前空間に発行します。`AWS/AIDevOps` 名前空間のみを許可する`cloudwatch:namespace`条件によってスコープされます。

- vpc-lattice:CreateResourceGateway – プライベート接続用の VPC Lattice リソースゲートウェイを作成します。サービスが AWSAIDevOpsManaged タグを持つリソースゲートウェイのみを作成できるように、 aws:RequestTag/AWSAIDevOpsManaged条件によってスコープされます。
- vpc-lattice:TagResource – VPC Lattice リソースゲートウェイにタグを付けます。aws:RequestTag/AWSAIDevOpsManaged 条件によってスコープされます。
- vpc-lattice:DeleteResourceGateway – VPC Lattice リソースゲートウェイを削除します。サービスが作成したリソースゲートウェイのみを削除できるように、 aws:ResourceTag/AWSAIDevOpsManaged条件によってスコープされます。
- vpc-lattice:GetResourceGateway – VPC Lattice リソースゲートウェイに関する情報を取得します。サービスが作成したリソースゲートウェイのみを読み取ることができるように、aws:ResourceTag/AWSAIDevOpsManaged条件によってスコープされます。
- ec2:DescribeVpcs、ec2:DescribeSubnets、 ec2:DescribeSecurityGroups – リソースゲートウェイの設定に必要な VPC ネットワークリソースに関する情報を取得します。EC2 API は Describe 呼び出しのリソースレベルのアクセス許可をサポートしていないため、これらの読み取り専用アクションはすべての VPC リソースに適用されます。
- iam:CreateServiceLinkedRole – リソースゲートウェイオペレーションに必要な VPC Lattice サービスにリンクされたロールを作成します。このアクセス許可はvpc-lattice.amazonaws.comサービスプリンシパルのみを対象としており、他のサービスのサービスにリンクされたロールの作成には使用できません。

サービスリンクロールの作成

AWSServiceRoleForAIDevOps サービスリンクロールを手動で作成する必要はありません。 AWS DevOps エージェントの使用を開始すると、サービスにリンクされたロールが作成されます。

サービスがユーザーに代わってロールを作成できるようにするには、 アクセスiam:CreateServiceLinkedRole許可が必要です。最小特権の原則に従うaidevops.amazonaws.comには、 iam:AWSServiceNameの条件を使用してこのアクセス許可の範囲を設定することをお勧めします。詳細については、「サービスにリンクされたロールのアクセス許可」を参照してください。

サービスリンクロールの編集

サービスにリンクされたロール AWSServiceRoleForAIDevOps を編集することはできません。ロールの作成後は、さまざまなエンティティがロールを名前で参照する可能性があるため、ロールの

名前を変更することはできません。ただし、IAM を使用してロールの説明を編集することはできます。詳細については、「サービスにリンクされたロールの編集」を参照してください。

サービスリンクロールの削除

AWS DevOps エージェントを使用する必要がなくなった場合は、AWSServiceRoleForAIDevOpsサービスにリンクされたロールを削除することをお勧めします。ロールを削除する前に、エージェントスペースで設定されたプライベート接続をすべて削除する必要があります。サービスにリンクされたロールを削除しても、サービスによって以前に作成されたでタグ付けされた VPC Lattice AWSAIDevOpsManaged リソースゲートウェイは自動的に削除されません。これらのリソースゲートウェイが不要になった場合は、手動で削除する必要があります。詳細については、「サービスにリンクされたロールの削除」を参照してください。

AWS AWS DevOps エージェントの管理ポリシー

AWS は、 によって作成および管理されるスタンドアロン IAM ポリシーを提供することで、多くの一般的なユースケースに対処します AWS。これらの AWS 管理ポリシーは、一般的なユースケースに必要なアクセス許可を付与するため、必要なアクセス許可を調査する必要がなくなります。詳細については、_IAM ユーザーガイド_ の「 AWS 管理ポリシー」を参照してください。

アカウントのユーザーにアタッチできる次の AWS 管理ポリシーは、 AWS DevOps エージェントに固有です。

AIDevOpsAgentReadOnlyAccess

AWS マネジメントコンソールを介して Amazon DevOps エージェントへの読み取り専用アクセスを提供します

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AIDevOpsAgentReadOnlyAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:Get*",
    "aidevops:List*",
    "aidevops:SearchServiceAccessibleResource"
   ],
   "Resource": "*"
  }
```

```
 ]
}
```

AIDevOpsAgentFullAccess

AWS マネジメントコンソールを介して Amazon DevOps エージェントへのフルアクセスを提供します

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AIDevOpsAgentSpaceAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:CreateAgentSpace",
    "aidevops:DeleteAgentSpace",
    "aidevops:GetAgentSpace",
    "aidevops:ListAgentSpaces",
    "aidevops:UpdateAgentSpace"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsServiceAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:DeregisterService",
    "aidevops:GetService",
    "aidevops:ListServices",
    "aidevops:RegisterService",
    "aidevops:SearchServiceAccessibleResource"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsAssociationAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:AssociateService",
    "aidevops:DisassociateService",
    "aidevops:GetAssociation",
    "aidevops:ListAssociations",
```

```
    "aidevops:UpdateAssociation",
    "aidevops:ValidateAwsAssociations"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsWebhookAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:ListWebhooks"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsOperatorAppAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:DisableOperatorApp",
    "aidevops:EnableOperatorApp",
    "aidevops:GetOperatorApp",
    "aidevops:UpdateOperatorAppIdpConfig"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsKnowledgeAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:CreateKnowledgeItem",
    "aidevops:DeleteKnowledgeItem",
    "aidevops:GetKnowledgeItem",
    "aidevops:ListKnowledgeItems",
    "aidevops:ListKnowledgeItemVersions",
    "aidevops:UpdateKnowledgeItem"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsBacklogAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:CreateBacklogTask",
    "aidevops:GetBacklogTask",
    "aidevops:ListBacklogTasks",
```

```
    "aidevops:ListGoals",
    "aidevops:UpdateBacklogTask",
    "aidevops:UpdateGoal"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsRecommendationAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:GetRecommendation",
    "aidevops:ListRecommendations",
    "aidevops:UpdateRecommendation"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsAgentChatAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:CreateChat",
    "aidevops:ListChats",
    "aidevops:ListPendingMessages",
    "aidevops:SendMessage"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsJournalAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:ListExecutions",
    "aidevops:ListJournalRecords"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsTopologyAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:DiscoverTopology"
   ],
   "Resource": "*"
  },
```

```
  {
   "Sid": "AIDevOpsSupportAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:DescribeSupportLevel",
    "aidevops:EndChatForCase",
    "aidevops:InitiateChatForCase"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsUsageAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:GetAccountUsage"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsTaggingAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:ListTagsForResource",
    "aidevops:TagResource",
    "aidevops:UntagResource"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsVendedLogs",
   "Effect": "Allow",
   "Action": [
    "aidevops:AllowVendedLogDeliveryForResource"
   ],
   "Resource": "*"
  }
 ]
}
```

AIDevOpsOperatorAppAccessPolicy

エージェントスペースの AWS DevOps オペレーターウェブアプリを使用するためのアクセスを提供します。

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowOperatorAgentSpaceActions",
   "Effect": "Allow",
   "Action": [
    "aidevops:GetAgentSpace",
    "aidevops:GetAssociation",
    "aidevops:ListAssociations",
    "aidevops:CreateBacklogTask",
    "aidevops:GetBacklogTask",
    "aidevops:UpdateBacklogTask",
    "aidevops:ListBacklogTasks",
    "aidevops:ListJournalRecords",
    "aidevops:DiscoverTopology",
    "aidevops:ListGoals",
    "aidevops:ListRecommendations",
    "aidevops:ListExecutions",
    "aidevops:GetRecommendation",
    "aidevops:UpdateRecommendation",
    "aidevops:CreateKnowledgeItem",
    "aidevops:ListKnowledgeItems",
    "aidevops:ListKnowledgeItemVersions",
    "aidevops:GetKnowledgeItem",
    "aidevops:UpdateKnowledgeItem",
    "aidevops:DeleteKnowledgeItem",
    "aidevops:ListPendingMessages",
    "aidevops:InitiateChatForCase",
    "aidevops:EndChatForCase",
    "aidevops:DescribeSupportLevel",
    "aidevops:ListChats",
    "aidevops:CreateChat",
    "aidevops:SendMessage"
   ],
   "Resource": "arn:aws:aidevops:*:*:agentspace/${aws:PrincipalTag/AgentSpaceId}",
   "Condition": {
    "StringEquals": {
     "aws:ResourceAccount": "${aws:PrincipalAccount}"
    }
   }
  },
  {
```

```
    "Sid": "AllowOperatorAccountActions",
    "Effect": "Allow",
    "Action": [
     "aidevops:GetAccountUsage"
    ],
    "Resource": "*",
    "Condition": {
     "StringEquals": {
      "aws:ResourceAccount": "${aws:PrincipalAccount}"
     }
    }
   },
   {
    "Sid": "AllowSupportOperatorActions",
    "Effect": "Allow",
    "Action": [
     "support:DescribeCases",
     "support:InitiateChatForCase",
     "support:DescribeSupportLevel"
    ],
    "Resource": "*",
    "Condition": {
     "StringEquals": {
      "aws:ResourceAccount": "${aws:PrincipalAccount}"
     }
    }
   }
  ]
}
```

AIDevOpsAgentAccessPolicy

AWS DevOps Agent が顧客 AWS リソースの調査と分析を実行するために必要なアクセス許可を提供します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AIOPSServiceAccess",
            "Effect": "Allow",
            "Action": [
                "access-analyzer:GetAnalyzer",
```

```
                "access-analyzer:List*",
                "acm-pca:Describe*",
                "acm-pca:GetCertificate",
                "acm-pca:GetCertificateAuthorityCertificate",
                "acm-pca:GetCertificateAuthorityCsr",
                "acm-pca:List*",
                "acm:DescribeCertificate",
                "acm:GetAccountConfiguration",
                "aidevops:GetKnowledgeItem",
                "aidevops:ListKnowledgeItems",
                "airflow:List*",
                "amplify:GetApp",
                "amplify:GetBranch",
                "amplify:GetDomainAssociation",
                "amplify:List*",
                "aoss:BatchGetCollection",
                "aoss:BatchGetLifecyclePolicy",
                "aoss:BatchGetVpcEndpoint",
                "aoss:GetAccessPolicy",
                "aoss:GetSecurityConfig",
                "aoss:GetSecurityPolicy",
                "aoss:List*",
                "appconfig:GetApplication",
                "appconfig:GetConfigurationProfile",
                "appconfig:GetEnvironment",
                "appconfig:GetHostedConfigurationVersion",
                "appconfig:List*",
                "appflow:Describe*",
                "appflow:List*",
                "application-autoscaling:Describe*",
                "application-signals:BatchGetServiceLevelObjectiveBudgetReport",
                "application-signals:GetService",
                "application-signals:GetServiceLevelObjective",
                "application-signals:List*",
                "applicationinsights:Describe*",
                "applicationinsights:List*",
                "apprunner:Describe*",
                "apprunner:List*",
                "appstream:Describe*",
                "appstream:List*",
                "appsync:GetApiAssociation",
                "appsync:GetDataSource",
                "appsync:GetDomainName",
                "appsync:GetFunction",
```

```
                "appsync:GetGraphqlApi",
                "appsync:GetGraphqlApiEnvironmentVariables",
                "appsync:GetIntrospectionSchema",
                "appsync:GetResolver",
                "appsync:GetSourceApiAssociation",
                "appsync:List*",
                "aps:Describe*",
                "aps:List*",
                "arc-zonal-shift:GetManagedResource",
                "arc-zonal-shift:List*",
                "athena:GetCapacityAssignmentConfiguration",
                "athena:GetCapacityReservation",
                "athena:GetDataCatalog",
                "athena:GetNamedQuery",
                "athena:GetPreparedStatement",
                "athena:GetWorkGroup",
                "athena:List*",
                "auditmanager:GetAssessment",
                "auditmanager:List*",
                "autoscaling:Describe*",
                "backup-gateway:GetHypervisor",
                "backup-gateway:List*",
                "backup:Describe*",
                "backup:GetBackupPlan",
                "backup:GetBackupSelection",
                "backup:GetBackupVaultAccessPolicy",
                "backup:GetBackupVaultNotifications",
                "backup:GetRestoreTestingPlan",
                "backup:GetRestoreTestingSelection",
                "backup:List*",
                "batch:DescribeComputeEnvironments",
                "batch:DescribeJobQueues",
                "batch:DescribeSchedulingPolicies",
                "batch:List*",
                "bedrock:GetAgent",
                "bedrock:GetAgentActionGroup",
                "bedrock:GetAgentAlias",
                "bedrock:GetAgentKnowledgeBase",
                "bedrock:GetDataSource",
                "bedrock:GetGuardrail",
                "bedrock:GetKnowledgeBase",
                "bedrock:List*",
                "budgets:Describe*",
                "budgets:List*",
```

```
                "ce:Describe*",
                "ce:GetAnomalyMonitors",
                "ce:GetAnomalySubscriptions",
                "ce:List*",
                "chatbot:Describe*",
                "chatbot:GetMicrosoftTeamsChannelConfiguration",
                "chatbot:List*",
                "cleanrooms-ml:GetTrainingDataset",
                "cleanrooms-ml:List*",
                "cleanrooms:GetAnalysisTemplate",
                "cleanrooms:GetCollaboration",
                "cleanrooms:GetConfiguredTable",
                "cleanrooms:GetConfiguredTableAnalysisRule",
                "cleanrooms:GetConfiguredTableAssociation",
                "cleanrooms:GetMembership",
                "cleanrooms:List*",
                "cloudformation:Describe*",
                "cloudformation:GetResource",
                "cloudformation:GetStackPolicy",
                "cloudformation:GetTemplate",
                "cloudformation:List*",
                "cloudfront:Describe*",
                "cloudfront:GetCachePolicy",
                "cloudfront:GetCloudFrontOriginAccessIdentity",
                "cloudfront:GetContinuousDeploymentPolicy",
                "cloudfront:GetDistribution",
                "cloudfront:GetDistributionConfig",
                "cloudfront:GetFunction",
                "cloudfront:GetKeyGroup",
                "cloudfront:GetMonitoringSubscription",
                "cloudfront:GetOriginAccessControl",
                "cloudfront:GetOriginRequestPolicy",
                "cloudfront:GetPublicKey",
                "cloudfront:GetRealtimeLogConfig",
                "cloudfront:GetResponseHeadersPolicy",
                "cloudfront:List*",
                "cloudtrail:Describe*",
                "cloudtrail:GetChannel",
                "cloudtrail:GetEventConfiguration",
                "cloudtrail:GetEventDataStore",
                "cloudtrail:GetEventSelectors",
                "cloudtrail:GetInsightSelectors",
                "cloudtrail:GetQueryResults",
                "cloudtrail:GetResourcePolicy",
```

```
                "cloudtrail:GetTrail",
                "cloudtrail:GetTrailStatus",
                "cloudtrail:List*",
                "cloudtrail:LookupEvents",
                "cloudtrail:StartQuery",
                "cloudwatch:Describe*",
                "cloudwatch:GenerateQuery",
                "cloudwatch:GetDashboard",
                "cloudwatch:GetInsightRuleReport",
                "cloudwatch:GetMetricData",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:GetMetricStream",
                "cloudwatch:GetService",
                "cloudwatch:GetServiceLevelObjective",
                "cloudwatch:List*",
                "codeartifact:Describe*",
                "codeartifact:GetDomainPermissionsPolicy",
                "codeartifact:GetRepositoryPermissionsPolicy",
                "codeartifact:List*",
                "codebuild:BatchGetFleets",
                "codebuild:List*",
                "codecommit:GetRepository",
                "codecommit:GetRepositoryTriggers",
                "codedeploy:BatchGetDeployments",
                "codedeploy:BatchGetDeploymentTargets",
                "codedeploy:GetApplication",
                "codedeploy:GetDeploymentConfig",
                "codedeploy:GetDeploymentTarget",
                "codedeploy:List*",
                "codeguru-profiler:Describe*",
                "codeguru-profiler:GetNotificationConfiguration",
                "codeguru-profiler:GetPolicy",
                "codeguru-profiler:List*",
                "codeguru-reviewer:Describe*",
                "codeguru-reviewer:List*",
                "codepipeline:GetPipeline",
                "codepipeline:GetPipelineState",
                "codepipeline:List*",
                "codestar-connections:GetConnection",
                "codestar-connections:GetRepositoryLink",
                "codestar-connections:GetSyncConfiguration",
                "codestar-connections:List*",
                "codestar-notifications:Describe*",
                "codestar-notifications:List*",
```

```
                "cognito-identity:DescribeIdentityPool",
                "cognito-identity:GetIdentityPoolRoles",
                "cognito-identity:ListIdentityPools",
                "cognito-identity:ListTagsForResource",
                "cognito-idp:AdminListGroupsForUser",
                "cognito-idp:DescribeIdentityProvider",
                "cognito-idp:DescribeResourceServer",
                "cognito-idp:DescribeRiskConfiguration",
                "cognito-idp:DescribeUserImportJob",
                "cognito-idp:DescribeUserPool",
                "cognito-idp:DescribeUserPoolDomain",
                "cognito-idp:GetGroup",
                "cognito-idp:GetLogDeliveryConfiguration",
                "cognito-idp:GetUICustomization",
                "cognito-idp:GetUserPoolMfaConfig",
                "cognito-idp:GetWebACLForResource",
                "cognito-idp:ListGroups",
                "cognito-idp:ListIdentityProviders",
                "cognito-idp:ListResourceServers",
                "cognito-idp:ListUserPoolClients",
                "cognito-idp:ListUserPools",
                "cognito-idp:ListTagsForResource",
                "comprehend:Describe*",
                "comprehend:List*",
                "config:Describe*",
                "config:GetStoredQuery",
                "config:List*",
                "connect:Describe*",
                "connect:GetTaskTemplate",
                "connect:List*",
                "databrew:Describe*",
                "databrew:List*",
                "datapipeline:Describe*",
                "datapipeline:GetPipelineDefinition",
                "datapipeline:List*",
                "datasync:Describe*",
                "datasync:List*",
                "deadline:GetFarm",
                "deadline:GetFleet",
                "deadline:GetLicenseEndpoint",
                "deadline:GetMonitor",
                "deadline:GetQueue",
                "deadline:GetQueueEnvironment",
                "deadline:GetQueueFleetAssociation",
```

```
                "deadline:GetStorageProfile",
                "deadline:List*",
                "detective:GetMembers",
                "detective:List*",
                "devicefarm:GetDevicePool",
                "devicefarm:GetInstanceProfile",
                "devicefarm:GetNetworkProfile",
                "devicefarm:GetProject",
                "devicefarm:GetTestGridProject",
                "devicefarm:GetVPCEConfiguration",
                "devicefarm:List*",
                "devops-guru:Describe*",
                "devops-guru:GetResourceCollection",
                "devops-guru:List*",
                "dms:Describe*",
                "dms:List*",
                "ds:Describe*",
                "dynamodb:Describe*",
                "dynamodb:GetResourcePolicy",
                "dynamodb:List*",
                "ec2:Describe*",
                "ec2:GetAssociatedEnclaveCertificateIamRoles",
                "ec2:GetIpamPoolAllocations",
                "ec2:GetIpamPoolCidrs",
                "ec2:GetManagedPrefixListEntries",
                "ec2:GetNetworkInsightsAccessScopeContent",
                "ec2:GetSnapshotBlockPublicAccessState",
                "ec2:GetTransitGatewayMulticastDomainAssociations",
                "ec2:GetTransitGatewayRouteTableAssociations",
                "ec2:GetTransitGatewayRouteTablePropagations",
                "ec2:GetVerifiedAccessEndpointPolicy",
                "ec2:GetVerifiedAccessGroupPolicy",
                "ec2:GetVerifiedAccessInstanceWebAcl",
                "ec2:SearchLocalGatewayRoutes",
                "ec2:SearchTransitGatewayRoutes",
                "ecr:Describe*",
                "ecr:GetLifecyclePolicy",
                "ecr:GetRegistryPolicy",
                "ecr:GetRepositoryPolicy",
                "ecr:List*",
                "ecs:Describe*",
                "ecs:List*",
                "eks:AccessKubernetesApi",
                "eks:Describe*",
```

```
                "eks:List*",
                "elasticache:Describe*",
                "elasticache:List*",
                "elasticbeanstalk:Describe*",
                "elasticbeanstalk:List*",
                "elasticfilesystem:Describe*",
                "elasticloadbalancing:GetResourcePolicy",
                "elasticloadbalancing:GetTrustStoreCaCertificatesBundle",
                "elasticloadbalancing:GetTrustStoreRevocationContent",
                "elasticloadbalancing:Describe*",
                "elasticmapreduce:Describe*",
                "elasticmapreduce:List*",
                "emr-containers:Describe*",
                "emr-containers:List*",
                "emr-serverless:GetApplication",
                "emr-serverless:List*",
                "es:Describe*",
                "es:List*",
                "events:Describe*",
                "events:List*",
                "evidently:GetExperiment",
                "evidently:GetFeature",
                "evidently:GetLaunch",
                "evidently:GetProject",
                "evidently:GetSegment",
                "evidently:List*",
                "firehose:Describe*",
                "firehose:List*",
                "fis:GetExperimentTemplate",
                "fis:GetTargetAccountConfiguration",
                "fis:List*",
                "fms:GetNotificationChannel",
                "fms:GetPolicy",
                "fms:List*",
                "forecast:Describe*",
                "forecast:List*",
                "frauddetector:BatchGetVariable",
                "frauddetector:Describe*",
                "frauddetector:GetDetectors",
                "frauddetector:GetDetectorVersion",
                "frauddetector:GetEntityTypes",
                "frauddetector:GetEventTypes",
                "frauddetector:GetExternalModels",
                "frauddetector:GetLabels",
```

```
                "frauddetector:GetListElements",
                "frauddetector:GetListsMetadata",
                "frauddetector:GetModelVersion",
                "frauddetector:GetOutcomes",
                "frauddetector:GetRules",
                "frauddetector:GetVariables",
                "frauddetector:List*",
                "fsx:Describe*",
                "gamelift:Describe*",
                "gamelift:List*",
                "globalaccelerator:Describe*",
                "globalaccelerator:List*",
                "glue:GetDatabase",
                "glue:GetDatabases",
                "glue:GetJob",
                "glue:GetRegistry",
                "glue:GetSchema",
                "glue:GetSchemaVersion",
                "glue:GetTable",
                "glue:GetTags",
                "glue:GetTrigger",
                "glue:List*",
                "glue:querySchemaVersionMetadata",
                "grafana:Describe*",
                "grafana:List*",
                "greengrass:Describe*",
                "greengrass:GetDeployment",
                "greengrass:List*",
                "groundstation:GetConfig",
                "groundstation:GetDataflowEndpointGroup",
                "groundstation:GetMissionProfile",
                "groundstation:List*",
                "guardduty:GetDetector",
                "guardduty:GetFilter",
                "guardduty:GetIPSet",
                "guardduty:GetMalwareProtectionPlan",
                "guardduty:GetMasterAccount",
                "guardduty:GetMembers",
                "guardduty:GetThreatIntelSet",
                "guardduty:List*",
                "health:DescribeEvents",
                "health:DescribeEventDetails",
                "healthlake:Describe*",
                "healthlake:List*",
```

```
                "iam:GetGroup",
                "iam:GetGroupPolicy",
                "iam:GetInstanceProfile",
                "iam:GetLoginProfile",
                "iam:GetOpenIDConnectProvider",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:GetSAMLProvider",
                "iam:GetServerCertificate",
                "iam:GetServiceLinkedRoleDeletionStatus",
                "iam:GetUser",
                "iam:GetUserPolicy",
                "iam:ListAttachedRolePolicies",
                "iam:ListOpenIDConnectProviders",
                "iam:ListRolePolicies",
                "iam:ListRoles",
                "iam:ListServerCertificates",
                "iam:ListVirtualMFADevices",
                "identitystore:DescribeGroup",
                "identitystore:DescribeGroupMembership",
                "identitystore:ListGroupMemberships",
                "identitystore:ListGroups",
                "imagebuilder:GetComponent",
                "imagebuilder:GetContainerRecipe",
                "imagebuilder:GetDistributionConfiguration",
                "imagebuilder:GetImage",
                "imagebuilder:GetImagePipeline",
                "imagebuilder:GetImageRecipe",
                "imagebuilder:GetInfrastructureConfiguration",
                "imagebuilder:GetLifecyclePolicy",
                "imagebuilder:GetWorkflow",
                "imagebuilder:List*",
                "inspector2:List*",
                "inspector:Describe*",
                "inspector:List*",
                "internetmonitor:GetMonitor",
                "internetmonitor:List*",
                "iot:Describe*",
                "iot:GetPackage",
                "iot:GetPackageVersion",
                "iot:GetPolicy",
                "iot:GetThingShadow",
```

```
                "iot:GetTopicRule",
                "iot:GetTopicRuleDestination",
                "iot:GetV2LoggingOptions",
                "iot:List*",
                "iotanalytics:Describe*",
                "iotanalytics:List*",
                "iotevents:Describe*",
                "iotevents:List*",
                "iotsitewise:Describe*",
                "iotsitewise:List*",
                "iotwireless:GetDestination",
                "iotwireless:GetDeviceProfile",
                "iotwireless:GetFuotaTask",
                "iotwireless:GetMulticastGroup",
                "iotwireless:GetNetworkAnalyzerConfiguration",
                "iotwireless:GetServiceProfile",
                "iotwireless:GetWirelessDevice",
                "iotwireless:GetWirelessGateway",
                "iotwireless:GetWirelessGatewayTaskDefinition",
                "iotwireless:List*",
                "ivs:GetChannel",
                "ivs:GetEncoderConfiguration",
                "ivs:GetPlaybackRestrictionPolicy",
                "ivs:GetRecordingConfiguration",
                "ivs:GetStage",
                "ivs:List*",
                "ivschat:GetLoggingConfiguration",
                "ivschat:GetRoom",
                "ivschat:List*",
                "kafka:Describe*",
                "kafka:GetClusterPolicy",
                "kafka:List*",
                "kafkaconnect:Describe*",
                "kafkaconnect:List*",
                "kendra:Describe*",
                "kendra:List*",
                "kinesis:Describe*",
                "kinesis:GetResourcePolicy",
                "kinesis:List*",
                "kinesisanalytics:Describe*",
                "kinesisanalytics:List*",
                "kinesisvideo:Describe*",
                "kms:DescribeKey",
                "kms:ListResourceTags",
```

```
                "kms:ListKeys",
                "kms:GetKeyPolicy",
                "kms:GetKeyRotationStatus",
                "kms:ListAliases",
                "kms:ListKeyRotations",
                "lakeformation:Describe*",
                "lakeformation:GetLFTag",
                "lakeformation:GetResourceLFTags",
                "lakeformation:List*",
                "lambda:GetAlias",
                "lambda:GetCodeSigningConfig",
                "lambda:GetEventSourceMapping",
                "lambda:GetFunctionCodeSigningConfig",
                "lambda:GetFunctionConfiguration",
                "lambda:GetFunctionEventInvokeConfig",
                "lambda:GetFunctionRecursionConfig",
                "lambda:GetFunctionUrlConfig",
                "lambda:GetLayerVersion",
                "lambda:GetLayerVersionPolicy",
                "lambda:GetPolicy",
                "lambda:GetProvisionedConcurrencyConfig",
                "lambda:GetRuntimeManagementConfig",
                "lambda:List*",
                "launchwizard:GetDeployment",
                "launchwizard:List*",
                "license-manager:GetLicense",
                "license-manager:List*",
                "lightsail:GetAlarms",
                "lightsail:GetBuckets",
                "lightsail:GetCertificates",
                "lightsail:GetContainerServices",
                "lightsail:GetDisk",
                "lightsail:GetDisks",
                "lightsail:GetInstance",
                "lightsail:GetInstances",
                "lightsail:GetLoadBalancer",
                "lightsail:GetLoadBalancers",
                "lightsail:GetLoadBalancerTlsCertificates",
                "lightsail:GetStaticIp",
                "lightsail:GetStaticIps",
                "logs:Describe*",
                "logs:FilterLogEvents",
                "logs:GetDataProtectionPolicy",
                "logs:GetDelivery",
```

```
                "logs:GetDeliveryDestination",
                "logs:GetDeliveryDestinationPolicy",
                "logs:GetDeliverySource",
                "logs:GetLogAnomalyDetector",
                "logs:GetLogDelivery",
                "logs:GetLogGroupFields",
                "logs:GetQueryResults",
                "logs:List*",
                "logs:StartQuery",
                "logs:StopLiveTail",
                "logs:StopQuery",
                "logs:TestMetricFilter",
                "m2:GetApplication",
                "m2:GetEnvironment",
                "m2:List*",
                "macie2:GetAllowList",
                "macie2:GetCustomDataIdentifier",
                "macie2:GetFindingsFilter",
                "macie2:GetMacieSession",
                "macie2:List*",
                "mediaconnect:Describe*",
                "mediaconnect:List*",
                "medialive:Describe*",
                "medialive:GetCloudWatchAlarmTemplate",
                "medialive:GetCloudWatchAlarmTemplateGroup",
                "medialive:GetEventBridgeRuleTemplate",
                "medialive:GetEventBridgeRuleTemplateGroup",
                "medialive:GetSignalMap",
                "medialive:List*",
                "mediapackage-vod:Describe*",
                "mediapackage-vod:List*",
                "mediapackage:Describe*",
                "mediapackage:List*",
                "mediapackagev2:GetChannel",
                "mediapackagev2:GetChannelGroup",
                "mediapackagev2:GetChannelPolicy",
                "mediapackagev2:GetOriginEndpoint",
                "mediapackagev2:GetOriginEndpointPolicy",
                "mediapackagev2:List*",
                "memorydb:Describe*",
                "memorydb:List*",
                "mobiletargeting:GetInAppTemplate",
                "mobiletargeting:List*",
                "mq:Describe*",
```

```
                "mq:List*",
                "network-firewall:Describe*",
                "network-firewall:List*",
                "networkmanager:Describe*",
                "networkmanager:GetConnectAttachment",
                "networkmanager:GetConnectPeer",
                "networkmanager:GetCoreNetwork",
                "networkmanager:GetCoreNetworkPolicy",
                "networkmanager:GetCustomerGatewayAssociations",
                "networkmanager:GetDevices",
                "networkmanager:GetLinkAssociations",
                "networkmanager:GetLinks",
                "networkmanager:GetSites",
                "networkmanager:GetSiteToSiteVpnAttachment",
                "networkmanager:GetTransitGatewayPeering",
                "networkmanager:GetTransitGatewayRegistrations",
                "networkmanager:GetTransitGatewayRouteTableAttachment",
                "networkmanager:GetVpcAttachment",
                "networkmanager:List*",
                "oam:GetLink",
                "oam:GetSink",
                "oam:GetSinkPolicy",
                "oam:List*",
                "omics:GetAnnotationStore",
                "omics:GetReferenceStore",
                "omics:GetRunGroup",
                "omics:GetSequenceStore",
                "omics:GetVariantStore",
                "omics:GetWorkflow",
                "omics:List*",
                "organizations:Describe*",
                "organizations:List*",
                "osis:GetPipeline",
                "osis:List*",
                "payment-cryptography:GetAlias",
                "payment-cryptography:GetKey",
                "payment-cryptography:List*",
                "pca-connector-ad:GetConnector",
                "pca-connector-ad:GetDirectoryRegistration",
                "pca-connector-ad:GetServicePrincipalName",
                "pca-connector-ad:GetTemplate",
                "pca-connector-ad:GetTemplateGroupAccessControlEntry",
                "pca-connector-ad:List*",
                "pca-connector-scep:GetChallengeMetadata",
```

```
                "pca-connector-scep:GetConnector",
                "pca-connector-scep:List*",
                "personalize:Describe*",
                "personalize:List*",
                "pi:DescribeDimensionKeys",
                "pi:GetResourceMetadata",
                "pi:GetResourceMetrics",
                "pi:ListAvailableResourceDimensions",
                "pi:ListAvailableResourceMetrics",
                "pipes:Describe*",
                "pipes:List*",
                "proton:GetEnvironmentTemplate",
                "proton:GetServiceTemplate",
                "proton:List*",
                "qbusiness:GetApplication",
                "qbusiness:GetDataSource",
                "qbusiness:GetIndex",
                "qbusiness:GetPlugin",
                "qbusiness:GetRetriever",
                "qbusiness:GetWebExperience",
                "qbusiness:List*",
                "ram:GetPermission",
                "ram:GetResourceShares",
                "ram:List*",
                "rds:Describe*",
                "rds:List*",
                "redshift-serverless:GetNamespace",
                "redshift-serverless:GetWorkgroup",
                "redshift-serverless:List*",
                "redshift:Describe*",
                "refactor-spaces:GetApplication",
                "refactor-spaces:GetEnvironment",
                "refactor-spaces:GetRoute",
                "refactor-spaces:List*",
                "rekognition:Describe*",
                "rekognition:List*",
                "resiliencehub:Describe*",
                "resiliencehub:List*",
                "resource-explorer-2:GetDefaultView",
                "resource-explorer-2:GetIndex",
                "resource-explorer-2:GetView",
                "resource-explorer-2:List*",
                "resource-explorer-2:Search",
                "resource-groups:GetGroup",
```

```
                "resource-groups:GetGroupConfiguration",
                "resource-groups:GetGroupQuery",
                "resource-groups:GetTags",
                "resource-groups:List*",
                "route53-recovery-control-config:Describe*",
                "route53-recovery-control-config:List*",
                "route53-recovery-readiness:GetCell",
                "route53-recovery-readiness:GetReadinessCheck",
                "route53-recovery-readiness:GetRecoveryGroup",
                "route53-recovery-readiness:GetResourceSet",
                "route53-recovery-readiness:List*",
                "route53:GetDNSSEC",
                "route53:GetHealthCheck",
                "route53:GetHealthCheckStatus",
                "route53:GetHostedZone",
                "route53:List*",
                "route53profiles:GetProfile",
                "route53profiles:GetProfileAssociation",
                "route53profiles:GetProfileResourceAssociation",
                "route53profiles:List*",
                "route53resolver:GetFirewallDomainList",
                "route53resolver:GetFirewallRuleGroup",
                "route53resolver:GetFirewallRuleGroupAssociation",
                "route53resolver:GetOutpostResolver",
                "route53resolver:GetResolverConfig",
                "route53resolver:GetResolverQueryLogConfig",
                "route53resolver:GetResolverQueryLogConfigAssociation",
                "route53resolver:GetResolverRule",
                "route53resolver:GetResolverRuleAssociation",
                "route53resolver:List*",
                "rum:GetAppMonitor",
                "rum:List*",
                "s3-outposts:ListEndpoints",
                "s3-outposts:ListOutpostsWithS3",
                "s3:GetAccessGrant",
                "s3:GetAccessGrantsInstance",
                "s3:GetAccessGrantsLocation",
                "s3:GetAccessPoint",
                "s3:GetAccessPointConfigurationForObjectLambda",
                "s3:GetAccessPointForObjectLambda",
                "s3:GetAccessPointPolicy",
                "s3:GetAccessPointPolicyForObjectLambda",
                "s3:GetAccessPointPolicyStatusForObjectLambda",
                "s3:GetBucketAbac",
```

```
                "s3:GetBucketAcl",
                "s3:GetBucketCORS",
                "s3:GetBucketLocation",
                "s3:GetBucketLogging",
                "s3:GetBucketMetadataTableConfiguration",
                "s3:GetBucketNotification",
                "s3:GetBucketObjectLockConfiguration",
                "s3:GetBucketOwnershipControls",
                "s3:GetBucketPolicy",
                "s3:GetBucketPublicAccessBlock",
                "s3:GetBucketTagging",
                "s3:GetBucketVersioning",
                "s3:GetEncryptionConfiguration",
                "s3:GetLifecycleConfiguration",
                "s3:GetMultiRegionAccessPoint",
                "s3:GetMultiRegionAccessPointPolicy",
                "s3:GetMultiRegionAccessPointPolicyStatus",
                "s3:GetReplicationConfiguration",
                "s3:GetStorageLensConfiguration",
                "s3:GetStorageLensConfigurationTagging",
                "s3:GetStorageLensGroup",
                "s3:ListAllMyBuckets",
                "sagemaker:Describe*",
                "sagemaker:List*",
                "scheduler:GetSchedule",
                "scheduler:GetScheduleGroup",
                "scheduler:List*",
                "schemas:Describe*",
                "schemas:GetResourcePolicy",
                "schemas:List*",
                "secretsmanager:Describe*",
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:List*",
                "securityhub:BatchGetAutomationRules",
                "securityhub:BatchGetSecurityControls",
                "securityhub:Describe*",
                "securityhub:GetConfigurationPolicy",
                "securityhub:GetConfigurationPolicyAssociation",
                "securityhub:GetEnabledStandards",
                "securityhub:GetFindingAggregator",
                "securityhub:GetInsights",
                "securityhub:List*",
                "securitylake:GetSubscriber",
                "securitylake:List*",
```

```
                "servicecatalog:Describe*",
                "servicecatalog:GetApplication",
                "servicecatalog:GetAttributeGroup",
                "servicecatalog:List*",
                "servicequotas:GetServiceQuota",
                "ses:Describe*",
                "ses:GetAccount",
                "ses:GetAddonInstance",
                "ses:GetAddonSubscription",
                "ses:GetArchive",
                "ses:GetConfigurationSet",
                "ses:GetConfigurationSetEventDestinations",
                "ses:GetContactList",
                "ses:GetDedicatedIpPool",
                "ses:GetDedicatedIps",
                "ses:GetEmailIdentity",
                "ses:GetEmailTemplate",
                "ses:GetIngressPoint",
                "ses:GetRelay",
                "ses:GetRuleSet",
                "ses:GetTemplate",
                "ses:GetTrafficPolicy",
                "ses:List*",
                "shield:Describe*",
                "shield:List*",
                "signer:GetSigningProfile",
                "signer:List*",
                "sns:GetDataProtectionPolicy",
                "sns:GetSubscriptionAttributes",
                "sns:GetTopicAttributes",
                "sns:List*",
                "sqs:GetQueueAttributes",
                "sqs:GetQueueUrl",
                "sqs:List*",
                "ssm-contacts:GetContact",
                "ssm-contacts:GetContactChannel",
                "ssm-contacts:List*",
                "ssm-incidents:GetReplicationSet",
                "ssm-incidents:GetResponsePlan",
                "ssm-incidents:List*",
                "ssm-sap:GetApplication",
                "ssm-sap:List*",
                "ssm:Describe*",
                "ssm:GetDefaultPatchBaseline",
```

```
                "ssm:GetDocument",
                "ssm:GetParameters",
                "ssm:GetPatchBaseline",
                "ssm:GetResourcePolicies",
                "ssm:List*",
                "sso:GetInlinePolicyForPermissionSet",
                "sso:GetManagedApplicationInstance",
                "sso:GetPermissionsBoundaryForPermissionSet",
                "sso:GetSharedSsoConfiguration",
                "sso:ListAccountAssignments",
                "sso:ListApplicationAssignments",
                "sso:ListApplications",
                "sso:ListCustomerManagedPolicyReferencesInPermissionSet",
                "sso:ListInstances",
                "sso:ListManagedPoliciesInPermissionSet",
                "sso:ListTagsForResource",
                "states:GetExecutionHistory",
                "states:Describe*",
                "states:List*",
                "support:CreateCase",
                "support:DescribeCases",
                "synthetics:Describe*",
                "synthetics:GetCanary",
                "synthetics:GetCanaryRuns",
                "synthetics:GetGroup",
                "synthetics:List*",
                "tag:GetResources",
                "timestream:Describe*",
                "timestream:List*",
                "transfer:Describe*",
                "transfer:List*",
                "verifiedpermissions:GetIdentitySource",
                "verifiedpermissions:GetPolicy",
                "verifiedpermissions:GetPolicyStore",
                "verifiedpermissions:GetPolicyTemplate",
                "verifiedpermissions:GetSchema",
                "verifiedpermissions:List*",
                "vpc-lattice:GetAccessLogSubscription",
                "vpc-lattice:GetAuthPolicy",
                "vpc-lattice:GetListener",
                "vpc-lattice:GetResourcePolicy",
                "vpc-lattice:GetRule",
                "vpc-lattice:GetService",
                "vpc-lattice:GetServiceNetwork",
```

```
                "vpc-lattice:GetServiceNetworkServiceAssociation",
                "vpc-lattice:GetServiceNetworkVpcAssociation",
                "vpc-lattice:GetTargetGroup",
                "vpc-lattice:List*",
                "wafv2:GetIPSet",
                "wafv2:GetLoggingConfiguration",
                "wafv2:GetRegexPatternSet",
                "wafv2:GetRuleGroup",
                "wafv2:GetWebACL",
                "wafv2:GetWebACLForResource",
                "wafv2:List*",
                "workspaces-web:GetBrowserSettings",
                "workspaces-web:GetIdentityProvider",
                "workspaces-web:GetNetworkSettings",
                "workspaces-web:GetPortal",
                "workspaces-web:GetPortalServiceProviderMetadata",
                "workspaces-web:GetTrustStore",
                "workspaces-web:GetUserAccessLoggingSettings",
                "workspaces-web:GetUserSettings",
                "workspaces-web:List*",
                "workspaces:Describe*",
                "xray:BatchGetTraces",
                "xray:GetGroup",
                "xray:GetGroups",
                "xray:GetSamplingRules",
                "xray:GetServiceGraph",
                "xray:GetTraceSummaries",
                "xray:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AIOPSAPIGatewayAccess",
            "Effect": "Allow",
            "Action": [
                "apigateway:GET"
            ],
            "Resource": [
                "arn:aws:apigateway:*::/restapis",
                "arn:aws:apigateway:*::/restapis/*",
                "arn:aws:apigateway:*::/restapis/*/deployments",
                "arn:aws:apigateway:*::/restapis/*/deployments/*",
                "arn:aws:apigateway:*::/restapis/*/resources/*/methods/*/integrations",
```

```
                "arn:aws:apigateway:*::/restapis/*/resources/*/methods/*/integrations/
*",
                "arn:aws:apigateway:*::/restapis/*/stages",
                "arn:aws:apigateway:*::/restapis/*/stages/*",
                "arn:aws:apigateway:*::/apis",
                "arn:aws:apigateway:*::/apis/*",
                "arn:aws:apigateway:*::/apis/*/deployments",
                "arn:aws:apigateway:*::/apis/*/deployments/*",
                "arn:aws:apigateway:*::/apis/*/integrations",
                "arn:aws:apigateway:*::/apis/*/integrations/*",
                "arn:aws:apigateway:*::/apis/*/stages",
                "arn:aws:apigateway:*::/apis/*/stages/*",
                "arn:aws:apigateway:*::/domainnames/*"
            ]
        }
    ]
}
```

AWS アカウントでのエージェントアクセスの制限

AWS DevOps エージェントは IAM ロールを使用して、インシデント調査と予防的評価中に AWS リソースを検出して記述します。これらのロールにアタッチされた IAM ポリシーを設定することで、エージェントが持つアクセスレベルを制御できます。アプリケーショントポロジーには、エージェントがアクセスできるすべての情報が表示されるわけではありません。IAM ポリシーは、エージェントがアクセスできる AWS サービス APIsとリソースを実際に制限する唯一の方法です。

AWS DevOps エージェントの IAM ロールについて

AWS DevOps エージェントは IAM ロールを使用して、次の 2 種類のアカウントのリソースにアクセスします。

- プライマリアカウントロール – エージェントスペースを作成する AWS アカウントのリソースへのアクセス権をエージェントに付与します。
- セカンダリアカウントロール – エージェントスペースに接続する追加の AWS アカウントのリソースへのアクセス権をエージェントに付与します。

どちらのタイプのアカウントでも、エージェントがアクセスできる AWS サービスを制限し、それらのサービス内の特定のリソースへのアクセスを制限し、エージェントが操作できるリージョンを制御できます。

リソースの境界の選択

リソースアクセスを制限する場合は、エージェントがアプリケーションインシデントを正常に調査するための十分なアクセス許可を含める必要があります。これには、以下が含まれます。

- エージェントがモニタリングおよび調査する必要がある対象範囲内のアプリケーションのすべてのリソース
- これらのアプリケーションが依存するすべてのサポートインフラストラクチャ

インフラストラクチャのサポートには以下が含まれます。

- ネットワークコンポーネント (VPCs、サブネット、ロードバランサー、API ゲートウェイ)
- データストア (データベース、キャッシュ、オブジェクトストレージ)
- コンピューティングリソース (EC2 インスタンス、Lambda 関数、コンテナ)
- サービスのモニタリングと ログ記録 (CloudWatch、CloudTrail)
- アクセス許可を理解するために必要な Identity and Access Management リソース

アクセスを狭く制限しすぎると、エージェントは定義された境界外のサポートインフラストラクチャに起因する根本原因を特定できない可能性があります。

サービスアクセスの制限

エージェントのロールにアタッチされている IAM ポリシーを変更することで、エージェントがアクセスできる AWS サービスを制限できます。カスタムポリシーを作成するときは、次のベストプラクティスに従ってください。

- 読み取り専用アクセス許可のみを付与する – エージェントは調査中にリソース設定、メトリクス、ログを読み取る必要があります。エージェントがリソースを変更または削除できるようにするアクセス許可を付与することは避けてください。
- 必要なサービスの制限 – アプリケーションに関連するリソースを含む AWS サービスのみを含めます。たとえば、アプリケーションが Amazon RDS を使用していない場合は、ポリシーに RDS アクセス許可を含めないでください。
- ワイルドカードの代わりに特定のアクションを使用する – アクセス`service:*`許可を付与する代わりに、 `cloudwatch:GetMetricData`や などの個々のアクションを指定します`ec2:DescribeInstances`。

特定のサービスに制限するポリシーの例:

```
json

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:GetMetricData",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:DescribeAlarms",
        "logs:GetLogEvents",
        "logs:FilterLogEvents",
        "ec2:DescribeInstances",
        "lambda:GetFunction",
        "lambda:GetFunctionConfiguration"
      ],
      "Resource": "*"
    }
  ]
}
```

リソースアクセスの制限

エージェントをサービス内の特定のリソースに制限するには、IAM ポリシーでリソースレベルのアクセス許可を使用します。これにより、特定のパターンに一致するリソースにのみアクセス権を付与できます。

リソース ARN パターンの使用:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "lambda:GetFunction",
        "lambda:GetFunctionConfiguration"
      ],
      "Resource": "arn:aws:lambda:*:*:function:production-*"
```

```
    }
  ]
}
```

この例では、「production-」で始まる名前の Lambda 関数にのみアクセスするようにエージェントを制限します。

タグベースの制限の使用:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeInstanceStatus"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Environment": "production"
        }
      }
    }
  ]
}
```

この例では、エージェントが でタグ付けされた EC2 インスタンスのみにアクセスするように制限していますEnvironment=production。

リージョンアクセスの制限

エージェントがアクセスできる AWS リージョンを制限するには、IAM ポリシーで aws:RequestedRegion条件キーを使用します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```
        "ec2:Describe*",
        "lambda:Get*",
        "cloudwatch:Get*"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestedRegion": [
            "us-east-1",
            "us-west-2"
          ]
        }
      }
    }
  ]
}
```

この例では、us-east-1 および us-west-2 リージョンのリソースにのみアクセスするようにエージェントを制限します。

カスタム IAM ポリシーの作成

エージェントスペースを作成するとき、またはセカンダリアカウントを追加するときに、ポリシーテンプレートを使用してカスタム IAM ロールを作成するオプションがあります。これにより、最小特権の原則を実装できます。

エージェントスペースを作成する場合

AWS マネジメントコンソールの DevOps エージェントコンソールから..。

- ポリシードキュメントを使用して新しい DevOps エージェントロールを作成し、手順に従います。

エージェントスペースを編集する場合

AWS マネジメントコンソールの DevOps エージェントコンソールから..。

- 機能タブを選択する
- クラウドセクションから編集するセカンダリアカウントを選択し、編集をクリックします
- テンプレートを使用して新しい DevOps エージェントポリシーを作成し、手順に従います。

カスタムポリシーのベストプラクティス

- 読み取り専用アクセス許可のみを付与する – リソースの変更または削除を許可するアクセス許可は避けてください
- 可能な場合はリソースレベルのアクセス許可を使用する — ARN パターンまたはタグを使用して特定のリソースへのアクセスを制限する
- アクセス許可の定期的なレビューと監査 – エージェントの IAM ポリシーを定期的にレビューして、セキュリティ要件に合致していることを確認します。

IAM アイデンティティセンター認証の設定

IAM アイデンティティセンター認証は、 AWS DevOps エージェントスペースウェブアプリケーションへのユーザーアクセスを一元的に管理する方法を提供します。このガイドでは、IAM Identity Center 認証を設定してユーザーを管理する方法について説明します。

前提条件

IAM Identity Center 認証を設定する前に、以下を確認してください。

- 組織またはアカウントで有効になっている IAM Identity Center
- AWS DevOps Agent の管理者権限
- エージェントスペースが設定済みまたは作成準備完了

認証オプション

AWS DevOps エージェントには、エージェントスペースウェブアプリケーションにアクセスするための 2 つの認証方法があります。

IAM Identity Center 認証 – 本番環境に推奨されます。一元化されたユーザー管理、外部 ID プロバイダーとの統合、最大 12 時間のセッションを提供します。

管理者アクセス (IAM 認証) – 初期セットアップと設定中に管理者にクイックアクセスを提供します。セッションは 30 分に制限されています。

エージェントスペースの作成中に IAM アイデンティティセンターを設定する

エージェントスペースを作成するときは、アクセスタブで IAM アイデンティティセンター認証を設定できます。

ステップ 1: ウェブアプリ設定に移動する

1. エージェントスペースの詳細と AWS アカウントアクセスを設定したら、アクセスタブに進みます。
2. 「IAM アイデンティティセンターを接続する」と「管理者アクセス」の 2 つのセクションが表示されます。

ステップ 2: IAM Identity Center 統合を設定する

「〔エージェントスペース] を IAM アイデンティティセンターに接続する」セクションで、次の操作を行います。

1. IAM Identity Center インスタンスを確認する – コンソールには、ウェブアプリのユーザーアクセスを管理する Identity Center インスタンス (など) が表示されますssoins-7223a9580931edbe。最も近い IAM Identity Center インスタンスは自動的に事前入力されます。
2. IAM Identity Center アプリケーションロール名オプションを選択する – 次の 3 つのオプションのいずれかを選択します。

新しい DevOps エージェントロールを自動作成します (推奨):

- システムは、適切なアクセス許可を持つ新しいサービスロールを自動的に作成します。
- これは最も簡単なオプションで、ほとんどのユースケースで機能します。

既存のロールを割り当てます。

- 既に作成した既存の IAM ロールを使用する
- システムは、ロールに必要なアクセス許可があることを確認します。
- 組織に AWS DevOps Agent のロールが事前に作成されている場合は、このオプションを選択します。

ポリシーテンプレートを使用して新しい DevOps エージェントロールを作成します。

- 提供されたポリシーの詳細を使用して、IAM コンソールで独自のカスタムロールを作成します。
- ロールのアクセス許可をカスタマイズする必要がある場合は、このオプションを選択します。

Connect をクリックすると、システムは自動的に次の操作を行います。

- 指定された IAM ロールを作成または設定します
- エージェントスペースの IAM アイデンティティセンターアプリケーションをセットアップします
- IAM アイデンティティセンターとエージェントスペースウェブアプリ間の信頼関係を確立します
- 安全なユーザーアクセスのために OAuth 2.0 認証フローを設定します

代替: 管理者アクセスの使用

IAM アイデンティティセンターを設定せずにエージェントスペースウェブアプリにすぐにアクセスする場合:

1. 管理者アクセスセクションで、管理者アクセスを提供する IAM ロール ARN を書き留めます (例: `arn:aws:iam::440491339484:role/service-role/DevOpsAgentRole-WebappAdmin-15ppoc42`)。
2. 青い管理者アクセスボタンをクリックして、IAM 認証でエージェントスペースウェブアプリを起動します。
3. この方法を使用するセッションは 30 分に制限されます

> **Note**
>
> 管理者アクセスは、初期設定と設定を目的としています。本番稼働用および継続的なオペレーションの場合は、IAM Identity Center 認証を設定します。

ユーザーとグループの追加

IAM Identity Center 認証を設定したら、特定のユーザーとグループに エージェントスペースウェブアプリへのアクセスを許可する必要があります。

ステップ 1: ユーザー管理にアクセスする

1. AWS DevOps エージェントコンソールで、エージェントスペースを選択します。
2. アクセスタブに移動する
3. ユーザーアクセスで、ユーザーとグループの管理をクリックします。

ステップ 2: ユーザーまたはグループを追加する

1. ユーザーまたはグループの追加を選択する
2. IAM Identity Center ディレクトリでユーザーまたはグループを検索する
3. 追加するユーザーまたはグループの横にあるチェックボックスをオンにします。
4. 追加をクリックしてアクセス権を付与する

選択したユーザーは、IAM アイデンティティセンターの認証情報を使用してエージェントスペースウェブアプリにアクセスできるようになりました。

外部 ID プロバイダーの使用

IAM アイデンティティセンターで外部 ID プロバイダー (Okta、Microsoft Entra ID、Ping Identity など) を使用している場合:

- ユーザーとグループは、外部 ID プロバイダーから IAM アイデンティティセンターに同期されます。
- エージェントスペースウェブアプリにユーザーとグループを追加すると、同期されたディレクトリから選択されます。
- ユーザー属性とグループメンバーシップは、外部 ID プロバイダーによって維持されます。
- ID プロバイダーの変更は、同期後に IAM アイデンティティセンターに自動的に反映されます。

ユーザーが エージェントスペースウェブアプリにアクセスする方法

エージェントスペースにユーザーを追加した後:

1. エージェントスペースのウェブアプリ URL を承認されたユーザーと共有する
2. ユーザーが URL に移動すると、IAM Identity Center のログインページにリダイレクトされます。

3. 認証情報を入力する (設定されている場合は MFA を完了する) と、エージェントスペースウェブアプリにリダイレクトされます。
4. セッションはデフォルトで 8 時間有効です (Identity Center 管理者が設定可能)

ユーザーアクセスの管理

ユーザーアクセスはいつでも更新できます。

ユーザーまたはグループの追加:

- 上記と同じ手順に従って、ユーザーまたはグループを追加します。

アクセスの削除:

1. ユーザーアクセスセクションで、削除するユーザーまたはグループを見つけます。
2. 名前の横にある削除ボタンをクリックします。
3. 削除を確認する

削除されたユーザーはすぐにアクセスできなくなりますが、アクティブなセッションは有効期限が切れるまで継続される場合があります。

セッション管理

エージェントスペースウェブアプリの IAM Identity Center セッションには、次の特性があります。

- デフォルトのセッション期間 – 8 時間
- セッションセキュリティ – 保護を強化するための HTTP 専用 Cookie
- 多要素認証 – IAM Identity Center で設定されている場合にサポートされます
- API 認証情報 – API コールに対して短時間 (15 分) SigV4 認証情報が発行され、自動的に更新されます。

セッション期間を設定するには:

1. IAM Identity Center コンソールに移動する
2. 設定 > 認証に進む

3. セッション期間で、希望する期間 (1 時間から 12 時間) を設定します。
4. [変更を保存] を選択します。

Identity Center の切断

1. エージェントスペースのコンソールで、右上のアクションをクリックし、IAM アイデンティティセンターから切断を選択します。
2. 確認ダイアログで確認する

外部 ID プロバイダー (IdP) 認証の設定

外部 ID プロバイダー (IdP) 認証を使用すると、組織は Okta や Microsoft Entra ID などの既存の OIDC 互換 ID プロバイダーを使用して、 AWS DevOps エージェントスペースウェブアプリケーションへのユーザーアクセスを管理できます。ユーザーは、 AWS IAM アイデンティティセンターを必要とせずに、IdP から直接企業認証情報でサインインします。

前提条件

外部 IdP 認証を設定する前に、以下を確認してください。

- OIDC 互換 ID プロバイダー (Okta または Microsoft Entra ID)
- ID プロバイダーへの管理者アクセス
- AWS DevOps エージェントコンソールにアクセスするための管理者権限
- エージェントスペースが設定済みまたは作成準備完了

仕組み

外部 IdP 認証を設定する場合:

- ユーザーがエージェントスペースウェブアプリ URL に移動する
- ID プロバイダーのログインページにリダイレクトされます。
- 企業の認証情報を使用して認証すると、ウェブアプリにリダイレクトされます。
- ウェブアプリケーションは、認証トークンをエージェントスペースにスコープされた有効期間の短い AWS 認証情報と交換します。

セッションは最大 8 時間有効です。認証情報は、ユーザーが再認証する必要なく、OIDC 更新トークンを使用して自動的に更新されます。

外部 IdP 認証の設定

ステップ 1: ID プロバイダーにアプリケーションを登録する

ID プロバイダーを選択し、対応するセットアップ手順に従います。

オプション A: Okta

1. Okta 管理コンソールで、アプリケーション > アプリケーションに移動し、アプリケーション統合の作成を選択します。
2. サインイン方法として OIDC - OpenID Connect を選択し、アプリケーションタイプとしてウェブアプリケーションを選択します。[次へ] を選択します。
3. アプリケーションのわかりやすい名前を設定する (例: `AWS DevOps Agent`)
4. 付与タイプで、以下がチェックされていることを確認します。
   - 認可コード (デフォルト)
   - 更新トークン — これはセッションの更新に必要です。有効になっていない場合、ユーザーはセッションを維持できません。

> **Note**
>
> Okta は、デフォルトで更新トークン許可タイプを有効にしません。明示的に有効にする必要があります。

1. サインインリダイレクト URIsは、現時点ではデフォルト値のままにしておきます。エージェントスペースの設定後に更新します。
2. 「割り当て」で、アクセス権を持つユーザーまたはグループを割り当てます。
3. [保存] を選択します。
4. アプリケーションの全般タブで、次の値を書き留めます。
   - クライアント ID
   - クライアントシークレット — コピーを選択してこの値を安全に保存します
5. Okta ドメインに注意してください。これは発行者 URL です (例: `https://dev-12345678.okta.com`)。

> **Note**
>
> サインオンタブで、発行者が Okta URL (動的ではない) に設定されていることを確認します。これにより、発行者の URL が安定します。

> **Note**
>
> 認可サーバーのクレームタブの ID トークンにグループクレームを追加しないでください。AWS DevOps エージェントは IdP のグループメンバーシップを使用しません。

オプション B: Microsoft Entra ID

1. Azure ポータルで、Microsoft Entra ID > アプリ登録 > 新規登録に移動します。
2. わかりやすい名前を設定する (例: `AWS DevOps Agent`)
3. サポートされているアカウントタイプで、組織に適したオプションを選択します (通常はこの組織ディレクトリのアカウントのみ)。
4. 現時点では、リダイレクト URI は空白のままにしておきます。登録を選択する
5. アプリケーションの概要ページで、次の値を書き留めます。
   - アプリケーション (クライアント) ID — エージェントスペースを設定するときにクライアント ID として使用されます
   - ディレクトリ (テナント) ID — 発行者 URL の構築に使用されます
6. 証明書とシークレット > 新しいクライアントシークレットに移動する
   - 説明と有効期限を設定する
   - シークレット値の追加とコピーをすぐに選択する — 再度表示されません
7. Entra ID の発行者 URL はこの形式に従います。をステップ 5 のディレクトリ (テナント) ID `{tenant-id}`に置き換えます。
   - `https://login.microsoftonline.com/{tenant-id}/v2.0`

> **Note**
>
> トークン設定 でグループオプションクレームを有効にしないでください。 AWS DevOps エージェントは IdP のグループメンバーシップを使用しません。

ステップ 2: IdP 認証でオペレーターアプリを有効にする

1. AWS DevOps エージェントコンソールで、エージェントスペースを選択します。
2. アクセスタブに移動する
3. ユーザーアクセスで、外部 ID プロバイダーを選択します。
4. 設定フォームで、以下を設定します。
   - ID プロバイダー — ID プロバイダー (Okta または Microsoft Entra ID) を選択します。
   - 発行者 URL — ID プロバイダーからの OIDC 発行者 URL
   - クライアント ID — 作成した OIDC アプリケーションのクライアント ID
   - クライアントシークレット — OIDC アプリケーションからのクライアントシークレット
5. ID プロバイダーアプリケーションロール名で、次の 3 つのオプションのいずれかを選択します。
   - 新しい DevOps エージェントロールの自動作成 (推奨) — 適切なアクセス許可を持つ新しいサービスロールを作成します
   - 既存のロールを割り当てる — 既に作成した既存の IAM ロールを使用します。
   - ポリシーテンプレートを使用して新しい DevOps エージェントロールを作成する — 提供された詳細を使用して、IAM コンソールで独自のロールを作成します。
6. フォームの下部に表示されるコールバック URL 警告アラートを確認します。この URL をコピーする — ユーザーがサインインする前に、ID プロバイダーの許可されたリダイレクト URIs に追加する必要があります。
7. [Connect (接続)] を選択します。

Connect を選択すると、コンソールに次の詳細を含む外部 ID プロバイダー設定が表示されます。

- プロバイダー — 選択した ID プロバイダー
- 発行者 URL — 設定された OIDC 発行者 URL
- クライアント ID — 設定されたクライアント ID
- IAM ロール ARN — ユーザーアクセスに使用される IAM ロール
- コールバック URL — ID プロバイダーでこの URL を許可されたリダイレクト URI として設定します。
- ログイン URL — この URL を使用して、ID プロバイダーを介してウェブアプリケーションにアクセスします。

ステップ 3: ID プロバイダーにコールバック URL を追加する

Okta

1. Okta 管理コンソールで、アプリケーションの全般タブに移動します。
2. ログイン で編集 を選択します。
3. コールバック URL をサインインリダイレクト URI として追加します。
   - `https://{agentSpaceId}.aidevops.global.app.aws/authorizer/idp/callback`
4. (オプション) ログインの開始 URI を設定して、Okta ダッシュボードから IdP 開始ログインを有効にします。
   - `https://{agentSpaceId}.aidevops.global.app.aws/authorizer/idp/login`
5. (推奨) サインアウトリダイレクト URI を追加して、ログアウト後にユーザーをウェブアプリにリダイレクトします。これを行わないと、ログアウト時にエラーページが表示されることがあります。
   - `https://{agentSpaceId}.aidevops.global.app.aws/authorizer/welcome`
6. [保存] を選択します。

Microsoft Entra ID

1. Azure ポータルで、アプリケーションの認証ページに移動します。
2. プラットフォーム設定で、プラットフォームの追加 > Web を選択します。
3. コールバック URL をリダイレクト URI として入力します。
   - `https://{agentSpaceId}.aidevops.global.app.aws/authorizer/idp/callback`
4. (オプション) サインアウトリダイレクト URI を追加して、ログアウト後にユーザーをウェブアプリにリダイレクトします。
   - `https://{agentSpaceId}.aidevops.global.app.aws/authorizer/welcome`
5. Configure を選択する

ステップ 4: 設定を確認する

1. コンソールに表示されるログイン URL に移動します。
   - `https://{agentSpaceId}.aidevops.global.app.aws/authorizer/idp/login`
2. ID プロバイダーのログインページにリダイレクトされます

3. 会社の認証情報を使用してサインインする
4. 認証に成功すると、エージェントスペースウェブアプリにリダイレクトされます。

IdP 設定の更新

クライアントシークレットは、切断せずにローテーションできます。

1. AWS DevOps エージェントコンソールで、エージェントスペースを選択します。
2. アクセスタブに移動する
3. 外部 ID プロバイダー設定で、クライアントシークレットのローテーションを選択します。
4. 新しいクライアントシークレットを入力する
5. [保存] を選択します。

他の IdP 設定フィールド (発行者 URL、クライアント ID、ID プロバイダーなど) を変更するには、既存の IdP を切断し、新しい IdP を設定する必要があります。

ユーザーが エージェントスペースウェブアプリにアクセスする方法

外部 IdP 認証を設定した後:

- エージェントスペースのウェブアプリ URL を承認されたユーザーと共有する
- ユーザーが URL に移動すると、ID プロバイダーのログインページにリダイレクトされます。
- 認証情報を入力する (IdP で設定されている場合は MFA を完了する) と、エージェントスペースウェブアプリにリダイレクトされます。
- セッションの自動更新 — 詳細については、「セッション管理」を参照してください。

セッション管理

エージェントスペースウェブアプリの外部 IdP セッションには、次の特性があります。

- セッション期間 — ブラウザセッションは最大 8 時間続きます。これは AWS DevOps エージェントでは設定できません。IdP のセッション有効期間が 8 時間を超える場合、ユーザーは認証情報を入力せずに次回のアクセス時に自動的に再認証される可能性があります。組織のセキュリティ要件に従って、IdP のセッションとトークンの有効期間を設定します。
- 認証情報の更新 — セッションは OIDC 更新トークンを使用して自動的に更新され、ユーザーが再認証する必要はありません。

- 多要素認証 — ID プロバイダーで設定されている場合にサポートされます。IdP はログイン中に MFA を処理します。 AWS DevOps エージェントで追加の設定は必要ありません

ログアウト動作

ユーザーがウェブアプリでログアウトをクリックすると、次のようになります。

1. すべてのセッション Cookie はすぐにクリアされます
2. ユーザーは ID プロバイダーの OIDC ログアウトエンドポイントにリダイレクトされ、SSO セッションを終了します。
3. サインアウトリダイレクト URI が設定されている場合、ユーザーはウェブアプリのウェルカムページにリダイレクトされます。

ユーザーアクセスの取り消し

ユーザーのアクセスをすぐに取り消すには、ID プロバイダーの管理ポータルでセッションを直接取り消すことができます。

- Okta — Okta 管理コンソールで、ディレクトリ > ユーザーに移動し、ユーザーを選択し、その他のアクション > ユーザーセッションのクリアを選択します。
- Microsoft Entra ID — Azure ポータルで、ユーザーに移動し、ユーザーを選択し、セッションの取り消しを選択します。

セキュリティに関する考慮事項

クライアントシークレットストレージ — セットアップ時に指定したクライアントシークレットは、エージェントスペースの作成時にカスタマーマネージド KMS キーを指定した場合、またはサービス所有キーを使用して暗号化されます。初期設定後に API レスポンスで返されたり、コンソールに表示されたりすることはありません。

クライアントシークレットのローテーション — Entra クライアントシークレットには設定可能な有効期限があります。 AWS DevOps エージェントコンソールのクライアントシークレットのローテーションオプションを使用して、有効期限が切れる前にシークレットをローテーションするようにリマインダーを設定します。シークレットの有効期限が切れると、ユーザーはローテーションされるまでログインできなくなります。

トークンの有効期間管理 — ID プロバイダーによって発行されたトークン (アクセストークン、更新トークン) の有効期間は、IdP の設定によって制御されます。IdP で適切なトークン有効期間を設定することをお勧めします。

- Okta — セキュリティ > API > 認可サーバー > アクセスポリシーでトークンの有効期間を設定する
- Microsoft Entra ID — トークンライフタイムポリシーを使用してトークンライフタイムを設定する

グループクレーム — ID プロバイダーのトークン設定でグループクレームを有効にしないでください。 AWS DevOps エージェントは現在、IdP のグループメンバーシップを使用しません。

ユーザー識別子 — AWS DevOps Agent はプロバイダー固有のクレームを使用してユーザーを一意に識別します。

- Okta — ID トークンからの subクレームを使用します
- Microsoft Entra ID — ID トークンからの oid (オブジェクト識別子) クレームを使用します

これらの識別子はイミュータブルであり、監査目的で CloudTrail ログに表示されます。

外部 IdP の切断

1. AWS DevOps エージェントコンソールで、エージェントスペースを選択します。
2. アクセスタブに移動する
3. ユーザーアクセスで、切断を選択します。
4. 確認ダイアログに記載されている影響を確認し、確認します。

切断すると、次のようになります。

- エージェントスペースから IdP 設定を削除する
- ユーザーが外部 ID プロバイダー経由でログインできないようにする
- IdP ユーザーアカウントに関連付けられた個々のチャットとアーティファクトの履歴を削除する

アクティブなユーザーセッションは、有効期限が切れるか、次の認証情報の更新が失敗するまで続行されます。

トラブルシューティング

- IdP へのリダイレクトが失敗する — 発行者 URL が IdP の OIDC 検出エンドポイントと一致することを確認します。Okta の場合は、Sign On タブで発行者が Okta URL (動的ではない) に設定されていることを確認します。Entra の場合は、 の形式を使用しますhttps://login.microsoftonline.com/{tenant-id}/v2.0。
- アクセス拒否またはポリシーエラー (Okta) — ユーザーまたはそのグループが割り当ての下でアプリケーションに割り当てられていることを確認します。サインオン > サインオンポリシールールを確認します。
- ログイン後の IdP 設定エラー — ID プロバイダーが更新トークンを返さなかった。offline_access スコープと更新トークンのグラントタイプが有効になっていることを確認します。
  - Okta — アプリケーションの全般タブに移動し、グラントタイプの下のトークンの更新チェックボックスを有効にします
  - Entra — API アクセス許可に移動し、 offline_accessが委任されたアクセス許可の下にリストされていることを確認します。
- 認証は成功しましたが、ウェブアプリケーションにエラーが表示されます — IdP のリダイレクト URI が、 AWS DevOps エージェントコンソールに表示されるコールバック URL と正確に一致することを確認します。
- 認証の失敗 — IdP でグループオプショングレームが有効になっている場合は、無効にします。AWS DevOps エージェントはグループクレームを使用しません。
- IdP 認証後にログインが失敗する — Entra の場合、アプリケーションマニフェストnullで requestedAccessTokenVersion が に設定されていないことを確認します。Okta の場合は、発行者 URL が正しいことを確認します。
- ログアウト (Okta) をクリックした後のエラーページ — ログアウト後にpost_logout_redirect_uriエラーが表示された場合は、Okta アプリケーションの全般タブにサインアウトリダイレクト URI https://{agentSpaceId}.aidevops.global.app.aws/authorizer/welcomeとして を追加します。
- ユーザーはログアウト後に ID プロバイダーページに留まる (Entra) — ログアウト後にユーザーをウェブアプリケーションにリダイレクトするには、Entra アプリケーションの認証ページでリダイレクト URI https://{agentSpaceId}.aidevops.global.app.aws/authorizer/welcomeとして を追加します。

AWS DevOps Agent の保管時の暗号化

AWS DevOps エージェントは、保管中のすべての顧客データを暗号化します。デフォルトでは、AWS DevOps エージェントは AWS 所有キーを使用して、追加料金なしでデータを自動的に暗号化します。 AWS 所有キーの使用を表示、管理、または監査することはできません。ただし、これらのキーを保護するためにアクションを実行する必要はありません。データは自動的に保護されます。

Key AWS Management Service (AWS KMS) で作成、所有、管理する対称カスタマーマネージドキーを使用してデータを暗号化することを選択できます。この暗号化レイヤーを完全に制御できるため、次のようなタスクを実行できます。

- キーポリシーの策定と維持
- キーポリシーの有効化と無効化
- キー暗号化マテリアルのローテーション
- タグを追加する
- キーエイリアスの作成
- 削除のためのキースケジューリング

詳細については、「 Key Management Service デベロッパーガイド」の「カスタマーマネージドキー」を参照してください。 AWS

> **Note**
>
> AWS DevOps エージェントは AWS 、所有キーを使用して保管時の暗号化を自動的に有効にし、顧客データを無償で保護します。カスタマーマネージドキーを使用する場合、標準の AWS KMS 料金が適用されます。料金の詳細については、AWS 「 Key Management Service の料金」を参照してください。

カスタマーマネージドキー

カスタマーマネージドキーは、作成、所有、管理する AWS アカウントの KMS キーです。キーポリシーの確立と維持など、これらの KMS キーを完全に制御できます。

カスタマーマネージドキーを設定すると、 AWS DevOps Agent はそれを使用して機密リソースデータを保護します。 AWS DevOps Agent AWS は Encryption SDK 階層キーリングでエンベロープ暗号化を使用します。KMS キーを使用してブランチキーを生成し、データを保護します。

カスタマーマネージドキーは、次のリソースを作成するときに指定できます。

- エージェントスペース — 調査、スキル、チャットに関連する DevOps エージェントウェブアプリから作成されたエージェントスペースの詳細とコンテンツを暗号化します。
- サービス — 保管中のサードパーティーのサービス認証情報を暗号化します。

AWS DevOps Agent でカスタマーマネージドキーを設定するには、次の手順に従います。

ステップ 1：カスタマーマネージドキーを作成する

対称カスタマーマネージドキーは、KMS コンソールまたは AWS KMS API AWS を使用して作成できます。キーは次の要件を満たしている必要があります。

プロパティ	要件
キーのタイプ	対称
キー仕様	`SYMMETRIC_DEFAULT`
キーの用途	`ENCRYPT_DECRYPT`

> **Note**
>
> AWS DevOps Agent は、キー仕様とキー`ENCRYPT_DECRYPT`使用法を持つ対称暗号化 KMS `SYMMETRIC_DEFAULT` キーのみをサポートします。マルチリージョンキーと非対称キーは現在サポートされていません。

詳細については、AWS 「 Key Management Service デベロッパーガイド」の「対称カスタマーマネージドキーの作成」を参照してください。

ステップ 2: キーポリシーを設定する

キーポリシーは、カスタマーマネージドキーへのアクセスを制御します。すべてのカスタマーマネージドキーには、キーポリシーが 1 つだけ必要です。このポリシーには、そのキーを使用できるユーザーとその使用方法を決定するステートメントが含まれています。

キーポリシーは、呼び出し元のプリンシパル (IAM ID) と AWS DevOps エージェントサービスの両方にアクセス許可を付与する必要があります。 AWS DevOps エージェントは、次の 2 つの認証情報セットを使用してキーにアクセスします。

1. 発信者認証情報 — キー検証、リソース作成時の暗号化、発信者に直接応答を返す API コールなど、すべての同期オペレーションに使用されます。
2. AWS DevOps エージェントサービスプリンシパル — 運用調査、インシデント分析、イベント相関、根本原因分析の生成など、バックグラウンドで実行される非同期オペレーションに使用されます。

次の表に、必要な KMS アクションを示します。

KMS アクション	説明
`kms:DescribeKey`	リソース作成時にキー設定を検証する
`kms:GenerateDataKey`	エンベロープ暗号化用のデータ暗号化キーを生成する
`kms:Decrypt`	データの復号化
`kms:Encrypt`	[データの暗号化]
`kms:ReEncrypt`	同じキーまたは異なるキーでデータを再暗号化する

AWS DevOps Agent は、ドライランオペレーションを使用して、設定時にこれらのアクセス許可をすべて検証します。アクセス許可がない場合、リクエストは例外で失敗します。

以下は、キーポリシーの例です。プレースホルダーの値を独自の値に置き換えます。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCallerAccessViaService",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/DevOpsAgentUserRole"
```

```
      },
      "Action": [
        "kms:DescribeKey",
        "kms:GenerateDataKey*",
        "kms:Decrypt",
        "kms:Encrypt",
        "kms:ReEncrypt*"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:ViaService": "aidevops.us-east-1.amazonaws.com"
        }
      }
    },
    {
      "Sid": "AllowDevOpsAgentServiceDescribeKeyAccess",
      "Effect": "Allow",
      "Principal": {
        "Service": "aidevops.amazonaws.com"
      },
      "Action": [
        "kms:DescribeKey"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowDevOpsAgentAccessForAgentSpace",
      "Effect": "Allow",
      "Principal": {
        "Service": "aidevops.amazonaws.com"
      },
      "Action": [
        "kms:GenerateDataKey*",
        "kms:Decrypt",
        "kms:Encrypt",
        "kms:ReEncrypt*"
      ],
      "Resource": "*",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:aidevops:us-east-1:111122223333:agentspace/*"
        },
        "StringLike": {
```

```
          "kms:EncryptionContext:aws-crypto-ec:aws:aidevops:arn": "arn:aws:aidevops:us-
east-1:111122223333:agentspace/*"
        }
      }
    },
    {
      "Sid": "AllowDevOpsAgentAccessForService",
      "Effect": "Allow",
      "Principal": {
        "Service": "aidevops.amazonaws.com"
      },
      "Action": [
        "kms:GenerateDataKey*",
        "kms:Decrypt",
        "kms:Encrypt",
        "kms:ReEncrypt*"
      ],
      "Resource": "*",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:aidevops:us-east-1:111122223333:service/*"
        },
        "StringLike": {
          "kms:EncryptionContext:aws-crypto-ec:aws:aidevops:arn": "arn:aws:aidevops:us-
east-1:111122223333:service/*"
        }
      }
    }
  ]
}
```

ポリシーには、次のステートメントが含まれています。

- AllowKeyAdministration — アカウントのルートにキーへの完全な管理アクセスを付与します。を AWS アカウント ID 111122223333に置き換えます。
- AllowCallerAccessViaService — すべての同期 AWS DevOps エージェントオペレーションに必要な KMS アクセス許可を IAM プリンシパルに付与します。これには、リソース作成時のキー検証、および発信者に直接応答を返す API コールの暗号化および復号オペレーションが含まれます。このkms:ViaService条件により、 AWS DevOps エージェントサービスを介してのみキーを使用できます。を AWS アカウント ID 111122223333に、 を AWS リージョンus-east-1に置き換えます。

- AllowDevOpsAgentServiceAccessForAgentSpace / AllowDevOpsAgentServiceAccessForService — 非同期オペレーションに必要な KMS アクセス許可を`aidevops.amazonaws.com`サービスプリンシパルに付与します。 AWS DevOps Agent は、このサービスプリンシパルを使用して、運用調査、インシデントの分析、サービス間のイベントの関連付け、根本原因分析の生成などのバックグラウンドオペレーションを実行するときにデータを暗号化および復号します。このアクセスがないと、 AWS DevOps Agent はユーザーに代わって調査を実行するために必要な暗号化されたデータを読み取ることができません。`aws:SourceArn` 条件は、 AWS DevOps エージェントリソースから送信されるリクエストへのアクセスを制限し、`kms:EncryptionContext`条件は暗号化コンテキストがリソース ARNs と一致するようにします。を AWS アカウント ID `111122223333`に、を AWS リージョン`us-east-1`に置き換えます。

キーポリシーの詳細については、「 Key Management Service デベロッパーガイド」の「KMS AWS のキーポリシー」を参照してください。 AWS

ステップ 3: リソースを作成するときにキーを指定する

キーを作成してキーポリシーを設定したら、 AWS DevOps エージェントリソースの作成時にキーを指定できます。

コンソール

コンソールでエージェントスペースを作成するときにカスタマーマネージドキーを設定するには:

1. AWS DevOps エージェントコンソールを開きます。
2. エージェントスペースの作成またはサービスの登録を選択します。
3. エージェントスペースの詳細 (名前、説明、IAM ロール) を入力します。
4. 詳細設定セクションを展開します。
5. 暗号化キータイプで、カスタマーマネージドキーを選択します。
6. ドロップダウンリストから KMS キーを選択するか、KMS キー ARN を入力します。
7. キーポリシー展開可能セクションに表示されるキーポリシーを確認します。このポリシーが KMS キーにアタッチされていることを確認します。コピーボタンを使用してポリシーをコピーできます。
8. 残りの設定を完了し、作成を選択します。

> **Note**
>
> ドロップダウンリストに KMS キーが表示されない場合は、キーがステップ 1 の要件を満たし、 `kms:ListKeys` および アクセス`kms:DescribeKey`許可があることを確認します。

API

カスタマーマネージドキーを使用したエージェントスペースの作成

エージェントスペースを作成するときに `kmsKeyArn`パラメータを指定します。値は完全な KMS キー ARN である必要があります。

```
{
  "name": "my-agent-space",
  "description": "An encrypted agent space",
  "kmsKeyArn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
}
```

カスタマーマネージドキーでサービスを登録する

サービスを登録するときに `kmsKeyArn`パラメータを指定します。値は完全な KMS キー ARN である必要があります。このパラメータは、Dynatrace、ServiceNow、PagerDuty、GitLab、GitHub、および MCP サーバーを含むすべてのサービスタイプでサポートされています。

```
{
  "service": "dynatrace",
  "kmsKeyArn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "serviceDetails": { ... }
}
```

> **Note**
>
> リソースの作成時にカスタマーマネージドキーを指定する必要があります。既存のリソースのカスタマーマネージドキーを追加または変更することはできません。

AWS DevOps エージェント暗号化コンテキスト

暗号化コンテキストは、データに関する追加のコンテキスト情報を含むシークレット以外のキーと値のペアのセットです。 AWS KMS は、追加の認証データとして暗号化コンテキストを使用して、認証された暗号化をサポートします。データを暗号化するリクエストに暗号化コンテキストを含めると、 AWS KMS は暗号化コンテキストを暗号化されたデータにバインドします。データを復号するには、リクエストに同じ暗号化コンテキストを含める必要があります。

AWS DevOps エージェントは、すべての暗号化オペレーションで次の暗号化コンテキストを使用します。

```
{
  "aws-crypto-ec:aws:aidevops:arn": "arn:aws:aidevops:{region}:{accountId}:
{resourceType}/{resourceId}"
}
```

暗号化コンテキスト値は、暗号化されている AWS DevOps エージェントリソースの ARN です。この暗号化コンテキストは、キーポリシー条件と AWS CloudTrail ログで使用して、キーの使用方法を監査できます。

キー管理

KMS キーの削除を無効化またはスケジュールすると、 AWS DevOps Agent はデータを復号化できません。これにより、暗号化されたデータを読み取るオペレーションで`AccessDeniedException`エラーが発生します。

> **Important**
>
> カスタマーマネージドキーを使用する場合は、キーとそのアクセス許可を管理する責任があります。キーが無効または削除された場合、または AWS DevOps Agent がキーを使用するアクセス許可を失った場合、暗号化されたデータにアクセスできなくなります。

次の表に、一般的な障害シナリオを示します。

Action	Impact
キーポリシーのアクセス許可が取り消されました	`AccessDeniedException` 暗号化および復号オペレーションでの

Action	Impact
KMS キーが無効になっている	`DisabledException` 暗号化および復号オペレーションでの
KMS キーの削除が予定されています	`KMSInvalidStateException` 暗号化および復号オペレーションでの
KMS キーが削除されます	永続的なデータ損失 — 暗号化されたデータは復元できません

キーを無効化または削除する前に:

1. アクティブな AWS DevOps エージェントリソースがキーに依存していないことを確認します。
2. 削除をスケジュールする前に、まずキーを無効にして影響をテストすることを検討してください。
3. AWS KMS は、キーを削除する前の最小待機期間を適用し、必要に応じてキャンセルする時間を確保します。

> 注:: AWS DevOps エージェントは、新しいキーでデータを自動的に再暗号化しません。新しいカスタマーマネージドキーにローテーションする必要がある場合は、新しいキーを使用して新しいリソースを作成する必要があります。

暗号化キーのモニタリング

AWS DevOps Agent でカスタマーマネージドキーを使用する場合、AWS CloudTrail を使用して、AWS DevOps Agent が KMS AWS に送信するリクエストを追跡できます。

CloudTrail イベントは、次の方法でフィルタリングできます。

- イベントソース — `kms.amazonaws.com`
- 暗号化コンテキストキー — `aws-crypto-ec:aws:aidevops:arn`
- キー ARN — リクエストパラメータのカスタマーマネージドキー ARN

詳細については、AWS 「 Key Management Service デベロッパーガイドAWS 」の AWS CloudTrail を使用した KMS API コールのログ記録」を参照してください。

VPC エンドポイント (AWS PrivateLink)

AWS PrivateLink を使用して、VPC と AWS DevOps エージェント間のプライベート接続を作成できます。インターネットゲートウェイ、NAT デバイス、VPN 接続、または Direct Connect 接続を使用せずに、VPC 内にあるかのように AWS DevOps エージェントにアクセスできます。VPC 内のインスタンスは、 AWS DevOps エージェントにアクセスするためにパブリック IP アドレスを必要としません。

このプライベート接続を確立するには、 AWS PrivateLink を搭載したインターフェイスエンドポイントを作成します。インターフェイスエンドポイントに対して有効にする各サブネットにエンドポイントネットワークインターフェイスを作成します。これらは、 AWS DevOps エージェント宛てのトラフィックのエントリポイントとして機能するリクエスタ管理のネットワークインターフェイスです。

詳細については、「 AWS PrivateLink ガイド」の「PrivateLink 経由で AWS サービスにアクセスする」を参照してください。 AWS PrivateLink

AWS DevOps エージェント VPC エンドポイントに関する考慮事項

AWS DevOps Agent のインターフェイスエンドポイントを設定する前に、AWS 「 PrivateLink ガイド」の「考慮事項」を参照してください。

AWS DevOps Agent は、次の VPC エンドポイントを介した API コールをサポートしています。

Category	エンドポイントサフィックス
AWS DevOps エージェントコントロールプレーン API アクション	`aidevops`
AWS DevOps エージェントランタイムオペレーション	`aidevops-dataplane`
AWS DevOps エージェントウェブフックイベント	`event-ai`

AWS DevOps Agent のインターフェイスエンドポイントを作成する

Amazon VPC コンソールまたは コマンドラインインターフェイス (AWS CLI) を使用して、 AWS DevOps Agent AWS のインターフェイスエンドポイントを作成できます。詳細については、AWS 「 PrivateLink ガイド」の「インターフェイスエンドポイントの作成」を参照してください。

次のサービス名を使用して、 AWS DevOps Agent のインターフェイスエンドポイントを作成します。

- com.amazonaws.{region}.aidevops
- com.amazonaws.{region}.aidevops-dataplane
- com.amazonaws.{region}.event-ai

エンドポイントを作成後に、プライベート DNS ホスト名を有効にするオプションがあります。VPC エンドポイントの作成時に VPC コンソールで [プライベート DNS 名を有効にする] を選択して、この設定名を有効にします。

インターフェイスエンドポイントのプライベート DNS を有効にすると、デフォルトのリージョン DNS 名を使用して AWS DevOps エージェントに API リクエストを行うことができます。次の例は、デフォルトのリージョン DNS 名の形式を示しています。

- aidevops.{region}.api.aws
- aidevops-dataplane.{region}.amazonaws.com
- event-ai.{region}.api.aws

インターフェイスエンドポイントのエンドポイントポリシーを作成する

エンドポイントポリシーは、インターフェイスエンドポイントにアタッチできる IAM リソースです。デフォルトのエンドポイントポリシーでは、インターフェイスエンドポイントを介して AWS DevOps エージェントへのフルアクセスを許可します。VPC から AWS DevOps エージェントに許可されるアクセスを制御するには、カスタムエンドポイントポリシーをインターフェイスエンドポイントにアタッチします。

エンドポイントポリシーは以下の情報を指定します。

- アクションを実行できるプリンシパル (AWS アカウント、IAM ユーザー、IAM ロール) 。
- 実行可能なアクション。

- このアクションを実行できるリソース。

詳細については、AWS 「 PrivateLink ガイド」の「エンドポイントポリシーを使用してサービスへのアクセスを制御する」を参照してください。

クォータ

AWS DevOps エージェントクォータには、エージェントスペースの数、同時調査などが含まれます。一部のクォータについては引き上げをリクエストできますが、引き上げできないクォータもあります。これらの引き上げはすぐには付与されないため、引き上げが有効になるまでに数時間から数日かかる場合があります。特に明記していない限り、クォータはリージョン固有です。

次の表に、 AWS DevOps Agent のクォータを示します。

名前	デフォルト	引き上げ可能	説明
リージョンごとのアカウントあたりのエージェントスペース	10	[Yes (はい)]	各 AWS リージョンでアカウントごとに作成できるエージェントスペースの最大数。
エージェントスペースあたりの同時調査	3	はい	1 つのエージェントスペースで同時に実行できるインシデント解決調査の最大数。
エージェントスペースあたりの同時評価	1	いいえ	1 つのエージェントスペースで同時に実行できるインシデント防止評価の最大数。
エージェントスペースあたりの同時オンデマンド呼び出し	10	[Yes (はい)]	1 つのエージェントスペースで同時に実行できるオンデマンド DevOps 呼び出しの最大数。

クォータ引き上げのリクエスト

次のいずれかのオプションを使用して、クォータの引き上げをリクエストできます。

- AWS マネジメントコンソールから – Service Quotas コンソールを開きます。ナビゲーションペインで、AWS [サービス] を選択します。DevOps エージェントを選択し、クォータを選択し、指示に従ってクォータの引き上げをリクエストします。詳細については、「Service Quotas ユーザーガイド」の「クォータの引き上げのリクエスト」を参照してください。
- AWS CLI から – request-service-quota-increase AWS CLI コマンドを使用します。詳細については、「Service Quotasユーザーガイド」の「クォータ引き上げのリクエスト」を参照してください。

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。