aws

Panduan Pengguna

Application Auto Scaling

Application Auto Scaling: Panduan Pengguna

Table of Contents

Apa yang dimaksud dengan Application Auto Scaling?

Application Auto Scaling adalah layanan web untuk pengembang dan administrator sistem yang membutuhkan solusi untuk secara otomatis menskalakan sumber daya mereka yang dapat diskalakan untuk layanan individual di luar AWS Amazon Auto Scaling. EC2 Dengan Application Auto Scaling, Anda dapat mengonfigurasi penskalaan otomatis untuk sumber daya berikut: : AWS

- WorkSpaces Armada aplikasi
- Replika Aurora
- Klasifikasi dokumen dan titik akhir pengenal entitas Amazon Comprehend
- Tabel DynamoDB dan indeks sekunder global
- Layanan-layanan Amazon ECS
- ElastiCache kelompok replikasi (Redis OSS dan Valkey) dan cluster Memcached
- Kluster Amazon EMR
- Tabel Amazon Keyspaces (untuk Apache Cassandra)
- Konkurensi terprovisi fungsi Lambda
- Penyimpanan broker Amazon Managed Streaming untuk Apache Kafka (MSK)
- Cluster Amazon Neptunus
- SageMaker Varian titik akhir AI
- SageMaker Komponen inferensi AI
- SageMaker Konkurensi yang disediakan tanpa server AI
- Permintaan Armada Spot
- Kolam Amazon WorkSpaces
- Sumber daya kustom yang disediakan oleh aplikasi atau layanan Anda sendiri. Untuk informasi lebih lanjut, lihat GitHub repositori.

Untuk melihat ketersediaan regional untuk salah satu AWS layanan yang tercantum di atas, lihat tabel Wilayah tabel .

Untuk informasi tentang penskalaan armada EC2 instans Amazon menggunakan grup Auto Scaling, lihat Panduan Pengguna Auto EC2 Scaling Amazon.

Fitur Application Auto Scaling

Application Auto Scaling memungkinkan Anda untuk secara otomatis memperbesar skala sumber daya Anda sesuai dengan kondisi yang Anda tentukan.

- Penskalaan pelacakan target — Skala sumber daya berdasarkan nilai target untuk CloudWatch metrik tertentu.
- Penskalaan langkah — Skala sumber daya berdasarkan serangkaian penyesuaian penskalaan yang bervariasi berdasarkan ukuran pelanggaran alarm.
- Penskalaan terjadwal — Skala sumber daya hanya satu kali atau pada jadwal berulang.
- Penskalaan prediktif — Skala sumber daya secara proaktif untuk mencocokkan beban yang diantisipasi berdasarkan data historis.

Bekerja dengan Application Auto Scaling

Anda dapat mengonfigurasi penskalaan menggunakan antarmuka berikut tergantung pada sumber daya yang Anda skalakan:

- Konsol Manajemen AWS— Menyediakan antarmuka web yang dapat Anda gunakan untuk mengkonfigurasi penskalaan. Mendaftar untuk AWS akun dan masuk ke Konsol Manajemen AWS. Kemudian, buka konsol layanan untuk salah satu sumber daya yang tercantum dalam pendahuluan. Misalnya, untuk menskalakan fungsi Lambda, buka. AWS Lambda console Pastikan Anda membuka konsol AWS Region sama dengan sumber daya yang ingin Anda gunakan.

  > **Note**
  >
  > Akses konsol tidak tersedia untuk semua sumber daya. Untuk informasi selengkapnya, lihat Layanan AWS yang dapat Anda gunakan dengan Application Auto Scaling.

- AWS Command Line Interface (AWS CLI) — Menyediakan perintah untuk serangkaian luas Layanan AWS, dan didukung pada Windows, macOS, dan Linux. Untuk memulai, lihat AWS Command Line Interface. Untuk daftar perintah, lihat application-autoscaling di Command Reference.AWS CLI
- AWS Tools for Windows PowerShell— Menyediakan perintah untuk serangkaian AWS produk yang luas bagi mereka yang membuat skrip di PowerShell lingkungan. Untuk memulai, lihat AWS Tools

for PowerShell Panduan Pengguna. Untuk informasi lebih lanjut, lihat AWS Tools for PowerShell Referensi Cmdlet.

- AWS SDKsMenyediakan operasi API khusus bahasa dan menangani banyak detail koneksi, seperti menghitung tanda tangan, menangani percobaan ulang permintaan, dan menangani kesalahan. Untuk informasi selengkapnya, lihat Alat untuk Dibangun AWS.
- HTTPS API - Menyediakan tindakan API tingkat rendah yang Anda panggil menggunakan permintaan HTTPS. Untuk informasi selengkapnya, lihat Referensi API Application Auto Scaling.
- CloudFormation— Mendukung konfigurasi penskalaan menggunakan template. CloudFormation Untuk informasi selengkapnya, lihat Konfigurasikan sumber daya Application Auto Scaling menggunakan AWS CloudFormation.

Untuk terhubung secara terprogram ke sebuah Layanan AWS, Anda menggunakan endpoint. .

Konsep Application Auto Scaling

Topik ini menjelaskan konsep-konsep kunci untuk membantu Anda mempelajari Application Auto Scaling dan mulai menggunakannya.

Target yang dapat diskalakan

Entitas yang Anda buat untuk menentukan sumber daya yang ingin Anda skalakan. Setiap target yang dapat diskalakan diidentifikasi secara unik oleh namespace layanan, ID sumber daya, dan dimensi yang dapat diskalakan, yang mewakili beberapa dimensi kapasitas layanan yang mendasarinya. Misalnya, layanan Amazon ECS mendukung penskalaan otomatis jumlah tugasnya, tabel DynamoDB mendukung penskalaan otomatis kapasitas baca dan tulis tabel dan indeks sekunder globalnya, dan klaster Aurora mendukung penskalaan jumlah replika.

> **Tip**
>
> Setiap target yang dapat diskalakan juga memiliki kapasitas minimum dan maksimum. Kebijakan penskalaan tidak akan pernah lebih tinggi atau lebih rendah dari kisaran minimum-maksimum. Anda dapat membuat out-of-band perubahan langsung ke sumber daya dasar yang berada di luar rentang ini, yang tidak diketahui Application Auto Scaling. Namun, kapan saja kebijakan penskalaan dipanggil atau `RegisterScalableTarget` API dipanggil, Application Auto Scaling mengambil kapasitas saat ini dan membandingkannya dengan kapasitas minimum dan maksimum.

Jika berada di luar kisaran minimum-maksimum, maka kapasitas diperbarui untuk memenuhi minimum dan maksimum yang ditetapkan.

Penurunan skala

Ketika Application Auto Scaling secara otomatis mengurangi kapasitas untuk target yang dapat diskalakan, target yang dapat diskalakan akan masuk. Ketika kebijakan penskalaan ditetapkan, mereka tidak dapat menskalakan target yang dapat diskalakan lebih rendah dari kapasitas minimumnya.

Menskalakan ke luar

Ketika Application Auto Scaling secara otomatis meningkatkan kapasitas untuk target yang dapat diskalakan, target yang dapat diskalakan akan keluar. Ketika kebijakan penskalaan ditetapkan, mereka tidak dapat mengukur target yang dapat diskalakan lebih tinggi dari kapasitas maksimumnya.

Kebijakan penskalaan

Kebijakan penskalaan menginstruksikan Application Auto Scaling untuk melacak metrik tertentu. CloudWatch Kemudian, ini menentukan tindakan penskalaan apa yang harus diambil ketika metrik lebih tinggi atau lebih rendah dari nilai ambang tertentu. Misalnya, Anda mungkin ingin memperkecil skala jika penggunaan CPU di seluruh klaster Anda mulai meningkat, dan menskalakan saat CPU turun lagi.

Metrik yang digunakan untuk penskalaan otomatis diterbitkan oleh layanan target, tetapi Anda juga dapat mempublikasikan metrik Anda sendiri CloudWatch dan kemudian menggunakannya dengan kebijakan penskalaan.

Periode cooldown antara aktivitas penskalaan memungkinkan sumber daya stabil sebelum aktivitas penskalaan lainnya dimulai. Application Auto Scaling terus mengevaluasi metrik selama periode cooldown. Ketika periode cooldown berakhir, kebijakan penskalaan memulai aktivitas penskalaan lain jika diperlukan. Sementara periode cooldown berlaku, jika skala yang lebih besar diperlukan berdasarkan nilai metrik saat ini, kebijakan penskalaan segera keluar.

Tindakan terjadwal

Tindakan terjadwal secara otomatis menskalakan sumber daya pada tanggal dan waktu tertentu. Mereka bekerja dengan memodifikasi kapasitas minimum dan maksimum untuk target yang dapat diskalakan, dan oleh karena itu dapat digunakan untuk skala masuk dan keluar sesuai jadwal

dengan menetapkan kapasitas minimum tinggi atau kapasitas maksimum rendah. Misalnya, Anda dapat menggunakan tindakan terjadwal untuk menskalakan aplikasi yang tidak mengkonsumsi sumber daya pada akhir pekan dengan mengurangi kapasitas pada hari Jumat dan meningkatkan kapasitas pada hari Senin berikutnya.

Anda juga dapat menggunakan tindakan terjadwal untuk mengoptimalkan nilai minimum dan maksimum dari waktu ke waktu untuk beradaptasi dengan situasi di mana lalu lintas yang lebih tinggi dari normal diharapkan, misalnya, kampanye pemasaran atau fluktuasi musiman. Melakukan hal ini dapat membantu Anda meningkatkan kinerja untuk saat-saat ketika Anda perlu skala lebih tinggi untuk peningkatan penggunaan, dan mengurangi biaya pada saat Anda menggunakan lebih sedikit sumber daya.

Pelajari selengkapnya

Layanan AWS yang dapat Anda gunakan dengan Application Auto Scaling— Bagian ini memperkenalkan Anda pada layanan yang dapat Anda skalakan dan membantu Anda mengatur penskalaan otomatis dengan mendaftarkan target yang dapat diskalakan. Ini juga menjelaskan setiap peran terkait layanan IAM yang dibuat Application Auto Scaling untuk mengakses sumber daya dalam layanan target.

Kebijakan penskalaan pelacakan target untuk Application Auto Scaling- Salah satu fitur utama Application Auto Scaling adalah kebijakan penskalaan pelacakan target. Pelajari cara kebijakan pelacakan target secara otomatis menyesuaikan kapasitas yang diinginkan untuk menjaga pemanfaatan pada tingkat konstan berdasarkan metrik dan nilai target yang dikonfigurasi. Misalnya, Anda dapat mengonfigurasi pelacakan target untuk menjaga penggunaan CPU rata-rata untuk Armada Spot Anda sebesar 50 persen. Application Auto Scaling kemudian meluncurkan atau menghentikan EC2 instance yang diperlukan untuk menjaga pemanfaatan CPU agregat di semua server sebesar 50 persen.

Layanan AWS yang dapat Anda gunakan dengan Application Auto Scaling

Application Auto Scaling terintegrasi dengan AWS layanan lain sehingga Anda dapat menambahkan kemampuan penskalaan untuk memenuhi permintaan aplikasi Anda. Penskalaan otomatis adalah fitur opsional dari layanan yang dinonaktifkan secara default di hampir semua kasus.

Tabel berikut mencantumkan AWS layanan yang dapat Anda gunakan dengan Application Auto Scaling, termasuk informasi tentang metode yang didukung untuk mengonfigurasi penskalaan otomatis. Anda juga dapat menggunakan Application Auto Scaling dengan sumber daya khusus.

- Akses konsol — Anda dapat mengonfigurasi AWS layanan yang kompatibel untuk memulai penskalaan otomatis dengan mengonfigurasi kebijakan penskalaan di konsol layanan target.
- Akses CLI - Anda dapat mengonfigurasi AWS layanan yang kompatibel untuk memulai penskalaan otomatis menggunakan. AWS CLI
- Akses SDK — Anda dapat mengonfigurasi AWS layanan yang kompatibel untuk memulai penskalaan otomatis menggunakan. AWS SDKs
- CloudFormation akses - Anda dapat mengonfigurasi AWS layanan yang kompatibel untuk memulai penskalaan otomatis menggunakan template CloudFormation tumpukan. Untuk informasi selengkapnya, lihat Konfigurasikan sumber daya Application Auto Scaling menggunakan AWS CloudFormation.

AWS layanan	Akses konsol[1]	Akses CLI	Akses SDK	CloudFormation akses
WorkSpaces Aplikasi	Ya	Ya	Ya	Ya
Aurora	Ya	Ya	Ya	Ya

AWS layanan	Akses konsol[1]	Akses CLI	Akses SDK	CloudFormation akses
Amazon Comprehend	Tida	Ya	Ya	Ya
Amazon DynamoDB	Ya	Ya	Ya	Ya
Amazon ECS	Ya	Ya	Ya	Ya
Amazon ElastiCache	Ya	Ya	Ya	Ya
Amazon EMR	Ya	Ya	Ya	Ya
Keyspaces Amazon	Ya	Ya	Ya	Ya
Lambda	Tida	Ya	Ya	Ya
Amazon MSK	Ya	Ya	Ya	Ya

AWS layanan	Akses konsol[1]	Akses CLI	Akses SDK	CloudFormation akses
Amazon Neptune	Tida	Ya	Ya	Ya
SageMaker AI	Ya	Ya	Ya	Ya
Armada Spot	Ya	Ya	Ya	Ya
WorkSpaces	Ya	Ya	Ya	Ya
Sumber daya khusus	Tida	Ya	Ya	Ya

[1] Akses konsol untuk mengonfigurasi kebijakan penskalaan. Sebagian besar layanan tidak mendukung konfigurasi penskalaan terjadwal dari konsol. Saat ini, hanya WorkSpaces Aplikasi Amazon ElastiCache, dan Armada Spot yang menyediakan akses konsol untuk penskalaan terjadwal.

WorkSpaces Aplikasi Amazon dan Application Auto Scaling

Anda dapat menskalakan armada WorkSpaces Aplikasi menggunakan kebijakan penskalaan pelacakan target, kebijakan penskalaan langkah, dan penskalaan terjadwal.

Gunakan informasi berikut untuk membantu Anda mengintegrasikan WorkSpaces Aplikasi dengan Application Auto Scaling.

Peran terkait layanan yang dibuat untuk Aplikasi WorkSpaces

Peran terkait layanan berikut dibuat secara otomatis di Akun AWS saat mendaftarkan sumber daya WorkSpaces Aplikasi sebagai target yang dapat diskalakan dengan Application Auto Scaling. Peran ini memungkinkan Application Auto Scaling untuk melakukan operasi yang didukung dalam akun Anda. Untuk informasi selengkapnya, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_AppStreamFleet`

Prinsipal layanan yang digunakan oleh peran terkait layanan

Peran terkait layanan di bagian sebelumnya hanya dapat diasumsikan oleh kepala layanan yang diotorisasi oleh hubungan kepercayaan yang ditentukan untuk peran tersebut. Peran terkait layanan yang digunakan oleh Application Auto Scaling memberikan akses ke prinsipal layanan berikut:

- `appstream.application-autoscaling.amazonaws.com`

Mendaftarkan armada WorkSpaces Aplikasi sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk armada Aplikasi. WorkSpaces Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling. Target yang dapat diskalakan diidentifikasi secara unik dengan kombinasi ID sumber daya, dimensi yang dapat diskalakan, dan namespace.

Jika Anda mengonfigurasi penskalaan otomatis menggunakan konsol WorkSpaces Aplikasi, maka WorkSpaces Aplikasi secara otomatis mendaftarkan target yang dapat diskalakan untuk Anda.

Jika Anda ingin mengonfigurasi penskalaan otomatis menggunakan AWS CLI atau salah AWS SDKs satu, Anda dapat menggunakan opsi berikut:

- AWS CLI:

  Panggil register-scalable-targetperintah untuk armada WorkSpaces Aplikasi. Contoh berikut mencatat kapasitas yang diinginkan dari armada yang disebut`sample-fleet`, dengan kapasitas minimum satu instance armada dan kapasitas maksimum lima instance armada.

```
aws application-autoscaling register-scalable-target \
   --service-namespace appstream \
   --scalable-dimension appstream:fleet:DesiredCapacity \
   --resource-id fleet/sample-fleet \
   --min-capacity 1 \
   --max-capacity 5
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Sumber daya terkait

Untuk informasi selengkapnya, lihat Auto Scaling Armada untuk WorkSpaces Aplikasi Amazon di Panduan Administrasi WorkSpaces Aplikasi Amazon.

Amazon Aurora dan Application Auto Scaling

Anda dapat menskalakan klaster DB Aurora menggunakan kebijakan penskalaan pelacakan target, kebijakan penskalaan langkah, dan penskalaan terjadwal.

Gunakan informasi berikut untuk membantu Anda mengintegrasikan Aurora dengan Application Auto Scaling.

Peran terkait layanan yang dibuat untuk Aurora

Peran terkait layanan berikut dibuat secara otomatis di Anda Akun AWS saat mendaftarkan sumber daya Aurora sebagai target yang dapat diskalakan dengan Application Auto Scaling. Peran ini memungkinkan Application Auto Scaling untuk melakukan operasi yang didukung dalam akun Anda.

Untuk informasi selengkapnya, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_RDSCluster`

Prinsipal layanan yang digunakan oleh peran terkait layanan

Peran terkait layanan di bagian sebelumnya hanya dapat diasumsikan oleh kepala layanan yang diotorisasi oleh hubungan kepercayaan yang ditentukan untuk peran tersebut. Peran terkait layanan yang digunakan oleh Application Auto Scaling memberikan akses ke prinsipal layanan berikut:

- `rds.application-autoscaling.amazonaws.com`

Mendaftarkan cluster Aurora DB sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk klaster Aurora. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling. Target yang dapat diskalakan diidentifikasi secara unik dengan kombinasi ID sumber daya, dimensi yang dapat diskalakan, dan namespace.

Jika Anda mengonfigurasi penskalaan otomatis menggunakan konsol Aurora, maka Aurora secara otomatis mendaftarkan target yang dapat diskalakan untuk Anda.

Jika Anda ingin mengonfigurasi penskalaan otomatis menggunakan AWS CLI atau salah AWS SDKs satu, Anda dapat menggunakan opsi berikut:

- AWS CLI:

  Panggil register-scalable-targetperintah untuk cluster Aurora. Contoh berikut mencatat jumlah Replika Aurora dalam sebuah cluster yang `my-db-cluster` disebut, dengan kapasitas minimum satu Replika Aurora dan kapasitas maksimum delapan Replika Aurora.

```
aws application-autoscaling register-scalable-target \
  --service-namespace rds \
  --scalable-dimension rds:cluster:ReadReplicaCount \
  --resource-id cluster:my-db-cluster \
  --min-capacity 1 \
```

```
  --max-capacity 8
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Sumber daya terkait

Untuk informasi selengkapnya, lihat Auto Scaling Amazon Aurora dengan Replika Aurora di Panduan Pengguna Amazon RDS untuk Aurora.

Amazon Comprehend dan Application Auto Scaling

Anda dapat menskalakan klasifikasi dokumen Amazon Comprehend dan titik akhir pengenal entitas menggunakan kebijakan penskalaan pelacakan target dan penskalaan terjadwal.

Gunakan informasi berikut untuk membantu Anda mengintegrasikan Amazon Comprehend dengan Application Auto Scaling.

Peran terkait layanan yang dibuat untuk Amazon Comprehend

Peran terkait layanan berikut dibuat secara otomatis di Anda Akun AWS saat mendaftarkan sumber daya Amazon Comprehend sebagai target yang dapat diskalakan dengan Application Auto Scaling. Peran ini memungkinkan Application Auto Scaling untuk melakukan operasi yang didukung dalam akun Anda. Untuk informasi selengkapnya, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_ComprehendEndpoint`

Prinsipal layanan yang digunakan oleh peran terkait layanan

Peran terkait layanan di bagian sebelumnya hanya dapat diasumsikan oleh kepala layanan yang diotorisasi oleh hubungan kepercayaan yang ditentukan untuk peran tersebut. Peran terkait layanan yang digunakan oleh Application Auto Scaling memberikan akses ke prinsipal layanan berikut:

- `comprehend.application-autoscaling.amazonaws.com`

Mendaftarkan sumber daya Amazon Comprehend sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk klasifikasi dokumen Amazon Comprehend atau titik akhir pengenal entitas. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling. Target yang dapat diskalakan diidentifikasi secara unik dengan kombinasi ID sumber daya, dimensi yang dapat diskalakan, dan namespace.

Untuk mengonfigurasi penskalaan otomatis menggunakan AWS CLI atau salah AWS SDKs satu, Anda dapat menggunakan opsi berikut:

- AWS CLI:

  Panggil register-scalable-targetperintah untuk titik akhir klasifikasi dokumen. Contoh berikut mencatat jumlah unit inferensi yang diinginkan untuk digunakan oleh model untuk titik akhir pengklasifikasi dokumen menggunakan ARN titik akhir, dengan kapasitas minimum satu unit inferensi dan kapasitas maksimum tiga unit inferensi.

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace comprehend \
    --scalable-dimension comprehend:document-classifier-endpoint:DesiredInferenceUnits
   \
    --resource-id arn:aws:comprehend:us-west-2:123456789012:document-classifier-
  endpoint/EXAMPLE \
    --min-capacity 1 \
    --max-capacity 3
  ```

  Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

  ```
  {
  ```

```
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Panggil register-scalable-targetperintah untuk titik akhir pengenal entitas. Contoh berikut mencatat jumlah unit inferensi yang diinginkan untuk digunakan oleh model untuk pengenal entitas menggunakan ARN titik akhir, dengan kapasitas minimum satu unit inferensi dan kapasitas maksimum tiga unit inferensi.

```
aws application-autoscaling register-scalable-target \
  --service-namespace comprehend \
  --scalable-dimension comprehend:entity-recognizer-endpoint:DesiredInferenceUnits \
  --resource-id arn:aws:comprehend:us-west-2:123456789012:entity-recognizer-
endpoint/EXAMPLE \
  --min-capacity 1 \
  --max-capacity 3
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Sumber daya terkait

Untuk informasi selengkapnya, lihat Penskalaan otomatis dengan titik akhir di Panduan Pengembang Amazon Comprehend.

Amazon DynamoDB dan Application Auto Scaling

Anda dapat menskalakan tabel DynamoDB dan indeks sekunder global menggunakan kebijakan penskalaan pelacakan target dan penskalaan terjadwal.

Gunakan informasi berikut untuk membantu Anda mengintegrasikan DynamoDB dengan Application Auto Scaling.

Peran terkait layanan yang dibuat untuk DynamoDB

Peran terkait layanan berikut dibuat secara otomatis di Akun AWS saat mendaftarkan sumber daya DynamoDB sebagai target yang dapat diskalakan dengan Application Auto Scaling. Peran ini memungkinkan Application Auto Scaling untuk melakukan operasi yang didukung dalam akun Anda. Untuk informasi selengkapnya, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_DynamoDBTable`

Prinsipal layanan yang digunakan oleh peran terkait layanan

Peran terkait layanan di bagian sebelumnya hanya dapat diasumsikan oleh kepala layanan yang diotorisasi oleh hubungan kepercayaan yang ditentukan untuk peran tersebut. Peran terkait layanan yang digunakan oleh Application Auto Scaling memberikan akses ke prinsipal layanan berikut:

- `dynamodb.application-autoscaling.amazonaws.com`

Mendaftarkan sumber daya DynamoDB sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk tabel DynamoDB atau indeks sekunder global. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling. Target yang dapat diskalakan diidentifikasi secara unik dengan kombinasi ID sumber daya, dimensi yang dapat diskalakan, dan namespace.

Jika Anda mengonfigurasi penskalaan otomatis menggunakan konsol DynamoDB, maka DynamoDB secara otomatis mendaftarkan target yang dapat diskalakan untuk Anda.

Jika Anda ingin mengonfigurasi penskalaan otomatis menggunakan AWS CLI atau salah AWS SDKs satu, Anda dapat menggunakan opsi berikut:

- AWS CLI:

Panggil register-scalable-targetperintah untuk kapasitas tulis tabel. Contoh berikut mencatat kapasitas tulis yang disediakan dari tabel yang disebut`my-table`, dengan kapasitas minimum lima unit kapasitas tulis dan kapasitas maksimum 10 unit kapasitas tulis:

```
aws application-autoscaling register-scalable-target \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:WriteCapacityUnits \
  --resource-id table/my-table \
  --min-capacity 5 \
  --max-capacity 10
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan:

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Panggil register-scalable-targetperintah untuk kapasitas baca tabel. Contoh berikut mencatat kapasitas baca yang disediakan dari tabel yang disebut`my-table`, dengan kapasitas minimum lima unit kapasitas baca dan kapasitas maksimum 10 unit baca:

```
aws application-autoscaling register-scalable-target \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits \
  --resource-id table/my-table \
  --min-capacity 5 \
  --max-capacity 10
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan:

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Panggil register-scalable-targetperintah untuk kapasitas tulis indeks sekunder global. Contoh berikut mencatat kapasitas tulis yang disediakan dari indeks sekunder global yang disebut`my-`

`table-index`, dengan kapasitas minimum lima unit kapasitas tulis dan kapasitas maksimum 10 unit kapasitas tulis:

```
aws application-autoscaling register-scalable-target \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:index:WriteCapacityUnits \
  --resource-id table/my-table/index/my-table-index \
  --min-capacity 5 \
  --max-capacity 10
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan:

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Panggil register-scalable-targetperintah untuk kapasitas baca indeks sekunder global. Contoh berikut mencatat kapasitas baca yang disediakan dari indeks sekunder global yang disebut`my-table-index`, dengan kapasitas minimum lima unit kapasitas baca dan kapasitas maksimum 10 unit kapasitas baca:

```
aws application-autoscaling register-scalable-target \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:index:ReadCapacityUnits \
  --resource-id table/my-table/index/my-table-index \
  --min-capacity 5 \
  --max-capacity 10
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan:

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Sumber daya terkait

Jika Anda baru memulai Application Auto Scaling, Anda dapat menemukan informasi berguna tambahan tentang penskalaan sumber daya DynamoDB Anda dalam dokumentasi berikut:

- Mengelola kapasitas throughput dengan DynamoDB Auto Scaling di Panduan Pengembang Amazon DynamoDB
- Mengevaluasi pengaturan penskalaan otomatis tabel Anda di Panduan Pengembang Amazon DynamoDB
- Cara menggunakan CloudFormation untuk mengkonfigurasi auto scaling untuk tabel DynamoDB dan indeks di Blog AWS

Amazon ECS dan Application Auto Scaling

Anda dapat menskalakan layanan ECS menggunakan kebijakan penskalaan pelacakan target, kebijakan penskalaan prediktif, kebijakan penskalaan langkah, dan penskalaan terjadwal.

Gunakan informasi berikut untuk membantu Anda mengintegrasikan Amazon ECS dengan Application Auto Scaling.

Peran terkait layanan yang dibuat untuk Amazon ECS

Peran terkait layanan berikut dibuat secara otomatis di Akun AWS saat mendaftarkan resource Amazon ECS sebagai target yang dapat diskalakan dengan Application Auto Scaling. Peran ini memungkinkan Application Auto Scaling untuk melakukan operasi yang didukung dalam akun Anda. Untuk informasi selengkapnya, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_ECSService`

Prinsipal layanan yang digunakan oleh peran terkait layanan

Peran terkait layanan di bagian sebelumnya hanya dapat diasumsikan oleh kepala layanan yang diotorisasi oleh hubungan kepercayaan yang ditentukan untuk peran tersebut. Peran terkait layanan yang digunakan oleh Application Auto Scaling memberikan akses ke prinsipal layanan berikut:

- `ecs.application-autoscaling.amazonaws.com`

Mendaftarkan layanan ECS sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk layanan Amazon ECS. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling. Target yang dapat diskalakan diidentifikasi secara unik dengan kombinasi ID sumber daya, dimensi yang dapat diskalakan, dan namespace.

Jika Anda mengonfigurasi penskalaan otomatis menggunakan konsol Amazon ECS, Amazon ECS secara otomatis mendaftarkan target yang dapat diskalakan untuk Anda.

Jika Anda ingin mengonfigurasi penskalaan otomatis menggunakan AWS CLI atau salah AWS SDKs satu, Anda dapat menggunakan opsi berikut:

- AWS CLI:

  Panggil register-scalable-targetperintah untuk layanan Amazon ECS. Contoh berikut mendaftarkan target yang dapat diskalakan untuk layanan yang dipanggil`sample-app-service`, berjalan di `default` cluster, dengan jumlah tugas minimum satu tugas dan jumlah tugas maksimum 10 tugas.

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace ecs \
    --scalable-dimension ecs:service:DesiredCount \
    --resource-id service/default/sample-app-service \
    --min-capacity 1 \
    --max-capacity 10
  ```

  Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

  ```
  {
  ```

```
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Sumber daya terkait

Jika Anda baru memulai Application Auto Scaling, Anda dapat menemukan informasi berguna tambahan tentang penskalaan sumber daya Amazon ECS Anda dalam dokumentasi berikut:

- Layanan penskalaan otomatis di Panduan Pengembang Layanan Kontainer Elastis Amazon
- Optimalkan penskalaan otomatis layanan Amazon ECS di Panduan Pengembang Layanan Kontainer Elastis Amazon

> **Note**
>
> Untuk petunjuk penangguhan proses penskalaan saat penerapan Amazon ECS sedang berlangsung, lihat dokumentasi berikut:
> Servis penskalaan dan penerapan otomatis di Panduan Pengembang Layanan Amazon Elastic Container

ElastiCache dan Application Auto Scaling

Anda dapat menskalakan grup ElastiCache replikasi Amazon secara horizontal (Redis OSS dan Valkey) dan kluster yang dirancang sendiri Memcached menggunakan kebijakan penskalaan pelacakan target dan penskalaan terjadwal.

Untuk mengintegrasikan ElastiCache dengan Application Auto Scaling, gunakan informasi berikut.

Peran terkait layanan dibuat untuk ElastiCache

Peran terkait layanan berikut dibuat secara otomatis di dalam Anda Akun AWS saat mendaftarkan ElastiCache sumber daya sebagai target yang dapat diskalakan dengan Application Auto Scaling. Peran ini memungkinkan Application Auto Scaling untuk melakukan operasi yang didukung dalam akun Anda. Untuk informasi selengkapnya, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG`

Prinsipal layanan yang digunakan oleh peran terkait layanan

Peran terkait layanan di bagian sebelumnya hanya dapat diasumsikan oleh kepala layanan yang diotorisasi oleh hubungan kepercayaan yang ditentukan untuk peran tersebut. Peran terkait layanan yang digunakan oleh Application Auto Scaling memberikan akses ke prinsipal layanan berikut:

- `elasticache.application-autoscaling.amazonaws.com`

Mendaftarkan ElastiCache sumber daya sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk grup ElastiCache replikasi, cluster, atau node. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling. Target yang dapat diskalakan diidentifikasi secara unik dengan kombinasi ID sumber daya, dimensi yang dapat diskalakan, dan namespace.

Jika Anda mengonfigurasi penskalaan otomatis menggunakan ElastiCache konsol, maka ElastiCache secara otomatis mendaftarkan target yang dapat diskalakan untuk Anda.

Jika Anda ingin mengonfigurasi penskalaan otomatis menggunakan AWS CLI atau salah AWS SDKs satu, Anda dapat menggunakan opsi berikut:

- AWS CLI:

  Panggil register-scalable-targetperintah untuk grup ElastiCache replikasi. Contoh berikut mendaftarkan jumlah kelompok node yang diinginkan untuk grup replikasi yang disebut`mycluster1`, dengan kapasitas minimum satu dan kapasitas maksimum lima.

```
aws application-autoscaling register-scalable-target \
   --service-namespace elasticache \
   --scalable-dimension elasticache:replication-group:NodeGroups \
   --resource-id replication-group/mycluster1 \
   --min-capacity 1 \
   --max-capacity 5
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Contoh berikut mencatat jumlah replika yang diinginkan per grup node untuk grup replikasi yang disebut`mycluster2`, dengan kapasitas minimum satu dan kapasitas maksimum lima.

```
aws application-autoscaling register-scalable-target \
   --service-namespace elasticache \
   --scalable-dimension elasticache:replication-group:Replicas \
   --resource-id replication-group/mycluster2 \
   --min-capacity 1 \
   --max-capacity 5
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/234abcd56ab78cd901ef1234567890ab1234"
}
```

Contoh berikut mencatat jumlah node yang diinginkan untuk cluster yang disebut`mynode1`, dengan kapasitas minimum 20 dan kapasitas maksimum 50.

```
aws application-autoscaling register-scalable-target \
   --service-namespace elasticache \
   --scalable-dimension elasticache:cache-cluster:Nodes \
   --resource-id cache-cluster/mynode1 \
   --min-capacity 20 \
```

```
    --max-capacity 50
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/01234abcd56ab78cd901ef1234567890ab12"
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Sumber daya terkait

Untuk informasi selengkapnya, lihat Auto Scaling Valkey dan Redis OSS cluster dan Scaling cluster untuk Memcached di Panduan Pengguna Amazon. ElastiCache

Amazon Keyspaces (untuk Apache Cassandra) dan Application Auto Scaling

Anda dapat menskalakan tabel Amazon Keyspaces menggunakan kebijakan penskalaan pelacakan target dan penskalaan terjadwal.

Gunakan informasi berikut untuk membantu Anda mengintegrasikan Amazon Keyspaces dengan Application Auto Scaling.

Peran terkait layanan yang dibuat untuk Amazon Keyspaces

Peran terkait layanan berikut dibuat secara otomatis di Akun AWS saat mendaftarkan resource Amazon Keyspaces sebagai target yang dapat diskalakan dengan Application Auto Scaling. Peran ini memungkinkan Application Auto Scaling untuk melakukan operasi yang didukung dalam akun Anda. Untuk informasi selengkapnya, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_CassandraTable`

Prinsipal layanan yang digunakan oleh peran terkait layanan

Peran terkait layanan di bagian sebelumnya hanya dapat diasumsikan oleh kepala layanan yang diotorisasi oleh hubungan kepercayaan yang ditentukan untuk peran tersebut. Peran terkait layanan yang digunakan oleh Application Auto Scaling memberikan akses ke prinsipal layanan berikut:

- `cassandra.application-autoscaling.amazonaws.com`

Mendaftarkan tabel Amazon Keyspaces sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk tabel Amazon Keyspaces. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling. Target yang dapat diskalakan diidentifikasi secara unik dengan kombinasi ID sumber daya, dimensi yang dapat diskalakan, dan namespace.

Jika Anda mengonfigurasi penskalaan otomatis menggunakan konsol Amazon Keyspaces, Amazon Keyspaces secara otomatis mendaftarkan target yang dapat diskalakan untuk Anda.

Jika Anda ingin mengonfigurasi penskalaan otomatis menggunakan AWS CLI atau salah AWS SDKs satu, Anda dapat menggunakan opsi berikut:

- AWS CLI:

  Panggil register-scalable-targetperintah untuk tabel Amazon Keyspaces. Contoh berikut mencatat kapasitas tulis yang disediakan dari tabel yang disebut`mytable`, dengan kapasitas minimum lima unit kapasitas tulis dan kapasitas maksimum 10 unit kapasitas tulis.

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace cassandra \
    --scalable-dimension cassandra:table:WriteCapacityUnits \
    --resource-id keyspace/mykeyspace/table/mytable \
    --min-capacity 5 \
    --max-capacity 10
  ```

  Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

  ```
  {
  ```

```
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Contoh berikut mencatat kapasitas baca yang disediakan dari tabel yang disebut`mytable`, dengan kapasitas minimum lima unit kapasitas baca dan kapasitas maksimum 10 unit kapasitas baca.

```
aws application-autoscaling register-scalable-target \
  --service-namespace cassandra \
  --scalable-dimension cassandra:table:ReadCapacityUnits \
  --resource-id keyspace/mykeyspace/table/mytable \
  --min-capacity 5 \
  --max-capacity 10
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Sumber daya terkait

Untuk informasi selengkapnya, lihat Mengelola kapasitas throughput secara otomatis dengan penskalaan otomatis Amazon Keyspaces di Panduan Pengembang Amazon Keyspaces.

AWS Lambda dan Application Auto Scaling

Anda dapat menskalakan AWS Lambda konkurensi yang disediakan menggunakan kebijakan penskalaan pelacakan target dan penskalaan terjadwal.

Gunakan informasi berikut untuk membantu Anda mengintegrasikan Lambda dengan Application Auto Scaling.

Peran terkait layanan yang dibuat untuk Lambda

Peran terkait layanan berikut dibuat secara otomatis di dalam Anda Akun AWS saat mendaftarkan sumber daya Lambda sebagai target yang dapat diskalakan dengan Application Auto Scaling. Peran ini memungkinkan Application Auto Scaling untuk melakukan operasi yang didukung dalam akun Anda. Untuk informasi selengkapnya, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_LambdaConcurrency`

Prinsipal layanan yang digunakan oleh peran terkait layanan

Peran terkait layanan di bagian sebelumnya hanya dapat diasumsikan oleh kepala layanan yang diotorisasi oleh hubungan kepercayaan yang ditentukan untuk peran tersebut. Peran terkait layanan yang digunakan oleh Application Auto Scaling memberikan akses ke prinsipal layanan berikut:

- `lambda.application-autoscaling.amazonaws.com`

Mendaftarkan fungsi Lambda sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk fungsi Lambda. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling. Target yang dapat diskalakan diidentifikasi secara unik dengan kombinasi ID sumber daya, dimensi yang dapat diskalakan, dan namespace.

Untuk mengonfigurasi penskalaan otomatis menggunakan AWS CLI atau salah AWS SDKs satu, Anda dapat menggunakan opsi berikut:

- AWS CLI:

  Panggil register-scalable-targetperintah untuk fungsi Lambda. Contoh berikut mendaftarkan konkurensi yang disediakan untuk alias yang dipanggil `BLUE` untuk fungsi yang dipanggil`my-function`, dengan kapasitas minimum 0 dan kapasitas maksimum 100.

```
aws application-autoscaling register-scalable-target \
  --service-namespace lambda \
```

```
  --scalable-dimension lambda:function:ProvisionedConcurrency \
  --resource-id function:my-function:BLUE \
  --min-capacity 0 \
  --max-capacity 100
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Sumber daya terkait

Jika Anda baru memulai Application Auto Scaling, Anda dapat menemukan informasi berguna tambahan tentang penskalaan fungsi Lambda Anda dalam dokumentasi berikut:

- Mengkonfigurasi konkurensi yang disediakan dalam Panduan Pengembang AWS Lambda
- Menjadwalkan Lambda Provisioned Concurrency untuk penggunaan puncak berulang di Blog AWS

Amazon Managed Streaming untuk Apache Kafka (MSK) dan Application Auto Scaling

Anda dapat meningkatkan skala penyimpanan klaster MSK Amazon menggunakan kebijakan penskalaan pelacakan target. Penskalaan berdasarkan kebijakan pelacakan target dinonaktifkan.

Gunakan informasi berikut untuk membantu Anda mengintegrasikan Amazon MSK dengan Application Auto Scaling.

Peran terkait layanan yang dibuat untuk Amazon MSK

Peran terkait layanan berikut dibuat secara otomatis di Anda Akun AWS saat mendaftarkan sumber daya MSK Amazon sebagai target yang dapat diskalakan dengan Application Auto Scaling. Peran ini memungkinkan Application Auto Scaling untuk melakukan operasi yang didukung dalam akun Anda. Untuk informasi selengkapnya, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_KafkaCluster`

Prinsipal layanan yang digunakan oleh peran terkait layanan

Peran terkait layanan di bagian sebelumnya hanya dapat diasumsikan oleh kepala layanan yang diotorisasi oleh hubungan kepercayaan yang ditentukan untuk peran tersebut. Peran terkait layanan yang digunakan oleh Application Auto Scaling memberikan akses ke prinsipal layanan berikut:

- `kafka.application-autoscaling.amazonaws.com`

Mendaftarkan penyimpanan cluster MSK Amazon sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan untuk ukuran volume penyimpanan per broker kluster MSK Amazon. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan oleh Application Auto Scaling. Target yang dapat diskalakan diidentifikasi secara unik dengan kombinasi ID sumber daya, dimensi yang dapat diskalakan, dan namespace.

Jika Anda mengonfigurasi penskalaan otomatis menggunakan konsol MSK Amazon, maka Amazon MSK secara otomatis mendaftarkan target yang dapat diskalakan untuk Anda.

Jika Anda ingin mengonfigurasi penskalaan otomatis menggunakan AWS CLI atau salah AWS SDKs satu, Anda dapat menggunakan opsi berikut:

- AWS CLI:

  Panggil register-scalable-targetperintah untuk cluster MSK Amazon. Contoh berikut mencatat ukuran volume penyimpanan per broker dari cluster MSK Amazon, dengan kapasitas minimum 100 GiB dan kapasitas maksimum 800 GiB.

```
aws application-autoscaling register-scalable-target \
   --service-namespace kafka \
   --scalable-dimension kafka:broker-storage:VolumeSize \
   --resource-id arn:aws:kafka:us-east-1:123456789012:cluster/demo-
cluster-1/6357e0b2-0e6a-4b86-a0b4-70df934c2e31-5 \
   --min-capacity 100 \
   --max-capacity 800
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

> **Note**
>
> Ketika sebuah klaster Amazon MSK adalah target yang dapat diskalakan, skala di dinonaktifkan dan tidak dapat diaktifkan.

Sumber daya terkait

Untuk informasi selengkapnya, lihat Penskalaan otomatis untuk klaster MSK Amazon di Panduan Pengembang Amazon Managed Streaming for Apache Kafka.

Amazon Neptunus dan Application Auto Scaling

Anda dapat menskalakan klaster Neptunus menggunakan kebijakan penskalaan pelacakan target dan penskalaan terjadwal.

Gunakan informasi berikut untuk membantu Anda mengintegrasikan Neptunus dengan Application Auto Scaling.

Peran terkait layanan yang dibuat untuk Neptunus

Peran terkait layanan berikut dibuat secara otomatis di dalam Anda Akun AWS saat mendaftarkan sumber daya Neptunus sebagai target yang dapat diskalakan dengan Application Auto Scaling. Peran ini memungkinkan Application Auto Scaling untuk melakukan operasi yang didukung dalam akun Anda. Untuk informasi selengkapnya, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_NeptuneCluster`

Prinsipal layanan yang digunakan oleh peran terkait layanan

Peran terkait layanan di bagian sebelumnya hanya dapat diasumsikan oleh kepala layanan yang diotorisasi oleh hubungan kepercayaan yang ditentukan untuk peran tersebut. Peran terkait layanan yang digunakan oleh Application Auto Scaling memberikan akses ke prinsipal layanan berikut:

- `neptune.application-autoscaling.amazonaws.com`

Mendaftarkan cluster Neptunus sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk cluster Neptunus. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling. Target yang dapat diskalakan diidentifikasi secara unik dengan kombinasi ID sumber daya, dimensi yang dapat diskalakan, dan namespace.

Untuk mengonfigurasi penskalaan otomatis menggunakan AWS CLI atau salah AWS SDKs satu, Anda dapat menggunakan opsi berikut:

- AWS CLI:

  Panggil register-scalable-targetperintah untuk cluster Neptunus. Contoh berikut mencatat kapasitas yang diinginkan dari sebuah cluster yang disebut`mycluster`, dengan kapasitas minimum satu dan kapasitas maksimum delapan.

```
aws application-autoscaling register-scalable-target \
  --service-namespace neptune \
  --scalable-dimension neptune:cluster:ReadReplicaCount \
  --resource-id cluster:mycluster \
  --min-capacity 1 \
  --max-capacity 8
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Sumber daya terkait

Untuk informasi selengkapnya, lihat Penskalaan otomatis jumlah replika di kluster DB Amazon Neptunus di Panduan Pengguna Neptunus.

Amazon SageMaker AI dan Application Auto Scaling

Anda dapat menskalakan varian titik akhir SageMaker AI, konkurensi yang disediakan untuk titik akhir tanpa server, dan komponen inferensi menggunakan kebijakan penskalaan pelacakan target, kebijakan penskalaan langkah, dan penskalaan terjadwal.

Gunakan informasi berikut untuk membantu Anda mengintegrasikan SageMaker AI dengan Application Auto Scaling.

Peran terkait layanan yang dibuat untuk AI SageMaker

Peran terkait layanan berikut dibuat secara otomatis di Anda Akun AWS saat mendaftarkan sumber daya SageMaker AI sebagai target yang dapat diskalakan dengan Application Auto Scaling. Peran ini memungkinkan Application Auto Scaling untuk melakukan operasi yang didukung dalam akun Anda.

Untuk informasi selengkapnya, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint`

Prinsipal layanan yang digunakan oleh peran terkait layanan

Peran terkait layanan di bagian sebelumnya hanya dapat diasumsikan oleh kepala layanan yang diotorisasi oleh hubungan kepercayaan yang ditentukan untuk peran tersebut. Peran terkait layanan yang digunakan oleh Application Auto Scaling memberikan akses ke prinsipal layanan berikut:

- `sagemaker.application-autoscaling.amazonaws.com`

Mendaftarkan varian endpoint SageMaker AI sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk model SageMaker AI (varian). Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling. Target yang dapat diskalakan diidentifikasi secara unik dengan kombinasi ID sumber daya, dimensi yang dapat diskalakan, dan namespace.

Jika Anda mengonfigurasi penskalaan otomatis menggunakan konsol SageMaker AI, maka SageMaker AI secara otomatis mendaftarkan target yang dapat diskalakan untuk Anda.

Jika Anda ingin mengonfigurasi penskalaan otomatis menggunakan AWS CLI atau salah AWS SDKs satu, Anda dapat menggunakan opsi berikut:

- AWS CLI:

  Panggil register-scalable-targetperintah untuk varian produk. Contoh berikut mencatat jumlah instans yang diinginkan untuk varian produk yang disebut`my-variant`, berjalan pada `my-endpoint` titik akhir, dengan kapasitas minimum satu instance dan kapasitas maksimum delapan instance.

```
aws application-autoscaling register-scalable-target \
  --service-namespace sagemaker \
  --scalable-dimension sagemaker:variant:DesiredInstanceCount \
```

```
  --resource-id endpoint/my-endpoint/variant/my-variant \
  --min-capacity 1 \
  --max-capacity 8
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Mendaftarkan konkurensi titik akhir tanpa server yang disediakan sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling juga memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk konkurensi titik akhir tanpa server yang disediakan.

Jika Anda mengonfigurasi penskalaan otomatis menggunakan konsol SageMaker AI, maka SageMaker AI secara otomatis mendaftarkan target yang dapat diskalakan untuk Anda.

Jika tidak, gunakan salah satu metode berikut untuk mendaftarkan target yang dapat diskalakan:

- AWS CLI:

  Panggil register-scalable-targetperintah untuk varian produk. Contoh berikut mendaftarkan konkurensi yang disediakan untuk varian produk yang disebut`my-variant`, berjalan pada `my-endpoint` titik akhir, dengan kapasitas minimum satu dan kapasitas maksimum sepuluh.

```
aws application-autoscaling register-scalable-target \
  --service-namespace sagemaker \
  --scalable-dimension sagemaker:variant:DesiredProvisionedConcurrency \
  --resource-id endpoint/my-endpoint/variant/my-variant \
  --min-capacity 1 \
```

```
  --max-capacity 10
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Mendaftarkan komponen inferensi sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling juga memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk komponen inferensi.

- AWS CLI:

  Panggil register-scalable-targetperintah untuk komponen inferensi. Contoh berikut mencatat jumlah salinan yang diinginkan untuk komponen inferensi yang disebut`my-inference-component`, dengan kapasitas minimum nol salinan dan kapasitas maksimum tiga salinan.

```
aws application-autoscaling register-scalable-target \
  --service-namespace sagemaker \
  --scalable-dimension sagemaker:inference-component:DesiredCopyCount \
  --resource-id inference-component/my-inference-component \
  --min-capacity 0 \
  --max-capacity 3
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
```

```
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Sumber daya terkait

Jika Anda baru memulai Application Auto Scaling, Anda dapat menemukan informasi berguna tambahan tentang penskalaan sumber daya SageMaker AI Anda di Panduan Pengembang Amazon SageMaker AI:

- Secara otomatis menskalakan model Amazon SageMaker AI
- Secara otomatis menskalakan Konkurensi yang Disediakan untuk titik akhir tanpa server
- Menetapkan kebijakan penskalaan otomatis untuk penerapan titik akhir multi-model
- Skala otomatis titik akhir asinkron

> **Note**
>
> Pada tahun 2023, SageMaker AI memperkenalkan kemampuan inferensi baru yang dibangun di atas titik akhir inferensi waktu nyata. Anda membuat titik akhir SageMaker AI dengan konfigurasi titik akhir yang menentukan jenis instans dan jumlah instans awal untuk titik akhir. Kemudian, buat komponen inferensi, yang merupakan objek hosting SageMaker AI yang dapat Anda gunakan untuk menerapkan model ke titik akhir. Untuk informasi tentang penskalaan komponen inferensi, lihat Amazon SageMaker AI menambahkan kemampuan inferensi baru untuk membantu mengurangi biaya penerapan model pondasi dan latensi serta Mengurangi biaya penerapan model rata-rata sebesar 50% menggunakan fitur terbaru Amazon AI di Blog. SageMaker AWS

Armada Spot Amazon EC2 dan Application Auto Scaling

Anda dapat menskalakan Armada Spot menggunakan kebijakan penskalaan pelacakan target, kebijakan penskalaan langkah, dan penskalaan terjadwal.

Gunakan informasi berikut untuk membantu Anda mengintegrasikan Armada Spot dengan Application Auto Scaling.

Peran terkait layanan dibuat untuk Armada Spot

Peran terkait layanan berikut dibuat secara otomatis di dalam Anda Akun AWS saat mendaftarkan sumber daya Armada Spot sebagai target yang dapat diskalakan dengan Application Auto Scaling. Peran ini memungkinkan Application Auto Scaling untuk melakukan operasi yang didukung dalam akun Anda. Untuk informasi selengkapnya, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest`

Prinsipal layanan yang digunakan oleh peran terkait layanan

Peran terkait layanan di bagian sebelumnya hanya dapat diasumsikan oleh kepala layanan yang diotorisasi oleh hubungan kepercayaan yang ditentukan untuk peran tersebut. Peran terkait layanan yang digunakan oleh Application Auto Scaling memberikan akses ke prinsipal layanan berikut:

- `ec2.application-autoscaling.amazonaws.com`

Mendaftarkan Armada Spot sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk Armada Spot. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling. Target yang dapat diskalakan diidentifikasi secara unik dengan kombinasi ID sumber daya, dimensi yang dapat diskalakan, dan namespace.

Jika Anda mengonfigurasi penskalaan otomatis menggunakan konsol Spot Fleet, maka Spot Fleet secara otomatis mendaftarkan target yang dapat diskalakan untuk Anda.

Jika Anda ingin mengonfigurasi penskalaan otomatis menggunakan AWS CLI atau salah AWS SDKs satu, Anda dapat menggunakan opsi berikut:

- AWS CLI:

Panggil register-scalable-targetperintah untuk Armada Spot. Contoh berikut mencatat kapasitas target Armada Spot menggunakan ID permintaannya, dengan kapasitas minimum dua instans dan kapasitas maksimum 10 instans.

```
aws application-autoscaling register-scalable-target \
  --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --min-capacity 2 \
  --max-capacity 10
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Sumber daya terkait

Untuk informasi selengkapnya, lihat Memahami penskalaan otomatis untuk Armada Spot di Panduan Pengguna Amazon EC2.

Amazon WorkSpaces dan Application Auto Scaling

Anda dapat menskalakan kumpulan WorkSpaces menggunakan kebijakan penskalaan pelacakan target, kebijakan penskalaan langkah, dan penskalaan terjadwal.

Gunakan informasi berikut untuk membantu Anda berintegrasi WorkSpaces dengan Application Auto Scaling.

Peran terkait layanan dibuat untuk WorkSpaces

Application Auto Scaling secara otomatis membuat peran terkait layanan yang disebutkan AWSServiceRoleForApplicationAutoScaling_WorkSpacesPool dalam nama Anda Akun AWS saat Anda mendaftarkan WorkSpaces sumber daya sebagai target yang dapat diskalakan dengan Application Auto Scaling. Untuk informasi selengkapnya, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

Peran terkait layanan ini menggunakan kebijakan terkelola. AWSApplication AutoscalingWorkSpacesPoolPolicy Kebijakan ini memberikan izin Application Auto Scaling untuk memanggil WorkSpaces Amazon atas nama Anda. Untuk informasi selengkapnya, lihat AWSApplicationAutoscalingWorkSpacesPoolPolicydi Referensi Kebijakan AWS Terkelola.

Prinsipal layanan yang digunakan oleh peran terkait layanan

Peran terkait layanan mempercayai prinsip layanan berikut untuk mengambil peran:

- `workspaces.application-autoscaling.amazonaws.com`

Mendaftarkan WorkSpaces pool sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk. WorkSpaces Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling. Target yang dapat diskalakan diidentifikasi secara unik dengan kombinasi ID sumber daya, dimensi yang dapat diskalakan, dan namespace.

Jika Anda mengonfigurasi penskalaan otomatis menggunakan WorkSpaces konsol, maka WorkSpaces secara otomatis mendaftarkan target yang dapat diskalakan untuk Anda.

Jika Anda ingin mengonfigurasi penskalaan otomatis menggunakan AWS CLI atau salah AWS SDKs satu, Anda dapat menggunakan opsi berikut:

- AWS CLI:

  Panggil register-scalable-targetperintah untuk kumpulan WorkSpaces. Contoh berikut mencatat kapasitas target kumpulan WorkSpaces menggunakan ID permintaannya, dengan kapasitas minimum dua desktop virtual dan kapasitas maksimum sepuluh desktop virtual.

```
aws application-autoscaling register-scalable-target \
  --service-namespace workspaces \
  --resource-id workspacespool/wspool-abcdef012 \
  --scalable-dimension workspaces:workspacespool:DesiredUserSessions \
  --min-capacity 2 \
  --max-capacity 10
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Sumber daya terkait

Untuk informasi selengkapnya, lihat Auto Scaling for WorkSpaces Pools di Panduan WorkSpaces Administrasi Amazon.

Sumber daya khusus dan Application Auto Scaling

Anda dapat menskalakan sumber daya khusus menggunakan kebijakan penskalaan pelacakan target, kebijakan penskalaan langkah, dan penskalaan terjadwal.

Gunakan informasi berikut untuk membantu Anda mengintegrasikan sumber daya khusus dengan Application Auto Scaling.

Peran terkait layanan yang dibuat untuk sumber daya khusus

Peran terkait layanan berikut dibuat secara otomatis di Akun AWS saat mendaftarkan sumber daya kustom sebagai target yang dapat diskalakan dengan Application Auto Scaling. Peran ini memungkinkan Application Auto Scaling untuk melakukan operasi yang didukung dalam akun Anda.

Untuk informasi selengkapnya, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_CustomResource`

Prinsipal layanan yang digunakan oleh peran terkait layanan

Peran terkait layanan di bagian sebelumnya hanya dapat diasumsikan oleh kepala layanan yang diotorisasi oleh hubungan kepercayaan yang ditentukan untuk peran tersebut. Peran terkait layanan yang digunakan oleh Application Auto Scaling memberikan akses ke prinsipal layanan berikut:

- `custom-resource.application-autoscaling.amazonaws.com`

Mendaftarkan sumber daya kustom sebagai target yang dapat diskalakan dengan Application Auto Scaling

Application Auto Scaling memerlukan target yang dapat diskalakan sebelum Anda dapat membuat kebijakan penskalaan atau tindakan terjadwal untuk sumber daya kustom. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling. Target yang dapat diskalakan diidentifikasi secara unik dengan kombinasi ID sumber daya, dimensi yang dapat diskalakan, dan namespace.

Untuk mengonfigurasi penskalaan otomatis menggunakan AWS CLI atau salah AWS SDKs satu, Anda dapat menggunakan opsi berikut:

- AWS CLI:

  Panggil register-scalable-targetperintah untuk sumber daya khusus. Contoh berikut mendaftarkan sumber daya kustom sebagai target yang dapat diskalakan, dengan jumlah minimum yang diinginkan dari satu unit kapasitas dan jumlah maksimum yang diinginkan dari 10 unit kapasitas. `custom-resource-id.txt`File berisi string yang mengidentifikasi ID sumber daya, yang mewakili jalur ke sumber daya kustom melalui titik akhir Amazon API Gateway Anda.

```
aws application-autoscaling register-scalable-target \
  --service-namespace custom-resource \
  --scalable-dimension custom-resource:ResourceType:Property \
  --resource-id file://~/custom-resource-id.txt \
  --min-capacity 1 \
```

```
  --max-capacity 10
```

Isi dari `custom-resource-id.txt`:

```
https://example.execute-api.us-west-2.amazonaws.com/prod/
scalableTargetDimensions/1-23456789
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Panggil RegisterScalableTargetoperasi dan berikan`ResourceId`,`ScalableDimension`,`ServiceNamespace`,`MinCapacity`, dan `MaxCapacity` sebagai parameter.

Sumber daya terkait

Jika Anda baru memulai Application Auto Scaling, Anda dapat menemukan informasi berguna tambahan tentang penskalaan sumber daya kustom Anda dalam dokumentasi berikut:

GitHubrepositori

Konfigurasikan sumber daya Application Auto Scaling menggunakan AWS CloudFormation

Application Auto Scaling terintegrasi dengan AWS CloudFormation, layanan yang membantu Anda memodelkan dan mengatur AWS sumber daya Anda sehingga Anda dapat menghabiskan lebih sedikit waktu untuk membuat dan mengelola sumber daya dan infrastruktur Anda. Anda membuat templat yang menjelaskan semua AWS sumber daya yang Anda inginkan, dan menyediakan serta CloudFormation mengonfigurasi sumber daya tersebut untuk Anda.

Bila Anda menggunakan CloudFormation, Anda dapat menggunakan kembali template Anda untuk mengatur sumber daya Application Auto Scaling Anda secara konsisten dan berulang kali. Jelaskan sumber daya Anda sekali, lalu sediakan sumber daya yang sama berulang-ulang di beberapa Akun AWS dan Wilayah.

Application Auto Scaling dan template CloudFormation

Untuk menyediakan dan mengonfigurasi sumber daya untuk Application Auto Scaling dan layanan terkait, Anda harus memahami CloudFormation template. Templat adalah file teks dengan format JSON atau YAML. Template ini menjelaskan sumber daya yang ingin Anda sediakan di CloudFormation tumpukan Anda. Jika Anda tidak terbiasa dengan JSON atau YAMM, Anda dapat menggunakan CloudFormation Designer untuk membantu Anda memulai dengan template. CloudFormation Untuk informasi selengkapnya, lihat Apa itu CloudFormation Designer? di Panduan Pengguna AWS CloudFormation .

Saat Anda membuat template tumpukan untuk sumber daya Application Auto Scaling, Anda harus memberikan yang berikut:

- Namespace untuk layanan target (misalnya,**`appstream`**). Lihat AWS::ApplicationAutoScaling::ScalableTargetreferensi untuk mendapatkan ruang nama layanan.
- Dimensi skalabel yang terkait dengan sumber daya target (misalnya,**`appstream:fleet:DesiredCapacity`**). Lihat AWS::ApplicationAutoScaling::ScalableTargetreferensi untuk mendapatkan dimensi yang dapat diskalakan.
- ID sumber daya untuk sumber daya target (misalnya,**`fleet/sample-fleet`**). Lihat AWS::ApplicationAutoScaling::ScalableTargetreferensi untuk informasi tentang sintaks dan contoh sumber IDs daya tertentu.

- Peran terkait layanan untuk sumber daya target (misalnya,**`arn:aws:iam::012345678910:role/aws-service-role/appstream.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_AppStreamFleet`**). Lihat Referensi ARN peran yang ditautkan dengan layanan tabel untuk mendapatkan peran ARNs.

Untuk mempelajari lebih lanjut tentang sumber daya Application Auto Scaling, lihat referensi Application Auto Scaling di AWS CloudFormation Panduan Pengguna.

Contoh cuplikan template

Anda dapat menemukan contoh cuplikan untuk disertakan dalam CloudFormation templat di bagian AWS CloudFormation Panduan Pengguna berikut:

- Untuk contoh kebijakan penskalaan dan tindakan terjadwal, lihat Mengonfigurasi sumber daya Application Auto Scaling dengan. AWS CloudFormation
- Untuk contoh kebijakan penskalaan lainnya, lihat AWS::ApplicationAutoScaling::ScalingPolicy.

Pelajari lebih lanjut tentang CloudFormation

Untuk mempelajari selengkapnya CloudFormation, lihat sumber daya berikut:

- AWS CloudFormation
- AWS CloudFormation Panduan Pengguna
- CloudFormation Referensi API
- Panduan Pengguna Antarmuka Baris Perintah AWS CloudFormation

Penskalaan terjadwal untuk Application Auto Scaling

Dengan penskalaan terjadwal, Anda dapat mengatur penskalaan otomatis untuk aplikasi berdasarkan perubahan beban yang dapat diprediksi dengan membuat tindakan terjadwal yang menambah atau mengurangi kapasitas pada waktu tertentu. Ini memungkinkan Anda menskalakan aplikasi secara proaktif agar sesuai dengan perubahan beban yang dapat diprediksi.

Misalnya, katakanlah Anda mengalami pola lalu lintas mingguan reguler di mana beban meningkat pertengahan minggu dan menurun menjelang akhir minggu. Anda dapat mengonfigurasi jadwal penskalaan di Application Auto Scaling yang sejajar dengan pola ini:

- Pada Rabu pagi, satu tindakan yang dijadwalkan meningkatkan kapasitas dengan meningkatkan kapasitas minimum yang ditetapkan sebelumnya dari target yang dapat diskalakan.
- Pada Jumat malam, tindakan terjadwal lainnya mengurangi kapasitas dengan mengurangi kapasitas maksimum yang ditetapkan sebelumnya dari target yang dapat diskalakan.

Tindakan penskalaan terjadwal ini memungkinkan Anda mengoptimalkan biaya dan kinerja. Aplikasi Anda memiliki kapasitas yang cukup untuk menangani puncak lalu lintas pertengahan minggu, tetapi tidak menyediakan kapasitas yang tidak dibutuhkan secara berlebihan di lain waktu.

Anda dapat menggunakan kebijakan penskalaan dan penskalaan terjadwal bersama-sama untuk mendapatkan manfaat dari pendekatan proaktif dan reaktif untuk penskalaan. Setelah tindakan penskalaan terjadwal berjalan, kebijakan penskalaan dapat terus membuat keputusan tentang apakah akan meningkatkan kapasitas skala lebih lanjut. Ini membantu memastikan bahwa Anda memiliki kapasitas yang cukup untuk menangani beban untuk aplikasi Anda. Meskipun aplikasi Anda menskalakan sesuai dengan permintaan, kapasitas saat ini harus berada di antara kapasitas minimum dan maksimum yang ditetapkan oleh tindakan terjadwal Anda.

Daftar Isi

Cara kerja penskalaan terjadwal untuk Application Auto Scaling

Topik ini menjelaskan cara kerja penskalaan terjadwal dan memperkenalkan pertimbangan utama yang perlu Anda pahami untuk menggunakannya secara efektif.

Daftar Isi

Cara kerjanya

Untuk menggunakan penskalaan terjadwal, buatlah tindakan terjadwal, yang memberi tahu Application Auto Scaling untuk melakukan aktivitas penskalaan pada waktu tertentu. Saat Anda membuat tindakan terjadwal, Anda menentukan target yang dapat diskalakan, waktu aktivitas penskalaan seharusnya terjadi, kapasitas minimum, dan kapasitas maksimum. Anda dapat membuat tindakan terjadwal yang berskala satu kali saja atau skala itu pada jadwal berulang.

Pada waktu yang ditentukan, Application Auto Scaling menskalakan berdasarkan nilai kapasitas baru, dengan membandingkan kapasitas saat ini dengan kapasitas minimum dan maksimum yang ditentukan.

- Jika kapasitas saat ini kurang dari kapasitas minimum yang ditentukan, Application Auto Scaling akan menskalakan naik (meningkatkan kapasitas) ke kapasitas minimum yang ditentukan.
- Jika kapasitas saat ini lebih besar daripada kapasitas maksimum yang ditentukan, Application Auto Scaling menskalakan turun (menurunkan kapasitas) ke kapasitas maksimum yang ditentukan.

Pertimbangan-pertimbangan

Saat Anda membuat tindakan terjadwal, ingatlah hal berikut:

- Tindakan terjadwal menetapkan `MinCapacity` dan apa `MaxCapacity` yang ditentukan oleh tindakan terjadwal pada tanggal dan waktu yang ditentukan. Permintaan secara opsional hanya

dapat mencakup satu dari ukuran ini. Misalnya, Anda dapat membuat tindakan terjadwal hanya dengan kapasitas minimum yang ditentukan. Namun, dalam beberapa kasus, Anda harus memasukkan kedua ukuran untuk memastikan bahwa kapasitas minimum baru tidak lebih besar dari kapasitas maksimum, atau kapasitas maksimum baru tidak kurang dari kapasitas minimum.

- Secara default, jadwal berulang yang Anda tetapkan berada di Coordinated Universal Time (UTC). Anda dapat mengubah zona waktu agar sesuai dengan zona waktu lokal Anda atau zona waktu untuk bagian lain dari jaringan Anda. Saat Anda menentukan zona waktu yang mengamati waktu musim panas, tindakan secara otomatis menyesuaikan Daylight Saving Time (DST). Untuk informasi selengkapnya, lihat Jadwalkan tindakan penskalaan berulang menggunakan Application Auto Scaling.
- Anda dapat mematikan sementara penskalaan terjadwal untuk target yang dapat diskalakan. Ini membantu Anda mencegah tindakan terjadwal agar tidak aktif tanpa harus menghapusnya. Anda kemudian dapat melanjutkan penskalaan terjadwal ketika Anda ingin menggunakannya lagi. Untuk informasi selengkapnya, lihat Menangguhkan dan melanjutkan penskalaan untuk Application Auto Scaling.
- Urutan di mana tindakan terjadwal dijalankan dijamin untuk target skalabel yang sama, tetapi tidak untuk tindakan terjadwal di seluruh target yang dapat diskalakan.
- Untuk menyelesaikan tindakan terjadwal dengan sukses, sumber daya yang ditentukan harus dalam status skalabel dalam layanan target. Jika tidak, permintaan gagal dan mengembalikan pesan kesalahan, misalnya,`Resource Id [ActualResourceId] is not scalable. Reason: The status of all DB instances must be 'available' or 'incompatible-parameters'`.
- Karena sifat terdistribusi dari Application Auto Scaling dan layanan target, penundaan antara waktu tindakan terjadwal dipicu dan waktu layanan target menghormati tindakan penskalaan mungkin beberapa detik. Karena tindakan terjadwal dijalankan dalam urutan yang sesuai dengan urutan ditentukannya tindakan, tindakan-tindakan terjadwal dengan waktu mulai yang berdekatan dapat memakan waktu lebih lama untuk dijalankan.

Perintah yang umum digunakan untuk pembuatan, pengelolaan, dan penghapusan tindakan terjadwal

Perintah yang umum digunakan untuk bekerja dengan penskalaan jadwal meliputi:

- register-scalable-targetuntuk mendaftarkan AWS atau menyesuaikan sumber daya sebagai target yang dapat diskalakan (sumber daya yang dapat diskalakan oleh Application Auto Scaling), dan untuk menangguhkan dan melanjutkan penskalaan.
- put-scheduled-action untuk menambah atau mengubah tindakan terjadwal untuk target yang dapat diskalakan yang ada.
- describe-scaling-activitiesuntuk mengembalikan informasi tentang aktivitas penskalaan di suatu AWS Wilayah.
- describe-scheduled-actionsuntuk mengembalikan informasi tentang tindakan terjadwal di suatu AWS Wilayah.
- delete-scheduled-action untuk menghapus tindakan terjadwal.

Sumber daya terkait

Untuk contoh mendetail tentang penggunaan penskalaan terjadwal, lihat posting blog Menjadwalkan Konkurensi yang AWS Lambda Disediakan untuk penggunaan puncak berulang di Blog Komputasi. AWS

Untuk informasi tentang membuat tindakan terjadwal untuk grup Auto Scaling, lihat Penskalaan terjadwal untuk Penskalaan Otomatis Amazon EC2 Auto Scaling di Panduan Pengguna Amazon EC2 Auto Scaling.

Batasan

Berikut ini adalah keterbatasan saat menggunakan penskalaan terjadwal:

- Nama-nama tindakan terjadwal harus unik per target yang dapat diskalakan.
- Application Auto Scaling tidak memberikan presisi tingkat kedua dalam ekspresi jadwal. Resolusi terbaik yang menggunakan ekspresi cron adalah 1 menit.
- Target yang dapat diskalakan tidak bisa menjadi klaster Amazon MSK. Penskalaan terjadwal tidak didukung untuk Amazon MSK.
- Akses konsol untuk melihat, menambah, memperbarui, atau menghapus tindakan terjadwal pada sumber daya yang dapat diskalakan bergantung pada sumber daya yang Anda gunakan. Untuk informasi selengkapnya, lihat Layanan AWS yang dapat Anda gunakan dengan Application Auto Scaling.

Buat tindakan terjadwal untuk Application Auto Scaling menggunakan AWS CLI

Contoh berikut menunjukkan cara membuat tindakan terjadwal menggunakan AWS CLI put-scheduled-actionperintah. Saat menentukan kapasitas baru, Anda dapat menentukan kapasitas minimum, kapasitas maksimum, atau keduanya.

Contoh-contoh ini menggunakan target yang dapat diskalakan untuk beberapa layanan yang terintegrasi dengan Application Auto Scaling. Untuk menggunakan target skalabel yang berbeda, tentukan namespace di, dimensi yang dapat diskalakan di `--service-namespace--scalable-dimension`, dan ID sumber dayanya di. `--resource-id`

Saat menggunakan AWS CLI, ingatlah bahwa perintah Anda berjalan di AWS Region konfigurasi untuk profil Anda. Jika Anda ingin menjalankan perintah di Wilayah yang berbeda, ubah Wilayah default untuk profil Anda, atau gunakan parameter `--region` bersama perintah tersebut.

Contoh

- Buat tindakan terjadwal yang hanya terjadi sekali
- Buat tindakan terjadwal yang berjalan pada interval berulang
- Buat tindakan terjadwal yang berjalan pada jadwal berulang
- Buat tindakan terjadwal satu kali yang menentukan zona waktu
- Buat tindakan terjadwal berulang yang menentukan zona waktu

Buat tindakan terjadwal yang hanya terjadi sekali

Untuk secara otomatis menskalakan target yang dapat diskalakan Anda hanya satu kali, pada tanggal dan waktu tertentu, gunakan opsi `--schedule "at(yyyy-mm-ddThh:mm:ss)"`.

Example Contoh: Untuk menskalakan naik hanya satu kali

Berikut adalah contoh pembuatan tindakan terjadwal untuk melakukan penskalaan naik terhadap kapasitas pada tanggal dan waktu tertentu.

Pada tanggal dan waktu yang ditentukan untuk `--schedule` (pukul 22.00 UTC pada 31 Maret 2021), jika nilai yang ditentukan untuk `MinCapacity` melebihi kapasitas saat ini, Application Auto Scaling menskalakan naik ke `MinCapacity`.

Linux, macOS, atau Unix

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource \
  --scalable-dimension custom-resource:ResourceType:Property \
  --resource-id file://~/custom-resource-id.txt \
  --scheduled-action-name scale-out \
  --schedule "at(2021-03-31T22:00:00)" \
  --scalable-target-action MinCapacity=3
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource ^
  --scalable-dimension custom-resource:ResourceType:Property ^
  --resource-id file://~/custom-resource-id.txt ^
  --scheduled-action-name scale-out ^
  --schedule "at(2021-03-31T22:00:00)" ^
  --scalable-target-action MinCapacity=3
```

Ketika tindakan terjadwal ini berjalan, jika kapasitas maksimum kurang dari nilai yang ditentukan untuk kapasitas minimum, Anda harus menentukan kapasitas minimum dan maksimum baru, dan bukan hanya kapasitas minimum.

Example Contoh: Untuk menskalakan turun hanya satu kali

Berikut adalah contoh pembuatan tindakan terjadwal untuk melakukan penskalaan turun terhadap kapasitas pada tanggal dan waktu tertentu.

Pada tanggal dan waktu yang ditentukan untuk `--schedule` (pukul 22.30 UTC pada 31 Maret 2021), jika nilai yang ditentukan untuk `MaxCapacity` berada di bawah kapasitas saat ini, Application Auto Scaling menskalakan turun ke `MaxCapacity`.

Linux, macOS, atau Unix

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource \
  --scalable-dimension custom-resource:ResourceType:Property \
  --resource-id file://~/custom-resource-id.txt \
  --scheduled-action-name scale-in \
  --schedule "at(2021-03-31T22:30:00)" \
  --scalable-target-action MinCapacity=0,MaxCapacity=0
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource ^
  --scalable-dimension custom-resource:ResourceType:Property ^
```

```
  --resource-id file://~/custom-resource-id.txt ^
  --scheduled-action-name scale-in ^
  --schedule "at(2021-03-31T22:30:00)" ^
  --scalable-target-action MinCapacity=0,MaxCapacity=0
```

Buat tindakan terjadwal yang berjalan pada interval berulang

Untuk menjadwalkan penskalaan pada interval berulang, gunakan opsi `--schedule "rate(value unit)"`. Nilai harus berupa bilangan bulat positif. Unit dapat `minute`, `minutes`, `hour`, `hours`, `day`, atau `days`. Untuk informasi selengkapnya, lihat Nilai ekspresi di Panduan EventBridge Pengguna Amazon.

Berikut ini adalah contoh tindakan terjadwal yang menggunakan ekspresi rate.

Pada jadwal yang ditentukan (setiap 5 jam mulai tanggal 30 Januari 2021 pukul 12.00 UTC dan berakhir pada tanggal 31 Januari 2021 pukul 22.00 UTC), jika nilai yang ditentukan untuk `MinCapacity` berada di atas kapasitas saat ini, Application Auto Scaling menskalakan keluar ke `MinCapacity`. Jika nilai yang ditentukan untuk `MaxCapacity` berada di bawah kapasitas saat ini, Application Auto Scaling menskalakan kedalam ke `MaxCapacity`.

Linux, macOS, atau Unix

```
aws application-autoscaling put-scheduled-action --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --scheduled-action-name my-recurring-action \
  --schedule "rate(5 hours)" \
  --start-time 2021-01-30T12:00:00 \
  --end-time 2021-01-31T22:00:00 \
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
  --resource-id service/my-cluster/my-service ^
  --scheduled-action-name my-recurring-action ^
  --schedule "rate(5 hours)" ^
  --start-time 2021-01-30T12:00:00 ^
  --end-time 2021-01-31T22:00:00 ^
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

Buat tindakan terjadwal yang berjalan pada jadwal berulang

Untuk menjadwalkan penskalaan pada jadwal berulang, gunakan opsi `--schedule "cron(fields)"`. Untuk informasi selengkapnya, lihat Jadwalkan tindakan penskalaan berulang menggunakan Application Auto Scaling.

Berikut ini adalah contoh tindakan terjadwal yang menggunakan ekspresi cron.

Pada jadwal yang ditentukan (setiap hari pukul 09.00 UTC), jika nilai yang ditentukan untuk `MinCapacity` berada di atas kapasitas saat ini, Application Auto Scaling menskalakan naik ke `MinCapacity`. Jika nilai yang ditentukan untuk `MaxCapacity` berada di bawah kapasitas saat ini, Application Auto Scaling menskalakan kedalam ke `MaxCapacity`.

Linux, macOS, atau Unix

```
aws application-autoscaling put-scheduled-action --service-namespace appstream \
  --scalable-dimension appstream:fleet:DesiredCapacity \
  --resource-id fleet/sample-fleet \
  --scheduled-action-name my-recurring-action \
  --schedule "cron(0 9 * * ? *)" \
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace appstream ^
  --scalable-dimension appstream:fleet:DesiredCapacity ^
  --resource-id fleet/sample-fleet ^
  --scheduled-action-name my-recurring-action ^
  --schedule "cron(0 9 * * ? *)" ^
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

Buat tindakan terjadwal satu kali yang menentukan zona waktu

Tindakan terjadwal diatur ke zona waktu UTC secara default. Untuk menentukan zona waktu yang berbeda, sertakan opsi `--timezone` dan tentukan nama resmi untuk zona waktu (misalnya `America/New_York`). Untuk informasi selengkapnya https://www.joda.org/joda-time/timezones.html, lihat, yang menyediakan informasi tentang zona waktu IANA yang didukung saat menelepon put-scheduled-action.

Berikut adalah contoh yang menggunakan opsi `--timezone` saat membuat tindakan terjadwal untuk menskalakan kapasitas pada tanggal dan waktu tertentu.

Pada tanggal dan waktu yang ditentukan untuk `--schedule` (pukul 17.00 waktu setempat pada 31 Januari 2021), jika nilai yang ditentukan untuk `MinCapacity` melebihi kapasitas saat ini, Application Auto Scaling menskalakan naik ke `MinCapacity`. Jika nilai yang ditentukan untuk `MaxCapacity` berada di bawah kapasitas saat ini, Application Auto Scaling menskalakan kedalam ke `MaxCapacity`.

Linux, macOS, atau Unix

```
aws application-autoscaling put-scheduled-action --service-namespace comprehend \
  --scalable-dimension comprehend:document-classifier-endpoint:DesiredInferenceUnits \
  --resource-id arn:aws:comprehend:us-west-2:123456789012:document-classifier-endpoint/
EXAMPLE \
  --scheduled-action-name  my-one-time-action \
  --schedule "at(2021-01-31T17:00:00)" --timezone "America/New_York" \
  --scalable-target-action MinCapacity=1,MaxCapacity=3
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace comprehend ^
  --scalable-dimension comprehend:document-classifier-endpoint:DesiredInferenceUnits ^
  --resource-id arn:aws:comprehend:us-west-2:123456789012:document-classifier-endpoint/
EXAMPLE ^
  --scheduled-action-name  my-one-time-action ^
  --schedule "at(2021-01-31T17:00:00)" --timezone "America/New_York" ^
  --scalable-target-action MinCapacity=1,MaxCapacity=3
```

Buat tindakan terjadwal berulang yang menentukan zona waktu

Berikut adalah contoh yang menggunakan opsi `--timezone` saat membuat tindakan terjadwal berulang untuk menskalakan kapasitas. Untuk informasi selengkapnya, lihat Jadwalkan tindakan penskalaan berulang menggunakan Application Auto Scaling.

Pada jadwal yang ditentukan (setiap Senin hingga Jumat pukul 18.00 waktu setempat), jika nilai yang ditentukan untuk `MinCapacity` berada di atas kapasitas saat ini, Application Auto Scaling menskalakan naik ke `MinCapacity`. Jika nilai yang ditentukan untuk `MaxCapacity` berada di bawah kapasitas saat ini, Application Auto Scaling menskalakan kedalam ke `MaxCapacity`.

Linux, macOS, atau Unix

```
aws application-autoscaling put-scheduled-action --service-namespace lambda \
```

```
  --scalable-dimension lambda:function:ProvisionedConcurrency \
  --resource-id function:my-function:BLUE \
  --scheduled-action-name my-recurring-action \
  --schedule "cron(0 18 ? * MON-FRI *)" --timezone "Etc/GMT+9" \
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace lambda ^
  --scalable-dimension lambda:function:ProvisionedConcurrency ^
  --resource-id function:my-function:BLUE ^
  --scheduled-action-name my-recurring-action ^
  --schedule "cron(0 18 ? * MON-FRI *)" --timezone "Etc/GMT+9" ^
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

Jelaskan penskalaan terjadwal untuk Application Auto Scaling menggunakan AWS CLI

Contoh AWS CLI perintah ini menjelaskan aktivitas penskalaan dan tindakan terjadwal menggunakan sumber daya dari layanan yang terintegrasi dengan Application Auto Scaling. Untuk target skalabel yang berbeda, tentukan namespace di`--service-namespace`, dimensi yang dapat diskalakan di`--scalable-dimension`, dan ID sumber dayanya di. `--resource-id`

Saat menggunakan AWS CLI, ingatlah bahwa perintah Anda berjalan di AWS Region konfigurasi untuk profil Anda. Jika Anda ingin menjalankan perintah di Wilayah yang berbeda, ubah Wilayah default untuk profil Anda, atau gunakan parameter `--region` bersama perintah tersebut.

Contoh

- Jelaskan aktivitas penskalaan untuk layanan
- Jelaskan tindakan terjadwal untuk suatu layanan
- Jelaskan tindakan terjadwal untuk target yang dapat diskalakan

Jelaskan aktivitas penskalaan untuk layanan

Untuk melihat aktivitas penskalaan untuk semua target yang dapat diskalakan dalam namespace layanan tertentu, gunakan perintah describe-scaling-activities.

Contoh berikut mengambil aktivitas penskalaan yang terkait dengan namespace layanan dynamodb.

Linux, macOS, atau Unix

```
aws application-autoscaling describe-scaling-activities --service-namespace dynamodb
```

Windows

```
aws application-autoscaling describe-scaling-activities --service-namespace dynamodb
```

Output

Jika perintah berhasil, ia mengembalikan output yang mirip dengan berikut ini.

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 10.",
            "ResourceId": "table/my-table",
            "ActivityId": "4d1308c0-bbcf-4514-a673-b0220ae38547",
            "StartTime": 1561574415.086,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574449.51,
            "Cause": "maximum capacity was set to 10",
            "StatusMessage": "Successfully set write capacity units to 10. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 5 and max capacity to 10",
            "ResourceId": "table/my-table",
            "ActivityId": "f2b7847b-721d-4e01-8ef0-0c8d3bacc1c7",
            "StartTime": 1561574414.644,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-second-scheduled-action was triggered",
            "StatusMessage": "Successfully set min capacity to 5 and max capacity to
 10",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 15.",
```

```
            "ResourceId": "table/my-table",
            "ActivityId": "d8ea4de6-9eaa-499f-b466-2cc5e681ba8b",
            "StartTime": 1561574108.904,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574140.255,
            "Cause": "minimum capacity was set to 15",
            "StatusMessage": "Successfully set write capacity units to 15. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 15 and max capacity to 20",
            "ResourceId": "table/my-table",
            "ActivityId": "3250fd06-6940-4e8e-bb1f-d494db7554d2",
            "StartTime": 1561574108.512,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-first-scheduled-action was triggered",
            "StatusMessage": "Successfully set min capacity to 15 and max capacity to
 20",
            "StatusCode": "Successful"
        }
    ]
}
```

Untuk mengubah perintah ini sehingga mengambil aktivitas penskalaan hanya untuk satu target yang dapat diskalakan Anda, tambahkan opsi `--resource-id`.

Jelaskan tindakan terjadwal untuk suatu layanan

Untuk menjelaskan tindakan terjadwal untuk semua target yang dapat diskalakan dalam namespace layanan tertentu, gunakan perintah describe-scheduled-actions.

Contoh berikut mengambil tindakan terjadwal yang terkait dengan namespace layanan `ec2`.

Linux, macOS, atau Unix

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2
```

Windows

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2
```

Output

Jika perintah berhasil, ia mengembalikan output yang mirip dengan berikut ini.

```
{
    "ScheduledActions": [
        {
            "ScheduledActionName": "my-one-time-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:493a6261-fbb9-432d-855d-3c302c14bdb9:resource/ec2/
spot-fleet-request/sfr-107dc873-0802-4402-a901-37294EXAMPLE:scheduledActionName/my-one-
time-action",
            "ServiceNamespace": "ec2",
            "Schedule": "at(2021-01-31T17:00:00)",
            "Timezone": "America/New_York",
            "ResourceId": "spot-fleet-request/sfr-107dc873-0802-4402-
a901-37294EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "ScalableTargetAction": {
                "MaxCapacity": 1
            },
            "CreationTime": 1607454792.331
        },
        {
            "ScheduledActionName": "my-recurring-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:493a6261-fbb9-432d-855d-3c302c14bdb9:resource/ec2/
spot-fleet-request/sfr-107dc873-0802-4402-a901-37294EXAMPLE:scheduledActionName/my-
recurring-action",
            "ServiceNamespace": "ec2",
            "Schedule": "rate(5 minutes)",
            "ResourceId": "spot-fleet-request/sfr-107dc873-0802-4402-
a901-37294EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "StartTime": 1604059200.0,
            "EndTime": 1612130400.0,
            "ScalableTargetAction": {
                "MinCapacity": 3,
                "MaxCapacity": 10
            },
            "CreationTime": 1607454949.719
        },
        {
            "ScheduledActionName": "my-one-time-action",
```

```
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:4bce34c7-bb81-4ecf-b776-5c726efb1567:resource/ec2/
spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE:scheduledActionName/my-one-
time-action",
            "ServiceNamespace": "ec2",
            "Schedule": "at(2020-12-08T9:36:00)",
            "Timezone": "America/New_York",
            "ResourceId": "spot-fleet-request/sfr-40edeb7b-9ae7-44be-
bef2-5c4c8EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "ScalableTargetAction": {
                "MinCapacity": 1,
                "MaxCapacity": 3
            },
            "CreationTime": 1607456031.391
        }
    ]
}
```

Jelaskan tindakan terjadwal untuk target yang dapat diskalakan

Untuk mengambil informasi tentang tindakan terjadwal untuk target yang dapat diskalakan tertentu, tambahkan opsi `--resource-id` ketika menjelaskan tindakan terjadwal menggunakan perintah describe-scheduled-actions.

Jika Anda menyertakan opsi `--scheduled-action-names` dan menentukan nama tindakan terjadwal sebagai nilainya, perintah mengembalikan hanya tindakan terjadwal yang namanya cocok, seperti yang ditunjukkan pada contoh berikut.

Linux, macOS, atau Unix

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2 \
  --resource-id spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE \
  --scheduled-action-names my-one-time-action
```

Windows

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2 ^
  --resource-id spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE ^
  --scheduled-action-names my-one-time-action
```

Output

Jika perintah berhasil, ia mengembalikan output yang mirip dengan berikut ini. Jika Anda memberikan lebih dari satu nilai untuk`--scheduled-action-names`, output mencakup semua tindakan terjadwal yang namanya cocok.


```
{
    "ScheduledActions": [
        {
            "ScheduledActionName": "my-one-time-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:4bce34c7-bb81-4ecf-b776-5c726efb1567:resource/ec2/
spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE:scheduledActionName/my-one-
time-action",
            "ServiceNamespace": "ec2",
            "Schedule": "at(2020-12-08T9:36:00)",
            "Timezone": "America/New_York",
            "ResourceId": "spot-fleet-request/sfr-40edeb7b-9ae7-44be-
bef2-5c4c8EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "ScalableTargetAction": {
                "MinCapacity": 1,
                "MaxCapacity": 3
            },
            "CreationTime": 1607456031.391
        }
    ]
}
```


Jadwalkan tindakan penskalaan berulang menggunakan Application Auto Scaling

> **Important**
>
> Untuk bantuan dengan ekspresi cron untuk Amazon EC2 Auto Scaling, lihat topik Jadwal berulang di Panduan Pengguna Amazon EC2 Auto Scaling. Dengan Amazon EC2 Auto Scaling, Anda menggunakan sintaks cron tradisional alih-alih sintaks cron khusus yang digunakan Application Auto Scaling.

Anda dapat membuat tindakan terjadwal yang berjalan pada jadwal berulang menggunakan ekspresi cron.

Untuk membuat jadwal berulang, tentukan ekspresi cron dan zona waktu untuk menjelaskan kapan tindakan terjadwal itu akan berulang. Nilai zona waktu yang didukung adalah nama kanonik dari zona waktu IANA yang didukung oleh Joda-Time (seperti atau). `Etc/GMT+9` `Pacific/Tahiti` Anda dapat secara opsional menentukan tanggal dan waktu untuk waktu mulai, waktu akhir, atau keduanya. Untuk contoh perintah yang menggunakan AWS CLI untuk membuat tindakan terjadwal, lihatBuat tindakan terjadwal berulang yang menentukan zona waktu.

Format ekspresi cron yang didukung terdiri dari enam bidang yang dipisahkan oleh spasi putih: [Minutes] [Hours] [Day_of_month] [Month] [Day_of_week] [Year]. Misalnya, ekspresi cron `30 6 ? * MON *` mengonfigurasi tindakan terjadwal yang berulang setiap hari Senin pukul 6:30 pagi. Tanda bintang digunakan sebagai wildcard untuk mencocokkan semua nilai untuk bidang.

Untuk informasi selengkapnya tentang sintaks cron untuk tindakan terjadwal Application Auto Scaling, lihat Referensi ekspresi cron di Panduan Pengguna Amazon. EventBridge

Saat Anda membuat jadwal berulang, pilih waktu mulai dan akhir Anda dengan hati-hati. Ingatlah hal-hal berikut ini:

- Jika Anda menentukan waktu mulai, Application Auto Scaling melakukan tindakan saat ini, dan kemudian melakukan tindakan berdasarkan pengulangan yang ditentukan.
- Jika Anda menentukan waktu akhir, tindakan berhenti berulang setelah waktu ini. Application Auto Scaling tidak melacak nilai sebelumnya dan kembali ke nilai sebelumnya setelah waktu berakhir.
- Waktu mulai dan waktu akhir harus diatur dalam UTC saat Anda menggunakan AWS CLI atau AWS SDKs untuk membuat atau memperbarui tindakan terjadwal.

Contoh

Anda dapat merujuk ke tabel berikut saat membuat jadwal berulang untuk target Application Auto Scaling yang dapat diskalakan. Contoh berikut adalah sintaks yang benar untuk menggunakan Application Auto Scaling untuk membuat atau memperbarui tindakan terjadwal.

Menit	Jam	Hari dalam sebulan	Bulan	Hari dalam seminggu	Tahun	Arti
0	10	*	*	?	*	Jalankan pada pukul

Menit	Jam	Hari dalam sebulan	Bulan	Hari dalam seminggu	Tahun	Arti
						10:00 pagi (UTC) setiap hari
15	12	*	*	?	*	Jalankan pada pukul 12.15 (UTC) setiap hari
0	18	?	*	MON-FRI	*	Jalankan pada pukul 18.00 (UTC) setiap Senin hingga Jumat
0	8	1	*	?	*	Jalankan pukul 08.00 (UTC) pada hari pertama setiap bulan
0/15	*	*	*	?	*	Jalankan setiap 15 menit

Menit	Jam	Hari dalam sebulan	Bulan	Hari dalam seminggu	Tahun	Arti
0/10	*	?	*	MON-FRI	*	Jalankan setiap 10 menit Senin hingga Jumat
0/5	8-17	?	*	MON-FRI	*	Jalankan setiap 5 menit Senin hingga Jumat antara pukul 08.00 dan 17.55 (UTC)

Pengecualian

Anda juga dapat membuat ekspresi cron dengan nilai string yang berisi tujuh bidang. Dalam hal ini, Anda dapat menggunakan tiga bidang pertama untuk menentukan waktu kapan tindakan terjadwal harus dijalankan, termasuk detik. Ekspresi cron lengkap memiliki bidang yang dipisahkan spasi berikut: [Seconds] [Minutes] [Hours] [Day_of_month] [Month] [Day_of_week] [Year]. Namun, pendekatan ini tidak menjamin bahwa tindakan terjadwal akan berjalan pada detik yang tepat yang Anda tentukan. Selain itu, beberapa konsol layanan mungkin tidak mendukung bidang detik dalam ekspresi cron.

Matikan penskalaan terjadwal untuk target yang dapat diskalakan

Anda dapat mematikan sementara penskalaan terjadwal tanpa menghapus tindakan terjadwal. Untuk informasi selengkapnya, lihat Menangguhkan dan melanjutkan penskalaan untuk Application Auto Scaling.

Untuk menangguhkan penskalaan terjadwal

Menangguhkan penskalaan terjadwal pada target yang dapat diskalakan dengan menggunakan perintah register-scalable-target dengan opsi `--suspended-state`, dan menentukan `true` sebagai nilai dari atribut `ScheduledScalingSuspended`, seperti yang ditunjukkan dalam contoh berikut.

Linux, macOS, atau Unix

```
aws application-autoscaling register-scalable-target --service-namespace rds \
  --scalable-dimension rds:cluster:ReadReplicaCount --resource-id cluster:my-db-cluster
 \
  --suspended-state '{"ScheduledScalingSuspended": true}'
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace rds ^
  --scalable-dimension rds:cluster:ReadReplicaCount --resource-id cluster:my-db-cluster
 ^
  --suspended-state "{\"ScheduledScalingSuspended\": true}"
```

Output

Jika perintah berhasil, ia mengembalikan ARN dari target yang dapat diskalakan. Berikut ini adalah output contoh.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Untuk melanjutkan penskalaan terjadwal

Untuk melanjutkan penskalaan terjadwal, jalankan register-scalable-target perintah lagi, tentukan `false` sebagai nilai untuk. `ScheduledScalingSuspended`

Menghapus tindakan terjadwal untuk Application Auto Scaling menggunakan AWS CLI

Setelah selesai dengan tindakan terjadwal, Anda dapat menghapusnya.

Untuk menghapus tindakan terjadwal

Gunakan perintah delete-scheduled-action. Jika berhasil, perintah ini tidak mengembalikan output apa pun.

Linux, macOS, atau Unix

```
aws application-autoscaling delete-scheduled-action \
  --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-37294EXAMPLE \
  --scheduled-action-name my-recurring-action
```

Windows

```
aws application-autoscaling delete-scheduled-action ^
  --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-37294EXAMPLE ^
  --scheduled-action-name my-recurring-action
```

Untuk membatalkan pendaftaran target yang dapat diskalakan

Jika Anda juga selesai dengan target yang dapat diskalakan, Anda dapat membatalkan pendaftarannya. Gunakan perintah berikut deregister-scalable-target. Jika ada kebijakan penskalaan atau tindakan terjadwal yang belum dihapus, mereka akan dihapus oleh perintah ini. Jika berhasil, perintah ini tidak mengembalikan output apa pun.

Linux, macOS, atau Unix

```
aws application-autoscaling deregister-scalable-target \
  --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-37294EXAMPLE
```

Windows

```
aws application-autoscaling deregister-scalable-target ^
  --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-37294EXAMPLE
```

Kebijakan penskalaan pelacakan target untuk Application Auto Scaling

Kebijakan penskalaan pelacakan target secara otomatis menskalakan aplikasi Anda berdasarkan nilai metrik target. Hal ini memungkinkan aplikasi Anda untuk mempertahankan kinerja optimal dan efisiensi biaya tanpa intervensi manual.

Dengan pelacakan target, Anda memilih metrik dan nilai target untuk mewakili tingkat pemanfaatan atau throughput rata-rata yang ideal untuk aplikasi Anda. Application Auto Scaling membuat dan mengelola CloudWatch alarm yang memicu peristiwa penskalaan saat metrik menyimpang dari target. Ini mirip dengan bagaimana termostat mempertahankan suhu target.

Misalnya, katakanlah Anda saat ini memiliki aplikasi yang berjalan di Spot Fleet, dan Anda ingin pemanfaatan CPU armada tetap sekitar 50 persen saat beban pada aplikasi berubah. Ini memberi Anda kapasitas ekstra untuk menangani lonjakan lalu lintas tanpa mempertahankan jumlah berlebih dari sumber daya yang tidak aktif.

Anda dapat memenuhi kebutuhan ini dengan membuat kebijakan penskalaan pelacakan target yang menargetkan pemanfaatan CPU rata-rata 50 persen. Kemudian, Application Auto Scaling akan keluar (meningkatkan kapasitas) ketika CPU melebihi 50 persen untuk menangani peningkatan beban. Ini akan menskalakan (penurunan kapasitas) ketika CPU turun di bawah 50 persen untuk mengoptimalkan biaya selama periode pemanfaatan rendah.

Kebijakan pelacakan target menghilangkan kebutuhan untuk menentukan CloudWatch alarm dan penyesuaian penskalaan secara manual. Application Auto Scaling menangani ini secara otomatis berdasarkan target yang Anda tetapkan.

Anda dapat mendasarkan kebijakan pelacakan target pada metrik yang telah ditentukan atau kustom:

- Metrik yang telah ditentukan sebelumnya —Metrik yang disediakan oleh Application Auto Scaling seperti penggunaan CPU rata-rata atau jumlah permintaan rata-rata per target.
- Metrik kustom —Anda dapat menggunakan matematika metrik untuk menggabungkan metrik, memanfaatkan metrik yang ada, atau menggunakan metrik kustom Anda sendiri yang dipublikasikan. CloudWatch

Pilih metrik yang berubah berbanding terbalik dengan perubahan kapasitas target Anda yang dapat diskalakan. Jadi jika Anda menggandakan kapasitas, metriknya berkurang 50 persen. Hal ini memungkinkan data metrik untuk secara akurat memicu peristiwa penskalaan proporsional.

Daftar Isi

Cara kerja penskalaan pelacakan target untuk Application Auto Scaling

Topik ini menjelaskan cara kerja penskalaan pelacakan target dan memperkenalkan elemen kunci dari kebijakan penskalaan pelacakan target.

Daftar Isi

Cara kerjanya

Untuk menggunakan penskalaan pelacakan target, Anda membuat kebijakan penskalaan pelacakan target dan menentukan hal berikut:

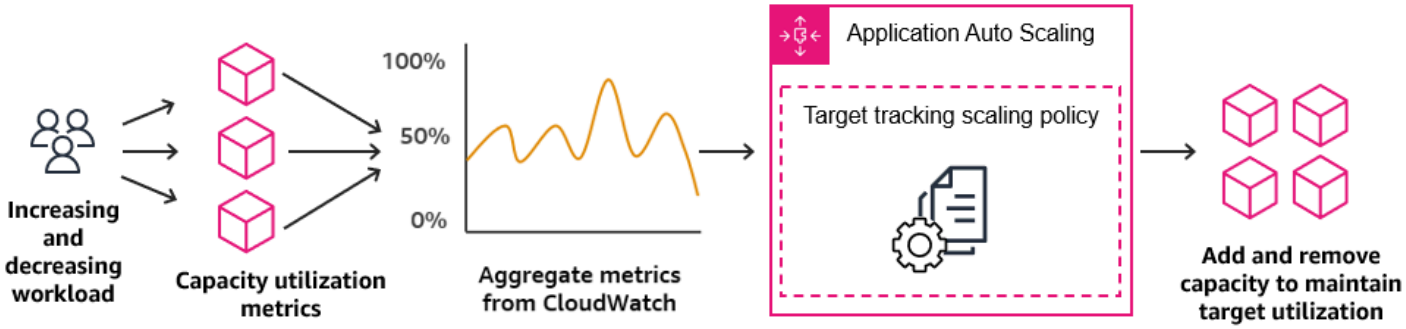
- Metrik CloudWatch —Metrik untuk dilacak, seperti pemanfaatan CPU rata-rata atau jumlah permintaan rata-rata per target.
- Nilai target — Nilai target untuk metrik, seperti 50 persen pemanfaatan CPU atau 1000 permintaan per target per menit.

Application Auto Scaling membuat dan mengelola CloudWatch alarm yang menjalankan kebijakan penskalaan dan menghitung penyesuaian penskalaan berdasarkan metrik dan nilai target. Ini menambah dan menghapus kapasitas seperti yang diperlukan untuk menjaga metrik pada, atau dekat dengan, nilai target yang ditentukan.

Ketika metrik berada di atas nilai target, Application Auto Scaling keluar dengan menambahkan kapasitas untuk mengurangi perbedaan antara nilai metrik dan nilai target. Ketika metrik berada di bawah nilai target, Application Auto Scaling masuk dengan menghapus kapasitas.

Aktivitas penskalaan dilakukan dengan periode cooldown di antara mereka untuk mencegah fluktuasi kapasitas yang cepat. Anda dapat secara opsional mengonfigurasi periode cooldown untuk kebijakan penskalaan Anda.

Diagram berikut menunjukkan gambaran umum tentang cara kerja kebijakan penskalaan pelacakan target saat penyiapan selesai.



Perhatikan bahwa kebijakan penskalaan pelacakan target lebih agresif dalam menambah kapasitas saat pemanfaatan meningkat daripada menghilangkan kapasitas saat pemanfaatan menurun. Misalnya, jika metrik yang ditentukan kebijakan mencapai nilai targetnya, kebijakan tersebut mengasumsikan bahwa aplikasi Anda sudah banyak dimuat. Jadi metrik akan merespons dengan

menambahkan kapasitas proporsional terhadap nilai metrik secepat mungkin. Semakin tinggi metrik, semakin banyak kapasitas yang ditambahkan.

Ketika metrik jatuh di bawah nilai target, kebijakan tidak akan menskalakan jika menghitung bahwa menghapus unit kapasitas minimum kemungkinan akan membawa metrik kembali di atas nilai target. Dalam hal ini, memperlambat penskalaan dengan menghapus kapasitas hanya ketika pemanfaatan melewati ambang batas yang cukup jauh di bawah nilai target (biasanya lebih dari 10% lebih rendah) agar pemanfaatan dianggap melambat. Tujuan dari perilaku yang lebih konservatif ini adalah untuk memastikan bahwa menghapus kapasitas hanya terjadi ketika aplikasi tidak lagi mengalami permintaan pada tingkat yang sama tinggi seperti sebelumnya.

Pilih metrik

Anda dapat membuat kebijakan penskalaan pelacakan target dengan metrik yang telah ditentukan sebelumnya atau metrik khusus.

Saat membuat kebijakan penskalaan pelacakan target dengan tipe metrik yang telah ditentukan sebelumnya, Anda memilih satu metrik dari daftar metrik yang telah ditentukan sebelumnya. Metrik yang telah ditentukan untuk kebijakan penskalaan pelacakan target

Ingatlah hal-hal berikut ini saat memilih metrik:

- Tidak semua metrik khusus berfungsi untuk pelacakan target. Metrik harus berupa metrik pemanfaatan yang valid dan menjelaskan seberapa sibuk target yang dapat diskalakan. Nilai metrik harus meningkat atau menurun secara proporsional sesuai kapasitas target yang dapat diskalakan, sehingga data metrik dapat digunakan untuk skala proporsional pada target yang dapat diskalakan.
- Untuk menggunakan metrik `ALBRequestCountPerTarget`, Anda harus menentukan parameter `ResourceLabel` guna mengidentifikasi grup target yang terkait dengan metrik.
- Ketika metrik memancarkan nilai 0 nyata ke CloudWatch (misalnya,`ALBRequestCountPerTarget`), Application Auto Scaling dapat menskalakan ke 0 ketika tidak ada lalu lintas ke aplikasi Anda untuk jangka waktu yang berkelanjutan. Untuk memiliki skala target yang dapat diskalakan Anda ke 0 ketika tidak ada permintaan yang dirutekan, kapasitas minimum target yang dapat diskalakan harus diatur ke 0.
- Alih-alih menerbitkan metrik baru untuk digunakan dalam kebijakan penskalaan, Anda dapat menggunakan matematika metrik untuk menggabungkan metrik yang ada. Untuk informasi selengkapnya, lihat Membuat kebijakan penskalaan pelacakan target untuk Application Auto Scaling menggunakan matematika metrik.

- Untuk melihat apakah layanan yang Anda gunakan mendukung penetapan metrik kustom di konsol layanan, lihat dokumentasi untuk layanan tersebut.
- Kami menyarankan Anda menggunakan metrik yang tersedia pada interval satu menit untuk membantu Anda menskalakan lebih cepat dalam menanggapi perubahan pemanfaatan. Pelacakan target akan mengevaluasi metrik yang dikumpulkan pada perincian satu menit untuk semua metrik dan metrik khusus yang telah ditentukan sebelumnya, tetapi metrik yang mendasarinya mungkin lebih jarang mempublikasikan data. Misalnya, semua metrik Amazon EC2 dikirim dalam interval lima menit secara default, tetapi dapat dikonfigurasi hingga satu menit (dikenal sebagai pemantauan terperinci). Pilihan ini terserah layanan individu. Sebagian besar mencoba menggunakan interval sekecil mungkin.

Tentukan nilai target

Saat membuat kebijakan penskalaan pelacakan target, Anda harus menentukan nilai target. Nilai target mewakili pemanfaatan atau throughput rata-rata optimal untuk aplikasi Anda. Untuk menggunakan biaya sumber daya secara efisien, tetapkan nilai target setinggi mungkin dengan buffer yang masuk akal untuk peningkatan lalu lintas yang tidak terduga. Ketika aplikasi Anda diskalakan secara optimal untuk arus lalu lintas normal, nilai metrik sebenarnya harus berada pada atau tepat di bawah nilai target.

Jika kebijakan penskalaan didasarkan pada throughput, seperti jumlah permintaan per target untuk Application Load Balancer, I/O jaringan, atau metrik hitungan lainnya, nilai target mewakili throughput rata-rata optimal dari satu entitas (seperti target tunggal grup target Application Load Balancer Anda), selama satu menit.

Tentukan periode cooldown

Anda dapat secara opsional menentukan periode cooldown dalam kebijakan penskalaan pelacakan target Anda.

Periode cooldown menentukan jumlah waktu kebijakan penskalaan menunggu aktivitas penskalaan sebelumnya berlaku.

Ada dua jenis periode cooldown:

- Dengan periode jeda pakai untuk pengecilan skala, tujuannya adalah untuk menskalakan secara terus-menerus (tetapi tidak berlebihan). Setelah Application Auto Scaling berhasil menskalakan menggunakan kebijakan penskalaan, Application Auto Scaling mulai menghitung

waktu cooldown. Kebijakan penskalaan tidak akan meningkatkan kapasitas yang diinginkan lagi kecuali jika skala keluar yang lebih besar dipicu atau periode cooldown berakhir. Selama periode pendinginan penskalaan keluar berlaku, kapasitas yang ditambahkan dengan cara memulai aktivitas penskalaan keluar dihitung sebagai bagian dari kapasitas yang diinginkan untuk aktivitas penskalaan keluar berikutnya.

- Dengan periode cooldown scale-in, tujuannya adalah untuk menskalakan secara konservatif untuk melindungi ketersediaan aplikasi Anda, sehingga aktivitas scale-in diblokir hingga periode cooldown scale-in berakhir. Namun, jika alarm lain memicu aktivitas pengecilan skala selama periode jeda pakai untuk pembesaran skala, Application Auto Scaling akan langsung memperkecil skala target. Dalam hal ini, periode cooldown scale-in berhenti dan tidak selesai.

Setiap periode jeda pakai diukur dalam hitungan detik dan hanya berlaku untuk aktivitas penskalaan terkait kebijakan. Selama periode jeda pakai, ketika tindakan terjadwal dimulai pada waktu yang dijadwalkan, tindakan ini dapat segera memicu aktivitas penskalaan tanpa menunggu periode jeda pakai berakhir.

Anda dapat memulai dengan nilai default, yang dapat diatur kemudian. Misalnya, Anda mungkin perlu meningkatkan periode jeda pakai untuk mencegah kebijakan penskalaan pelacakan target Anda menjadi terlalu agresif terhadap perubahan yang terjadi dalam periode waktu singkat.

Nilai default

Application Auto Scaling memberikan nilai default 600 untuk ElastiCache dan nilai default 300 untuk target skalabel berikut:

- WorkSpaces Armada aplikasi
- Klaster Aurora DB
- Layanan ECS
- Cluster Neptunus
- SageMaker Varian titik akhir AI
- SageMaker Komponen inferensi AI
- SageMaker Konkurensi yang disediakan tanpa server AI
- Armada Spot
- Kolam renang dari WorkSpaces
- Sumber daya khusus

Untuk semua target skalabel lainnya, nilai defaultnya adalah 0 atau null:

- Klasifikasi dokumen dan titik akhir pengenal entitas Amazon Comprehend
- Tabel DynamoDB dan indeks sekunder global
- Tabel Amazon Keyspaces
- Konkurensi terprovisi Lambda
- Penyimpanan broker Amazon MSK

Nilai nol diperlakukan sama dengan nilai nol saat Application Auto Scaling mengevaluasi periode cooldown.

Anda dapat memperbarui salah satu nilai default, termasuk nilai nol, untuk mengatur periode cooldown Anda sendiri.

Pertimbangan-pertimbangan

Pertimbangan berikut berlaku saat bekerja dengan kebijakan penskalaan pelacakan target:

- Jangan membuat, mengedit, atau menghapus CloudWatch alarm yang digunakan dengan kebijakan penskalaan pelacakan target. Application Auto Scaling membuat dan mengelola CloudWatch alarm yang terkait dengan kebijakan penskalaan pelacakan target Anda dan menghapusnya saat tidak diperlukan lagi.
- Jika metrik kehilangan titik data, ini menyebabkan status CloudWatch alarm berubah menjadi`INSUFFICIENT_DATA`. Ketika ini terjadi, Application Auto Scaling tidak dapat menskalakan target Anda yang dapat diskalakan hingga titik data baru ditemukan. Untuk informasi selengkapnya, lihat Mengonfigurasi cara CloudWatch alarm menangani data yang hilang di CloudWatch Panduan Pengguna Amazon.
- Jika metrik jarang dilaporkan oleh desain, matematika metrik dapat membantu. Misalnya, untuk menggunakan nilai terbaru, maka gunakan `FILL(m1,REPEAT)` fungsi di `m1` mana metrik.
- Anda mungkin melihat kesenjangan antara nilai target dan titik data metrik aktual. Ini karena Application Auto Scaling selalu bertindak konservatif dengan membulatkan ke atas atau ke bawah saat menentukan berapa banyak kapasitas yang dapat ditambahkan atau dihapus. Hal ini mencegah penambahan kapasitas yang tidak memadai atau menghilangkan kapasitas yang terlalu banyak. Namun, untuk target yang dapat diskalakan dengan kapasitas kecil, titik data metrik aktual mungkin tampak jauh dari nilai target.

Misalnya, Anda menetapkan nilai target 50 persen untuk pemanfaatan CPU dan grup Auto Scaling Anda kemudian melebihi target. Kami dapat menentukan bahwa penambahan 1,5 instance akan mengurangi penggunaan CPU hingga mendekati 50 persen. Karena tidak mungkin untuk menambahkan 1,5 instance, kami membulatkan ke atas dan menambahkan dua instance. Hal ini dapat mengurangi penggunaan CPU menjadi nilai di bawah 50 persen, tetapi memastikan bahwa aplikasi Anda memiliki sumber daya yang cukup untuk mendukungnya. Demikian pula, jika kami menentukan bahwa menghapus 0,5 instance meningkatkan pemanfaatan CPU Anda hingga di atas 50 persen, kami akan memilih untuk tidak meningkatkan skala sampai metrik cukup rendah sehingga kami pikir penskalaan tidak akan menyebabkan osilasi.

Untuk target yang dapat diskalakan dengan kapasitas yang lebih besar, menambah atau menghilangkan kapasitas menyebabkan berkurangnya kesenjangan antara nilai target dan titik data metrik aktual.

- Kebijakan penskalaan pelacakan target mengasumsikan bahwa penskalaan ke luar harus dilakukan saat metrik yang ditentukan berada di atas nilai target. Anda tidak dapat menggunakan kebijakan penskalaan pelacakan target untuk menskalakan ke luar jika metrik yang ditentukan berada di bawah nilai target.

Beberapa kebijakan penskalaan

Anda dapat memiliki beberapa kebijakan penskalaan pelacakan target untuk target yang dapat diskalakan, asalkan setiapnya menggunakan metrik yang berbeda. Tujuan Application Auto Scaling adalah untuk selalu memprioritaskan ketersediaan, sehingga perilakunya berbeda tergantung pada apakah kebijakan pelacakan target siap untuk diperkecil atau diperbesar. Hal ini akan memperkecil skala target yang dapat diskala jika salah satu kebijakan pelacakan target siap untuk diperkecil skalanya, tetapi hanya akan memperbesar skala jika semua kebijakan pelacakan target (dengan penskalaan dalam porsi aktif) siap untuk diperbesar skalanya.

Jika beberapa kebijakan penskalaan menginstruksikan target yang dapat diskalakan untuk skala keluar atau masuk pada saat yang sama, Application Auto Scaling menskalakan berdasarkan kebijakan yang menyediakan kapasitas terbesar untuk skala masuk dan skala keluar. Ini memberikan fleksibilitas yang lebih besar untuk mencakup beberapa skenario dan memastikan bahwa selalu ada kapasitas yang cukup untuk memproses beban kerja Anda.

Anda dapat menonaktifkan bagian penskalaan dari kebijakan penskalaan pelacakan target untuk menggunakan metode penskalaan yang berbeda dari yang Anda gunakan untuk skala keluar.

Misalnya, Anda dapat menggunakan kebijakan penskalaan langkah untuk memperbesar skala saat menggunakan kebijakan penskalaan pelacakan target untuk memperkecil skala.

Namun, kami menyarankan untuk berhati-hati ketika menggunakan kebijakan penskalaan pelacakan target dengan kebijakan penskalaan langkah karena konflik antarkebijakan ini dapat menyebabkan hal yang tidak diinginkan. Misalnya, jika kebijakan penskalaan langkah memulai aktivitas pembesaran skala sebelum kebijakan pelacakan target siap untuk memulai pembesaran skala, aktivitas pembesaran skala tidak akan diblokir. Setelah aktivitas pembesaran skala selesai, kebijakan pelacakan target dapat menginstruksikan target yang dapat diskalakan untuk kembali diperkecil skalanya.

Untuk muatan kerja yang bersifat siklus, Anda juga memiliki opsi untuk mengotomatisasi perubahan kapasitas pada jadwal menggunakan penskalaan terjadwal. Untuk setiap tindakan terjadwal, nilai kapasitas minimum dan maksimum yang baru dapat ditentukan. Nilai ini membentuk batasan kebijakan penskalaan. Kombinasi penskalaan terjadwal dan penskalaan pelacakan target dapat membantu mengurangi dampak peningkatan tajam pada tingkat pemanfaatan, ketika kapasitas diperlukan dengan segera.

Perintah yang umum digunakan untuk penskalaan pembuatan kebijakan, manajemen, dan penghapusan

Perintah yang umum digunakan untuk bekerja dengan kebijakan penskalaan meliputi:

- register-scalable-targetuntuk mendaftarkan AWS atau menyesuaikan sumber daya sebagai target yang dapat diskalakan (sumber daya yang dapat diskalakan oleh Application Auto Scaling), dan untuk menangguhkan dan melanjutkan penskalaan.
- put-scaling-policy untuk menambah atau mengubah kebijakan penskalaan untuk target terukur yang ada.
- describe-scaling-activitiesuntuk mengembalikan informasi tentang aktivitas penskalaan di suatu AWS Wilayah.
- describe-scaling-policies untuk mengembalikan informasi tentang kebijakan penskalaan di Wilayah AWS .
- delete-scaling-policy untuk menghapus kebijakan penskalaan.

Sumber daya terkait

Untuk informasi tentang membuat kebijakan penskalaan pelacakan target untuk grup Auto Scaling, lihat Kebijakan penskalaan pelacakan target untuk Penskalaan Otomatis Amazon EC2 di Panduan Pengguna Amazon EC2 Auto Scaling.

Batasan

Berikut ini adalah keterbatasan saat menggunakan kebijakan penskalaan pelacakan target:

- Target yang dapat diskalakan tidak bisa menjadi klaster Amazon EMR. Kebijakan penskalaan pelacakan target untuk Amazon EMR tidak didukung.
- Ketika sebuah klaster Amazon MSK adalah target yang dapat diskalakan, skala di dinonaktifkan dan tidak dapat diaktifkan.
- Anda tidak dapat menggunakan operasi `RegisterScalableTarget` atau `PutScalingPolicy` API untuk memperbarui rencana AWS Auto Scaling penskalaan.
- Akses konsol untuk melihat, menambah, memperbarui, atau menghapus kebijakan penskalaan pelacakan target pada sumber daya yang dapat diskalakan bergantung pada sumber daya yang Anda gunakan. Untuk informasi selengkapnya, lihat Layanan AWS yang dapat Anda gunakan dengan Application Auto Scaling.

Membuat kebijakan penskalaan pelacakan target untuk Application Auto Scaling menggunakan AWS CLI

Contoh ini menggunakan AWS CLI perintah untuk membuat kebijakan racking target untuk Armada Spot Amazon EC2. Untuk target skalabel yang berbeda, tentukan namespace di`--service-namespace`, dimensi yang dapat diskalakan di`--scalable-dimension`, dan ID sumber dayanya di. `--resource-id`

Saat menggunakan AWS CLI, ingatlah bahwa perintah Anda berjalan di AWS Region konfigurasi untuk profil Anda. Jika Anda ingin menjalankan perintah di Wilayah yang berbeda, ubah Wilayah default untuk profil Anda, atau gunakan parameter `--region` bersama perintah tersebut.

Tugas

- Langkah 1: Daftarkan target yang dapat diskalakan
- Langkah 2: Buat kebijakan penskalaan pelacakan target

Langkah 1: Daftarkan target yang dapat diskalakan

Jika Anda belum melakukannya, daftarkan target yang dapat diskalakan. Gunakan perintah register-scalable-target untuk mendaftar sumber daya tertentu dalam layanan target sebagai target yang dapat diskalakan. Contoh berikut meregistrasikan permintaan Armada Spot dengan Application Auto Scaling. Application Auto Scaling dapat menskalakan jumlah kasus di Armada Spot pada minimum 2 instans dan maksimum 10 instans. Ganti masing-masing *user input placeholder* dengan informasi Anda sendiri.

Linux, macOS, atau Unix

```
aws application-autoscaling register-scalable-target --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --min-capacity 2 --max-capacity 10
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE ^
  --min-capacity 2 --max-capacity 10
```

Output

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan. Berikut ini adalah output contoh.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Langkah 2: Buat kebijakan penskalaan pelacakan target

Untuk membuat kebijakan penskalaan pelacakan target, Anda dapat menggunakan contoh berikut untuk membantu Anda memulai.

Untuk membuat kebijakan penskalaan pelacakan target

1. Gunakan `cat` perintah berikut untuk menyimpan nilai target untuk kebijakan penskalaan Anda dan spesifikasi metrik yang telah ditentukan sebelumnya dalam file JSON yang diberi nama `config.json` di direktori home Anda. Berikut ini adalah contoh konfigurasi pelacakan target yang menjaga pemanfaatan CPU rata-rata sebesar 50 persen.

```
$ cat ~/config.json
{
  "TargetValue": 50.0,
  "PredefinedMetricSpecification":
    {
      "PredefinedMetricType": "EC2SpotFleetRequestAverageCPUUtilization"
    }
}
```

   Untuk informasi lebih lanjut, lihat PredefinedMetricSpecification dalam Referensi API Application Auto Scaling.

   Atau, Anda dapat menggunakan metrik khusus untuk penskalaan dengan membuat spesifikasi metrik yang disesuaikan dan menambahkan nilai untuk setiap parameter dari CloudWatch. Berikut ini adalah contoh konfigurasi pelacakan target yang menjaga pemanfaatan rata-rata metrik yang ditentukan pada 100.

```
$ cat ~/config.json
{
   "TargetValue": 100.0,
   "CustomizedMetricSpecification":{
      "MetricName": "MyUtilizationMetric",
      "Namespace": "MyNamespace",
      "Dimensions": [
         {
            "Name": "MyOptionalMetricDimensionName",
            "Value": "MyOptionalMetricDimensionValue"
         }
      ],
      "Statistic": "Average",
      "Unit": "Percent"
   }
}
```

Untuk informasi lebih lanjut, lihat CustomizedMetricSpecification dalam Referensi API Application Auto Scaling.

2. Gunakan perintah berikut ini put-scaling-policy, beserta `config.json` file yang Anda buat, untuk membuat kebijakan penskalaan yang dinamai `cpu50-target-tracking-scaling-policy`.

   Linux, macOS, atau Unix

```
aws application-autoscaling put-scaling-policy --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --policy-name cpu50-target-tracking-scaling-policy --policy-type
 TargetTrackingScaling \
  --target-tracking-scaling-policy-configuration file://config.json
```

   Windows

```
aws application-autoscaling put-scaling-policy --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE ^
  --policy-name cpu50-target-tracking-scaling-policy --policy-type
 TargetTrackingScaling ^
  --target-tracking-scaling-policy-configuration file://config.json
```

   Output

   Jika berhasil, perintah ini mengembalikan ARNs dan nama dari dua CloudWatch alarm yang dibuat atas nama Anda. Berikut ini adalah output contoh.

```
{
    "PolicyARN": "arn:aws:autoscaling:region:account-
id:scalingPolicy:policy-id:resource/ec2/spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE:policyName/cpu50-target-tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-
b46e-434a-a60f-3b36d653feca",
            "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca"
```

```
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-
d19b-4a63-a812-6c67aaf2910d",
            "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
        }
    ]
}
```

Langkah 3: Jelaskan kebijakan penskalaan pelacakan target

Anda dapat mendeskripsikan semua kebijakan penskalaan untuk spacename layanan yang ditentukan dengan menggunakan perintah describe-scaling-policies berikut.

```
aws application-autoscaling describe-scaling-policies --service-namespace ec2
```

Anda dapat memfilter hasil hanya untuk kebijakan penskalaan pelacakan target menggunakan parameter `--query`. Untuk informasi lebih lanjut tentang sintaks untuk `query`, lihat Mengontrol output perintah dari AWS CLI di AWS Command Line Interface Panduan Pengguna.

Linux, macOS, atau Unix

```
aws application-autoscaling describe-scaling-policies --service-namespace ec2 \
  --query 'ScalingPolicies[?PolicyType==`TargetTrackingScaling`]'
```

Windows

```
aws application-autoscaling describe-scaling-policies --service-namespace ec2 ^
  --query "ScalingPolicies[?PolicyType==`TargetTrackingScaling`]"
```

Output

Berikut ini adalah output contoh.

```
[
    {
        "PolicyARN": "PolicyARN",
        "TargetTrackingScalingPolicyConfiguration": {
```

```
            "PredefinedMetricSpecification": {
                "PredefinedMetricType": "EC2SpotFleetRequestAverageCPUUtilization"
            },
            "TargetValue": 50.0
        },
        "PolicyName": "cpu50-target-tracking-scaling-policy",
        "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
        "ServiceNamespace": "ec2",
        "PolicyType": "TargetTrackingScaling",
        "ResourceId": "spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
        "Alarms": [
            {
                "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-
b46e-434a-a60f-3b36d653feca",
                "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca"
            },
            {
                "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-
d19b-4a63-a812-6c67aaf2910d",
                "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
            }
        ],
        "CreationTime": 1515021724.807
    }
]
```

Menghapus kebijakan penskalaan pelacakan target untuk Application Auto Scaling menggunakan AWS CLI

Setelah selesai dengan kebijakan penskalaan pelacakan target, Anda dapat menghapusnya menggunakan perintah delete-scaling-policy.

Perintah berikut akan menghapus kebijakan penskalaan target tertentu untuk permintaan Armada Spot tertentu. Ini juga menghapus CloudWatch alarm yang Application Auto Scaling dibuat atas nama Anda.

Linux, macOS, atau Unix

```
aws application-autoscaling delete-scaling-policy --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --policy-name cpu50-target-tracking-scaling-policy
```

Windows

```
aws application-autoscaling delete-scaling-policy --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE ^
  --policy-name cpu50-target-tracking-scaling-policy
```

Membuat kebijakan penskalaan pelacakan target untuk Application Auto Scaling menggunakan matematika metrik

Menggunakan matematika metrik, Anda dapat menanyakan beberapa CloudWatch metrik dan menggunakan ekspresi matematika untuk membuat deret waktu baru berdasarkan metrik ini. Anda dapat memvisualisasikan deret waktu yang dihasilkan di CloudWatch konsol dan menambahkannya ke dasbor. Untuk informasi selengkapnya tentang matematika metrik, lihat Menggunakan matematika metrik di Panduan CloudWatch Pengguna Amazon.

Pertimbangan berikut berlaku untuk ekspresi matematika metrik:

- Anda dapat menanyakan CloudWatch metrik apa pun yang tersedia. Setiap metrik adalah kombinasi unik dari nama metrik, namespace, dan nol atau lebih dimensi.
- Anda dapat menggunakan operator aritmatika (+ - */^), fungsi statistik (seperti AVG atau SUM), atau fungsi lain yang mendukung. CloudWatch
- Anda dapat menggunakan metrik dan hasil ekspresi matematika lainnya dalam rumus ekspresi matematika.
- Setiap ekspresi yang digunakan dalam spesifikasi metrik pada akhirnya harus mengembalikan satu deret waktu.
- Anda dapat memverifikasi bahwa ekspresi matematika metrik valid dengan menggunakan CloudWatch konsol atau CloudWatch GetMetricDataAPI.

Topik

Contoh: backlog antrian Amazon SQS per tugas

Untuk menghitung backlog antrean Amazon SQS per tugas, ambil perkiraan jumlah pesan yang tersedia untuk diambil dari antrian dan bagi nomor tersebut dengan jumlah tugas Amazon ECS yang berjalan di layanan. Untuk informasi selengkapnya, lihat Auto Scaling Amazon Elastic Container Service (ECS) menggunakan metrik kustom di AWS Blog Komputasi.

Logika untuk ekspresi adalah ini:

```
sum of (number of messages in the queue)/(number of tasks that are
currently in the RUNNING state)
```

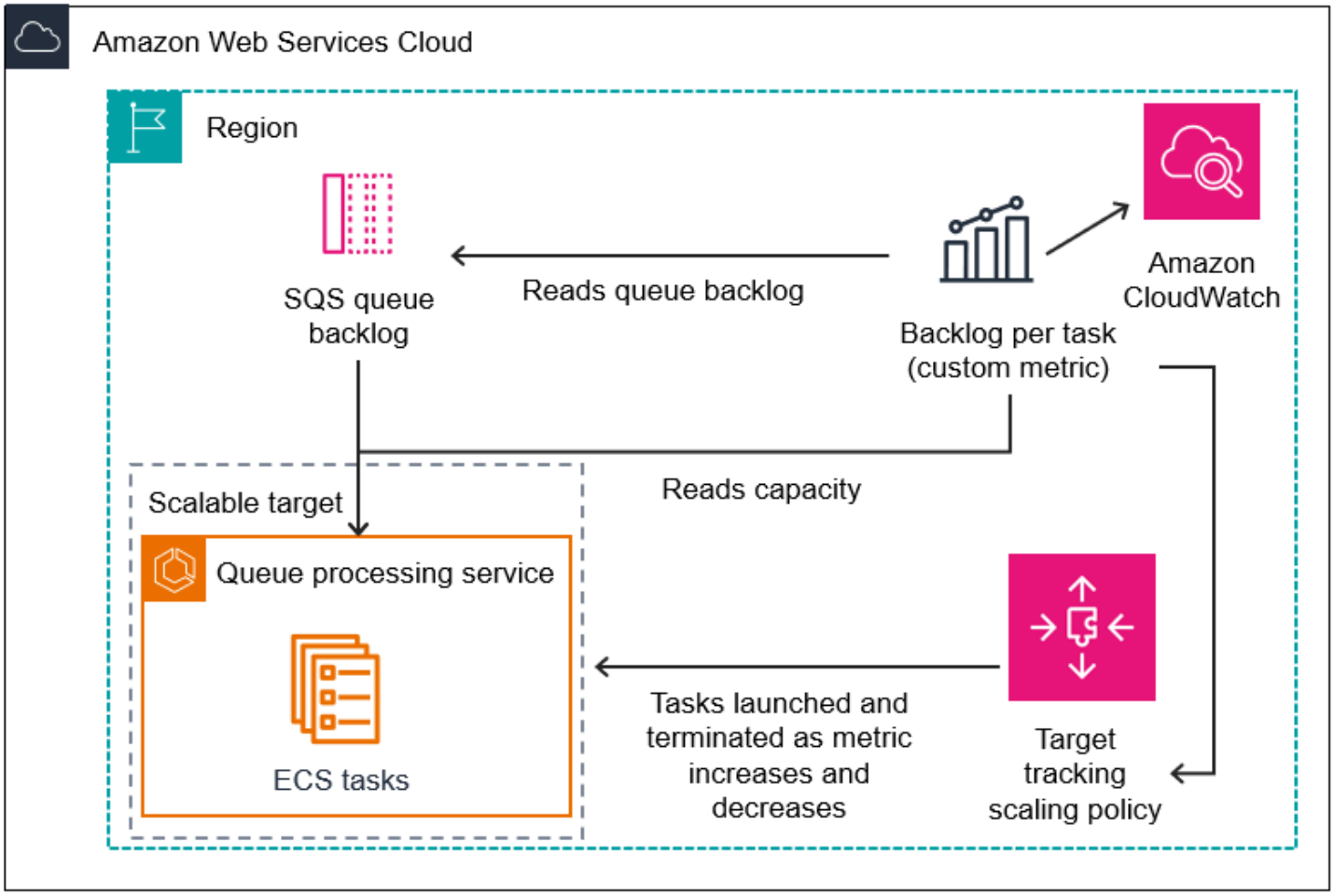
Maka informasi CloudWatch metrik Anda adalah sebagai berikut.

ID	CloudWatch metrik	Statistik	Periode
m1	ApproximateNumberOfMessagesVisible	Jumlah	1 menit
m2	RunningTaskCount	Rata-rata	1 menit

ID dan ekspresi matematika metrik Anda adalah sebagai berikut.

ID	Ekspresi
e1	`(m1)/(m2)`

Diagram berikut menggambarkan arsitektur untuk metrik ini:



Untuk menggunakan matematika metrik ini untuk membuat kebijakan penskalaan pelacakan target ()AWS CLI

1. Simpan ekspresi matematika metrik sebagai bagian dari spesifikasi metrik yang disesuaikan dalam file JSON bernama`config.json`.

   Gunakan contoh berikut untuk membantu Anda memulai. Ganti masing-masing *`user input placeholder`* dengan informasi Anda sendiri.

```
{
    "CustomizedMetricSpecification": {
        "Metrics": [
            {
                "Label": "Get the queue size (the number of messages waiting to be
 processed)",
                "Id": "m1",
                "MetricStat": {
```

```
                    "Metric": {
                        "MetricName": "ApproximateNumberOfMessagesVisible",
                        "Namespace": "AWS/SQS",
                        "Dimensions": [
                            {
                                "Name": "QueueName",
                                "Value": "my-queue"
                            }
                        ]
                    },
                    "Stat": "Sum"
                },
                "ReturnData": false
            },
            {
                "Label": "Get the ECS running task count (the number of currently
running tasks)",
                "Id": "m2",
                "MetricStat": {
                    "Metric": {
                        "MetricName": "RunningTaskCount",
                        "Namespace": "ECS/ContainerInsights",
                        "Dimensions": [
                            {
                                "Name": "ClusterName",
                                "Value": "my-cluster"
                            },
                            {
                                "Name": "ServiceName",
                                "Value": "my-service"
                            }
                        ]
                    },
                    "Stat": "Average"
                },
                "ReturnData": false
            },
            {
                "Label": "Calculate the backlog per instance",
                "Id": "e1",
                "Expression": "m1 / m2",
                "ReturnData": true
            }
        ]
```

```
    },
    "TargetValue": 100
}
```

Untuk informasi lebih lanjut, lihat TargetTrackingScalingPolicyConfiguration dalam Referensi API Application Auto Scaling.

> **Note**
>
> Berikut adalah beberapa sumber daya tambahan yang dapat membantu Anda menemukan nama metrik, ruang nama, dimensi, dan statistik untuk CloudWatch metrik:
>
> - Untuk informasi tentang metrik yang tersedia untuk AWS layanan, lihat AWS layanan yang memublikasikan CloudWatch metrik di CloudWatch Panduan Pengguna Amazon.
> - Untuk mendapatkan nama metrik, namespace, dan dimensi yang tepat (jika ada) untuk CloudWatch metrik dengan metrik AWS CLI, lihat daftar-metrik.

2. Untuk membuat kebijakan ini, jalankan put-scaling-policyperintah menggunakan file JSON sebagai input, seperti yang ditunjukkan dalam contoh berikut.

```
aws application-autoscaling put-scaling-policy --policy-name sqs-backlog-target-tracking-scaling-policy \
  --service-namespace ecs --scalable-dimension ecs:service:DesiredCount --resource-id service/my-cluster/my-service \
  --policy-type TargetTrackingScaling --target-tracking-scaling-policy-configuration file://config.json
```

Jika berhasil, perintah ini mengembalikan Amazon Resource Name (ARN) kebijakan dan dua CloudWatch alarm ARNs yang dibuat atas nama Anda.

```
{
    "PolicyARN": "arn:aws:autoscaling:us-west-2:012345678910:scalingPolicy:
 8784a896-b2ba-47a1-b08c-27301cc499a1:resource/ecs/service/my-cluster/my-service:policyName/sqs-backlog-target-tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:us-west-2:012345678910:alarm:TargetTracking-service/my-cluster/my-service-AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0",
```

```
            "AlarmName": "TargetTracking-service/my-cluster/my-service-
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/my-cluster/my-service-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4",
            "AlarmName": "TargetTracking-service/my-cluster/my-service-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4"
        }
    ]
}
```

> **Note**
>
> Jika perintah ini menimbulkan kesalahan, pastikan Anda telah memperbarui AWS CLI secara lokal ke versi terbaru.

Batasan

- Ukuran permintaan maksimum adalah 50 KB. Ini adalah ukuran payload total untuk permintaan PutScalingPolicyAPI saat Anda menggunakan matematika metrik dalam definisi kebijakan. Jika Anda melebihi batas ini, Application Auto Scaling menolak permintaan tersebut.
- Layanan berikut tidak didukung saat menggunakan matematika metrik dengan kebijakan penskalaan pelacakan target:
  - Amazon Keyspaces (untuk Apache Cassandra)
  - DynamoDB
  - Amazon EMR
  - Amazon MSK
  - Amazon Neptune

Kebijakan penskalaan langkah untuk Application Auto Scaling

Kebijakan penskalaan langkah menskalakan kapasitas aplikasi Anda dalam peningkatan yang telah ditentukan berdasarkan alarm. CloudWatch Anda dapat menentukan kebijakan penskalaan terpisah untuk menangani penskalaan (peningkatan kapasitas) dan penskalaan (penurunan kapasitas) saat ambang alarm dilanggar.

Dengan kebijakan penskalaan langkah, Anda membuat dan mengelola CloudWatch alarm yang menjalankan proses penskalaan. Ketika alarm dilanggar, Application Auto Scaling memulai kebijakan penskalaan yang terkait dengan alarm tersebut.

Kebijakan penskalaan langkah menskalakan kapasitas menggunakan serangkaian penyesuaian, yang dikenal sebagai penyesuaian langkah. Ukuran penyesuaian bervariasi berdasarkan besarnya pelanggaran alarm.

- Jika pelanggaran melebihi ambang batas pertama, Application Auto Scaling akan menerapkan penyesuaian langkah pertama.
- Jika pelanggaran melebihi ambang kedua, Application Auto Scaling akan menerapkan penyesuaian langkah kedua, dan seterusnya.

Hal ini memungkinkan kebijakan penskalaan untuk merespons dengan tepat perubahan kecil dan besar dalam metrik alarm.

Kebijakan ini akan terus menanggapi pelanggaran alarm tambahan, bahkan saat aktivitas penskalaan sedang berlangsung. Ini berarti Application Auto Scaling akan mengevaluasi semua pelanggaran alarm saat terjadi. Periode cooldown digunakan untuk melindungi terhadap penskalaan berlebih karena beberapa pelanggaran alarm yang terjadi secara berurutan.

Seperti pelacakan target, penskalaan langkah dapat membantu secara otomatis menskalakan kapasitas aplikasi Anda saat terjadi perubahan lalu lintas. Namun, kebijakan pelacakan target cenderung lebih mudah diterapkan dan dikelola untuk kebutuhan penskalaan yang stabil.

Target yang dapat diskalakan yang didukung

Anda dapat menggunakan kebijakan penskalaan langkah dengan target yang dapat diskalakan berikut:

- WorkSpaces Armada aplikasi
- Klaster Aurora DB
- Layanan ECS
- Klaster EMR
- SageMaker Varian titik akhir AI
- SageMaker Komponen inferensi AI
- SageMaker Konkurensi yang disediakan tanpa server AI
- Armada Spot
- Sumber daya khusus

Daftar Isi

Cara kerja penskalaan langkah untuk Application Auto Scaling

Topik ini menjelaskan cara kerja penskalaan langkah dan memperkenalkan elemen kunci dari kebijakan penskalaan langkah.

Daftar Isi

- Akses konsol

Cara kerjanya

Untuk menggunakan penskalaan langkah, Anda membuat CloudWatch alarm yang memantau metrik untuk target yang dapat diskalakan. Tentukan metrik, nilai ambang batas, dan jumlah periode evaluasi yang menentukan pelanggaran alarm. Anda juga membuat kebijakan penskalaan langkah yang menentukan cara menskalakan kapasitas saat ambang alarm dilanggar dan mengaitkannya dengan target yang dapat diskalakan.

Tambahkan penyesuaian langkah dalam kebijakan. Anda dapat menentukan penyesuaian langkah yang berbeda berdasarkan ukuran pelanggaran alarm. Contoh:

- Skalakan 10 unit kapasitas jika metrik alarm mencapai 60 persen
- Skalakan 30 unit kapasitas jika metrik alarm mencapai 75 persen
- Skalakan dengan 40 unit kapasitas jika metrik alarm mencapai 85 persen

Ketika ambang batas alarm dilanggar untuk jumlah periode evaluasi yang ditentukan, Application Auto Scaling akan menerapkan penyesuaian langkah yang ditentukan dalam kebijakan. Penyesuaian dapat berlanjut untuk pelanggaran alarm tambahan hingga status alarm kembali ke. 0K

Aktivitas penskalaan dilakukan dengan periode cooldown di antara mereka untuk mencegah fluktuasi kapasitas yang cepat. Anda dapat secara opsional mengonfigurasi periode cooldown untuk kebijakan penskalaan Anda.

Penyesuaian langkah

Saat membuat kebijakan penskalaan langkah, Anda menentukan satu atau beberapa penyesuaian langkah yang secara otomatis menskalakan kapasitas target secara dinamis berdasarkan ukuran pelanggaran alarm. Setiap penyesuaian langkah menentukan hal berikut:

- Batas bawah untuk nilai metrik
- Batas atas untuk nilai metrik
- Jumlah yang ditentukan untuk diskalakan, berdasarkan jenis penyesuaian penskalaan

CloudWatch mengumpulkan titik data metrik berdasarkan statistik untuk metrik yang terkait dengan CloudWatch alarm Anda. Ketika alarm dilanggar, kebijakan penskalaan yang sesuai akan dipanggil.

Application Auto Scaling menerapkan tipe agregasi yang Anda tentukan ke titik data metrik terbaru dari CloudWatch (sebagai lawan dari data metrik mentah). Penskalaan ini membandingkan nilai metrik agregat ini terhadap batas atas dan bawah yang ditentukan oleh penyesuaian langkah untuk menentukan penyesuaian langkah mana yang harus dilakukan.

Anda menentukan batas atas dan bawah terkait dengan ambang batas pelanggaran. Sebagai contoh, katakanlah Anda membuat CloudWatch alarm dan kebijakan scale-out ketika metrik di atas 50 persen. Anda kemudian membuat alarm kedua dan kebijakan skala dalam ketika metrik di bawah 50 persen. Anda membuat serangkaian penyesuaian langkah dengan jenis penyesuaian `PercentChangeInCapacity` untuk setiap kebijakan:

Contoh: Penyesuaian langkah untuk kebijakan menskalakan keluar

Batas bawah	Batas atas	Penyesuaian
0	10	0
10	20	10
20	nol	30

Contoh: Penyesuaian langkah untuk kebijakan penskalaaan kedalam

Batas bawah	Batas atas	Penyesuaian
-10	0	0
-20	-10	-10
nol	-20	-30

Ini membuat konfigurasi penskalaan berikut.

```
Metric value

-infinity          30%    40%          60%     70%             infinity
-------------------------------------------------------------------------
          -30%      | -10% | Unchanged   | +10%  |       +30%
-------------------------------------------------------------------------
```

Sekarang, katakanlah Anda menggunakan konfigurasi penskalaan ini pada target yang dapat diskalakan yang memiliki kapasitas 10. Poin-poin berikut merangkum perilaku konfigurasi penskalaan dalam kaitannya dengan kapasitas target yang dapat diskalakan:

- Kapasitas asli dipertahankan sementara nilai metrik agregat lebih besar dari 40 dan kurang dari 60.
- Jika nilai metrik mencapai 60, Application Auto Scaling meningkatkan kapasitas target yang dapat diskalakan sebesar 1, menjadi 11. Itu berdasarkan penyesuaian langkah kedua dari kebijakan penskalaan naik (tambahkan 10 persen dari 10). Setelah kapasitas baru ditambahkan, Application Auto Scaling meningkatkan kapasitas saat ini menjadi 11. Jika nilai metrik meningkat menjadi 70 bahkan setelah peningkatan kapasitas, Application Auto Scaling meningkatkan kapasitas target sebesar 3, menjadi 14. Itu berdasarkan penyesuaian langkah ketiga dari kebijakan penskalaan (tambahkan 30 persen dari 11, 3,3 dibulatkan ke bawah menjadi 3).
- Jika nilai metrik mencapai 40, Application Auto Scaling mengurangi kapasitas target yang dapat diskalakan sebesar 1, menjadi 13, berdasarkan penyesuaian langkah kedua dari kebijakan scale-in (hapus 10 persen dari 14, 1,4, dibulatkan ke bawah menjadi 1). Jika nilai metrik turun menjadi 30 bahkan setelah penurunan kapasitas ini, Application Auto Scaling mengurangi kapasitas target sebesar 3, menjadi 10, berdasarkan penyesuaian langkah ketiga pada kebijakan skala (hapus 30 persen 13, 3,9 dibulatkan ke bawah menjadi 3).

Saat Anda menentukan penyesuaian langkah untuk kebijakan penskalaan Anda, perhatikan hal berikut:

- Rentang penyesuaian langkah Anda tidak dapat tumpang tindih atau memiliki kesenjangan.
- Hanya satu penyesuaian langkah yang dapat memiliki batas bawah nol (negatif tak terhingga). Jika satu penyesuaian langkah memiliki batas bawah negatif, maka harus ada penyesuaian langkah dengan batas bawah nol.
- Hanya satu penyesuaian langkah yang dapat memiliki batas atas nol (positif tak terhingga). Jika satu penyesuaian langkah memiliki batas atas positif, maka harus ada penyesuaian langkah dengan batas atas nol.
- Batas atas dan bawah tidak boleh nol dalam penyesuaian langkah yang sama.
- Apabila nilai metrik berada di atas ambang batas penembusan, batas bawah bersifat inklusif dan batas atas bersifat eksklusif. Apabila nilai metrik berada di bawah ambang batas penembusan, batas bawah bersifat eksklusitf dan batas atas bersifat inklusif.

Jenis penyesuaian penskalaan

Anda dapat menentukan kebijakan penskalaan yang melakukan tindakan penskalaan optimal, berdasarkan jenis penyesuaian penskalaan yang Anda pilih. Anda dapat menentukan jenis penyesuaian sebagai persentase dari kapasitas saat ini pada target yang dapat diskalakan atau dalam jumlah mutlak.

Application Auto Scaling mendukung jenis penyesuaian berikut untuk kebijakan penskalaan langkah:

- ChangeInCapacity—Meningkatkan atau mengurangi kapasitas saat ini pada target yang dapat diskalakan berdasarkan nilai yang ditentukan. Nilai positif meningkatkan kapasitas dan nilai negatif menurunkan kapasitas. Misalnya: Jika kapasitas saat ini adalah 3 dan penyesuaiannya adalah 5, Application Auto Scaling akan menambahkan 5 ke kapasitas sehingga totalnya adalah 8.
- ExactCapacity—Mengubah kapasitas saat ini pada target yang dapat diskalakan menjadi nilai yang ditentukan. Tentukan nilai non-negatif dengan jenis penyesuaian ini. Misalnya: Jika kapasitas saat ini adalah 3 dan penyesuaiannya adalah 5, maka Application Auto Scaling akan mengubah kapasitas menjadi 5.
- PercentChangeInCapacity—Meningkatkan atau mengurangi kapasitas saat ini pada target yang dapat diskalakan berdasarkan persentase tertentu. Nilai positif meningkatkan kapasitas dan nilai negatif menurunkan kapasitas. Misalnya: Jika kapasitas saat ini adalah 10 dan penyesuaiannya adalah 10 persen, Application Auto Scaling menambahkan 1 ke kapasitas sehingga totalnya adalah 11.

  Jika nilai yang dihasilkan bukan bilangan bulat, Application Auto Scaling membulatkannya sebagai berikut:

  - Nilai yang lebih besar dari 1 dibulatkan ke bawah. Misalnya, `12.7` dibulatkan ke `12`.
  - Nilai antara 0 dan 1 dibulatkan ke 1. Misalnya, `.67` dibulatkan ke `1`.
  - Nilai antara 0 dan -1 dibulatkan ke -1. Misalnya, `-.58` dibulatkan ke `-1`.
  - Nilai kurang dari -1 dibulatkan ke atas. Misalnya, `-6.67` dibulatkan ke `-6`.

  Dengan PercentChangeInCapacity, Anda juga dapat menentukan jumlah minimum untuk diskalakan menggunakan parameter `MinAdjustmentMagnitude`. Misalnya, anggaplah Anda membuat kebijakan yang menambahkan 25 persen dan Anda menentukan jumlah minimum sebesar 2. Jika target yang dapat diskalakan memiliki kapasitas 4 dan kebijakan penskalaan dilakukan, 25 persen dari 4 adalah 1. Namun, karena Anda menentukan peningkatan minimum sebesar 2, Application Auto Scaling akan menambahkan 2.

Periode pendinginan

Anda dapat secara opsional menentukan periode cooldown dalam kebijakan penskalaan langkah Anda.

Periode cooldown menentukan jumlah waktu kebijakan penskalaan menunggu aktivitas penskalaan sebelumnya berlaku.

Ada dua cara untuk merencanakan penggunaan periode cooldown untuk konfigurasi penskalaan langkah:

- Dengan periode cooldown untuk kebijakan scale-out, tujuannya adalah untuk terus (tetapi tidak berlebihan) skala. Setelah Application Auto Scaling berhasil menskalakan menggunakan kebijakan penskalaan, Application Auto Scaling mulai menghitung waktu cooldown. Kebijakan penskalaan tidak akan meningkatkan kapasitas yang diinginkan lagi kecuali jika skala keluar yang lebih besar dipicu atau periode cooldown berakhir. Selama periode pendinginan penskalaan keluar berlaku, kapasitas yang ditambahkan dengan cara memulai aktivitas penskalaan keluar dihitung sebagai bagian dari kapasitas yang diinginkan untuk aktivitas penskalaan keluar berikutnya.
- Dengan periode cooldown untuk kebijakan scale-in, tujuannya adalah untuk meningkatkan skala secara konservatif untuk melindungi ketersediaan aplikasi Anda, sehingga aktivitas scale-in diblokir hingga periode cooldown scale-in berakhir. Namun, jika alarm lain memicu aktivitas pengecilan skala selama periode jeda pakai untuk pembesaran skala, Application Auto Scaling akan langsung memperkecil skala target. Dalam hal ini, periode cooldown scale-in berhenti dan tidak selesai.

Misalnya, ketika puncak lalu lintas terjadi, alarm dipicu dan Application Auto Scaling secara otomatis menambah kapasitas untuk membantu menangani peningkatan beban. Jika Anda menetapkan periode cooldown untuk kebijakan scale-out Anda, ketika alarm memicu kebijakan untuk meningkatkan kapasitas sebesar 2, aktivitas penskalaan berhasil diselesaikan, dan periode cooldown scale-out dimulai. Jika alarm terpicu lagi selama periode cooldown tetapi pada penyesuaian langkah yang lebih agresif 3, peningkatan sebelumnya 2 dianggap sebagai bagian dari kapasitas saat ini. Oleh karena itu, hanya 1 yang ditambahkan ke kapasitas. Ini memungkinkan penskalaan lebih cepat daripada menunggu cooldown berakhir tetapi tanpa menambahkan kapasitas lebih dari yang Anda butuhkan.

Periode jeda pakai diukur dalam hitungan detik dan hanya berlaku untuk aktivitas penskalaan yang terkait kebijakan. Selama periode jeda pakai, ketika tindakan terjadwal dimulai pada waktu yang dijadwalkan, tindakan ini dapat segera memicu aktivitas penskalaan tanpa menunggu periode jeda pakai berakhir.

Nilai default adalah 300 jika tidak ada nilai yang ditentukan.

Perintah yang umum digunakan untuk penskalaan pembuatan kebijakan, manajemen, dan penghapusan

Perintah yang umum digunakan untuk bekerja dengan kebijakan penskalaan meliputi:

- register-scalable-targetuntuk mendaftarkan AWS atau menyesuaikan sumber daya sebagai target yang dapat diskalakan (sumber daya yang dapat diskalakan oleh Application Auto Scaling), dan untuk menangguhkan dan melanjutkan penskalaan.
- put-scaling-policy untuk menambah atau mengubah kebijakan penskalaan untuk target terukur yang ada.
- describe-scaling-activities untuk mengembalikan informasi tentang aktivitas penskalaan di Wilayah AWS .
- describe-scaling-policies untuk mengembalikan informasi tentang kebijakan penskalaan di Wilayah AWS .
- delete-scaling-policy untuk menghapus kebijakan penskalaan.

Pertimbangan-pertimbangan

Pertimbangan berikut berlaku saat bekerja dengan kebijakan penskalaan langkah:

- Pertimbangkan apakah Anda dapat memprediksi penyesuaian langkah pada aplikasi dengan cukup akurat untuk menggunakan penskalaan langkah. Jika metrik penskalaan Anda meningkat atau menurun secara proporsional dengan kapasitas target yang dapat diskalakan, sebaiknya gunakan kebijakan penskalaan pelacakan target sebagai gantinya. Anda masih memiliki opsi untuk menggunakan penskalaan langkah sebagai kebijakan tambahan untuk konfigurasi yang lebih canggih. Misalnya, Anda dapat mengonfigurasi respons yang lebih agresif saat pemanfaatan mencapai tingkat tertentu.
- Pastikan untuk memilih margin yang memadai antara ambang scale-out dan scale-in untuk mencegah flapping. Flapping adalah lingkaran penskalaan dan penskalaan yang tak terbatas. Artinya, jika tindakan penskalaan diambil, nilai metrik akan berubah dan memulai tindakan penskalaan lain dalam arah sebaliknya.

Sumber daya terkait

Untuk informasi tentang membuat kebijakan penskalaan langkah untuk grup Auto Scaling, lihat Langkah dan kebijakan penskalaan sederhana untuk Amazon EC2 Auto Scaling di Panduan Pengguna Amazon EC2 Auto Scaling.

Akses konsol

Akses konsol untuk melihat, menambah, memperbarui, atau menghapus kebijakan penskalaan langkah pada sumber daya yang dapat diskalakan bergantung pada sumber daya yang Anda gunakan. Untuk informasi selengkapnya, lihat Layanan AWS yang dapat Anda gunakan dengan Application Auto Scaling.

Buat kebijakan penskalaan langkah untuk Application Auto Scaling menggunakan AWS CLI

Contoh ini menggunakan AWS CLI perintah untuk membuat kebijakan penskalaan langkah untuk layanan Amazon ECS. Untuk target skalabel yang berbeda, tentukan namespace di`--service-namespace`, dimensi yang dapat diskalakan di`--scalable-dimension`, dan ID sumber dayanya di. `--resource-id`

Saat menggunakan AWS CLI, ingatlah bahwa perintah Anda berjalan di AWS Region konfigurasi untuk profil Anda. Jika Anda ingin menjalankan perintah di Wilayah yang berbeda, ubah Wilayah default untuk profil Anda, atau gunakan parameter `--region` bersama perintah tersebut.

Tugas

- Langkah 1: Daftarkan target yang dapat diskalakan
- Langkah 2: Buat kebijakan penskalaan langkah
- Langkah 3: Buat alarm yang memanggil kebijakan penskalaan

Langkah 1: Daftarkan target yang dapat diskalakan

Jika Anda belum melakukannya, daftarkan target yang dapat diskalakan. Gunakan perintah register-scalable-target untuk mendaftar sumber daya tertentu dalam layanan target sebagai target yang dapat diskalakan. Contoh berikut mendaftarkan layanan ECS Amazon dengan Application Auto Scaling. Application Auto Scaling dapat menskalakan jumlah tugas pada minimum 2 tugas dan

maksimum 10 tugas. Ganti masing-masing *user input placeholder* dengan informasi Anda sendiri.

Linux, macOS, atau Unix

```
aws application-autoscaling register-scalable-target --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --min-capacity 2 --max-capacity 10
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
  --resource-id service/my-cluster/my-service ^
  --min-capacity 2 --max-capacity 10
```

Output

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan. Berikut ini adalah output contoh.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Langkah 2: Buat kebijakan penskalaan langkah

Untuk membuat kebijakan penskalaan langkah untuk target yang dapat diskalakan, Anda dapat menggunakan contoh berikut untuk membantu Anda memulai.

Scale out

Untuk membuat kebijakan penskalaan langkah untuk skala keluar (meningkatkan kapasitas)

1. Gunakan `cat` perintah berikut untuk menyimpan konfigurasi kebijakan penskalaan langkah dalam file JSON bernama `config.json` di direktori home Anda. Berikut ini adalah contoh konfigurasi dengan jenis penyesuaian `PercentChangeInCapacity` yang meningkatkan kapasitas target yang dapat diskalakan berdasarkan penyesuaian langkah berikut (dengan asumsi ambang CloudWatch alarm 70):

- Tingkatkan kapasitas sebesar 10 persen ketika nilai metrik lebih besar dari atau sama dengan 70 tetapi kurang dari 85
- Tingkatkan kapasitas sebesar 20 persen ketika nilai metrik lebih besar dari atau sama dengan 85 tetapi kurang dari 95
- Tingkatkan kapasitas sebesar 30 persen ketika nilai metrik lebih besar dari atau sama dengan 95

```
$ cat ~/config.json
{
  "AdjustmentType": "PercentChangeInCapacity",
  "MetricAggregationType": "Average",
  "Cooldown": 60,
  "MinAdjustmentMagnitude": 1,
  "StepAdjustments": [
    {
      "MetricIntervalLowerBound": 0.0,
      "MetricIntervalUpperBound": 15.0,
      "ScalingAdjustment": 10
    },
    {
      "MetricIntervalLowerBound": 15.0,
      "MetricIntervalUpperBound": 25.0,
      "ScalingAdjustment": 20
    },
    {
      "MetricIntervalLowerBound": 25.0,
      "ScalingAdjustment": 30
    }
  ]
}
```

Untuk informasi lebih lanjut, lihat StepScalingPolicyConfiguration dalam Referensi API Application Auto Scaling.

2. Gunakan perintah berikut put-scaling-policy, beserta file `config.json` yang Anda buat, untuk membuat kebijakan penskalaan dengan nama `my-step-scaling-policy`.

   Linux, macOS, atau Unix

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
```

```
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --policy-name my-step-scaling-policy --policy-type StepScaling \
  --step-scaling-policy-configuration file://config.json
```

Windows

```
aws application-autoscaling put-scaling-policy --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
  --resource-id service/my-cluster/my-service ^
  --policy-name my-step-scaling-policy --policy-type StepScaling ^
  --step-scaling-policy-configuration file://config.json
```

Output

Output mencakup ARN yang berfungsi sebagai nama unik untuk kebijakan tersebut. Anda membutuhkannya untuk membuat CloudWatch alarm untuk kebijakan Anda. Berikut ini adalah output contoh.

```
{
    "PolicyARN":
 "arn:aws:autoscaling:region:123456789012:scalingPolicy:ac542982-cbeb-4294-891c-a5a941dfa787:resource/ecs/service/my-cluster/my-service:policyName/my-step-scaling-policy"
}
```

Scale in

Untuk membuat kebijakan penskalaan langkah untuk skala (kurangi kapasitas)

1. Gunakan `cat` perintah berikut untuk menyimpan konfigurasi kebijakan penskalaan langkah dalam file JSON bernama `config.json` di direktori home Anda. Berikut ini adalah contoh konfigurasi dengan jenis penyesuaian `ChangeInCapacity` yang mengurangi kapasitas target yang dapat diskalakan berdasarkan penyesuaian langkah berikut (dengan asumsi ambang CloudWatch alarm 50):

   - Kurangi kapasitas sebesar 1 ketika nilai metrik kurang dari atau sama dengan 50 tetapi lebih besar dari 40

- Kurangi kapasitas sebesar 2 ketika nilai metrik kurang dari atau sama dengan 40 tetapi lebih besar dari 30
- Kurangi kapasitas sebesar 3 ketika nilai metrik kurang dari atau sama dengan 30

```
$ cat ~/config.json
{
  "AdjustmentType": "ChangeInCapacity",
  "MetricAggregationType": "Average",
  "Cooldown": 60,
  "StepAdjustments": [
    {
      "MetricIntervalUpperBound": 0.0,
      "MetricIntervalLowerBound": -10.0,
      "ScalingAdjustment": -1
    },
    {
      "MetricIntervalUpperBound": -10.0,
      "MetricIntervalLowerBound": -20.0,
      "ScalingAdjustment": -2
    },
    {
      "MetricIntervalUpperBound": -20.0,
      "ScalingAdjustment": -3
    }
  ]
}
```

Untuk informasi lebih lanjut, lihat StepScalingPolicyConfiguration dalam Referensi API Application Auto Scaling.

2. Gunakan perintah berikut put-scaling-policy, beserta file `config.json` yang Anda buat, untuk membuat kebijakan penskalaan dengan nama `my-step-scaling-policy`.

   Linux, macOS, atau Unix

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --policy-name my-step-scaling-policy --policy-type StepScaling \
  --step-scaling-policy-configuration file://config.json
```

Windows

```
aws application-autoscaling put-scaling-policy --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
  --resource-id service/my-cluster/my-service ^
  --policy-name my-step-scaling-policy --policy-type StepScaling ^
  --step-scaling-policy-configuration file://config.json
```

Output

Output mencakup ARN yang berfungsi sebagai nama unik untuk kebijakan tersebut. Anda memerlukan ARN ini untuk membuat CloudWatch alarm untuk kebijakan Anda. Berikut ini adalah output contoh.

```
{
    "PolicyARN":
 "arn:aws:autoscaling:region:123456789012:scalingPolicy:ac542982-cbeb-4294-891c-a5a941dfa787:resource/ecs/service/my-cluster/my-service:policyName/my-step-scaling-policy"
}
```

Langkah 3: Buat alarm yang memanggil kebijakan penskalaan

Terakhir, gunakan CloudWatch put-metric-alarmperintah berikut untuk membuat alarm untuk digunakan dengan kebijakan penskalaan langkah Anda. Dalam contoh ini, Anda memiliki alarm yang didasarkan pada pemanfaatan CPU rata-rata. Alarm dikonfigurasi untuk berada dalam status ALARM jika mencapai ambang batas 70 persen selama setidaknya dua periode evaluasi sepanjang 60 detik secara berturut-turut. Untuk menentukan CloudWatch metrik yang berbeda atau menggunakan metrik kustom Anda sendiri, tentukan namanya di `--metric-name` dan namespace di. `--namespace`

Linux, macOS, atau Unix

```
aws cloudwatch put-metric-alarm --alarm-name Step-Scaling-AlarmHigh-ECS:service/my-cluster/my-service \
  --metric-name CPUUtilization --namespace AWS/ECS --statistic Average \
  --period 60 --evaluation-periods 2 --threshold 70 \
  --comparison-operator GreaterThanOrEqualToThreshold \
  --dimensions Name=ClusterName,Value=default Name=ServiceName,Value=sample-app-service
 \
```

```
  --alarm-actions PolicyARN
```

Windows

```
aws cloudwatch put-metric-alarm --alarm-name Step-Scaling-AlarmHigh-ECS:service/my-
cluster/my-service ^
  --metric-name CPUUtilization --namespace AWS/ECS --statistic Average ^
  --period 60 --evaluation-periods 2 --threshold 70 ^
  --comparison-operator GreaterThanOrEqualToThreshold ^
  --dimensions Name=ClusterName,Value=default Name=ServiceName,Value=sample-app-service
 ^
  --alarm-actions PolicyARN
```

Jelaskan kebijakan penskalaan langkah untuk Application Auto Scaling menggunakan AWS CLI

Anda dapat menjelaskan semua kebijakan penskalaan untuk namespace layanan menggunakan perintah. describe-scaling-policies Contoh berikut menjelaskan semua kebijakan penskalaan untuk semua layanan Amazon ECS. Untuk mencantumkannya untuk layanan Amazon ECS tertentu hanya tambahkan `--resource-id` opsi.

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs
```

Anda dapat memfilter hasilnya menjadi hanya kebijakan penskalaan langkah menggunakan parameter `--query`. Untuk informasi lebih lanjut tentang sintaks untuk `query`, lihat Mengontrol output perintah dari AWS CLI di AWS Command Line Interface Panduan Pengguna.

Linux, macOS, atau Unix

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs \
  --query 'ScalingPolicies[?PolicyType==`StepScaling`]'
```

Windows

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs ^
  --query "ScalingPolicies[?PolicyType==`StepScaling`]"
```

Output

Berikut ini adalah output contoh.

```
[
    {
        "PolicyARN": "PolicyARN",
        "StepScalingPolicyConfiguration": {
            "MetricAggregationType": "Average",
            "Cooldown": 60,
            "StepAdjustments": [
                {
                    "MetricIntervalLowerBound": 0.0,
                    "MetricIntervalUpperBound": 15.0,
                    "ScalingAdjustment": 1
                },
                {
                    "MetricIntervalLowerBound": 15.0,
                    "MetricIntervalUpperBound": 25.0,
                    "ScalingAdjustment": 2
                },
                {
                    "MetricIntervalLowerBound": 25.0,
                    "ScalingAdjustment": 3
                }
            ],
            "AdjustmentType": "ChangeInCapacity"
        },
        "PolicyType": "StepScaling",
        "ResourceId": "service/my-cluster/my-service",
        "ServiceNamespace": "ecs",
        "Alarms": [
            {
                "AlarmName": "Step-Scaling-AlarmHigh-ECS:service/my-cluster/my-
service",
                "AlarmARN": "arn:aws:cloudwatch:region:012345678910:alarm:Step-Scaling-
AlarmHigh-ECS:service/my-cluster/my-service"
            }
        ],
        "PolicyName": "my-step-scaling-policy",
        "ScalableDimension": "ecs:service:DesiredCount",
        "CreationTime": 1515024099.901
    }
]
```

Menghapus kebijakan penskalaan langkah untuk Application Auto Scaling menggunakan AWS CLI

Saat Anda tidak lagi memerlukan suatu kebijakan penskalaan langkah, Anda dapat menghapusnya. Untuk menghapus kebijakan penskalaan dan CloudWatch alarm terkait, selesaikan tugas berikut.

Untuk menghapus kebijakan penskalaan Anda

Gunakan perintah delete-scaling-policy.

Linux, macOS, atau Unix

```
aws application-autoscaling delete-scaling-policy --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --policy-name my-step-scaling-policy
```

Windows

```
aws application-autoscaling delete-scaling-policy --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
  --resource-id service/my-cluster/my-service ^
  --policy-name my-step-scaling-policy
```

Untuk menghapus CloudWatch alarm

Gunakan perintah hapus-alarm. Anda dapat menghapus satu atau beberapa alarm sekaligus. Misalnya, gunakan perintah berikut untuk menghapus alarm `Step-Scaling-AlarmHigh-ECS:service/my-cluster/my-service` dan `Step-Scaling-AlarmLow-ECS:service/my-cluster/my-service`.

```
aws cloudwatch delete-alarms --alarm-name Step-Scaling-AlarmHigh-ECS:service/my-
cluster/my-service Step-Scaling-AlarmLow-ECS:service/my-cluster/my-service
```

Penskalaan prediktif untuk Application Auto Scaling

Penskalaan prediktif secara proaktif menskalakan aplikasi Anda. Penskalaan prediktif menganalisis data beban historis untuk mendeteksi pola harian atau mingguan dalam arus lalu lintas. Ini menggunakan informasi ini untuk memperkirakan kebutuhan kapasitas masa depan untuk secara proaktif meningkatkan kapasitas aplikasi Anda agar sesuai dengan beban yang diantisipasi.

Penskalaan prediktif sangat cocok untuk situasi di mana Anda memiliki:

- Lalu lintas siklus, seperti penggunaan sumber daya yang tinggi selama jam kerja reguler dan penggunaan sumber daya yang rendah selama malam hari dan akhir pekan
- Pola on-and-off beban kerja berulang, seperti pemrosesan batch, pengujian, atau analisis data berkala.
- Aplikasi yang membutuhkan waktu lama untuk diinisialisasi, menyebabkan dampak latensi yang nyata pada kinerja aplikasi selama peristiwa penskalaan

Daftar Isi

Cara kerja penskalaan prediktif Application Auto Scaling

Untuk menggunakan penskalaan prediktif, buat kebijakan penskalaan prediktif yang menentukan CloudWatch metrik untuk dipantau dan dianalisis. Anda dapat menggunakan metrik yang telah ditentukan atau metrik khusus. Agar penskalaan prediktif mulai meramalkan nilai masa depan, metrik ini harus memiliki setidaknya 24 jam data.

Setelah Anda membuat kebijakan, penskalaan prediktif mulai menganalisis data metrik hingga 14 hari terakhir untuk mengidentifikasi pola. Ini menggunakan analisis ini untuk menghasilkan perkiraan per jam persyaratan kapasitas untuk 48 jam ke depan. Prakiraan diperbarui setiap 6 jam menggunakan CloudWatch data terbaru. Saat data baru masuk, penskalaan prediktif mampu terus meningkatkan akurasi prakiraan masa depan.

Pertama-tama Anda dapat mengaktifkan penskalaan prediktif dalam mode perkiraan saja. Dalam mode ini, ini menghasilkan perkiraan kapasitas tetapi tidak benar-benar menskalakan kapasitas Anda berdasarkan perkiraan tersebut. Ini memungkinkan Anda untuk mengevaluasi keakuratan dan kesesuaian ramalan.

Setelah Anda meninjau data perkiraan dan memutuskan untuk memulai penskalaan berdasarkan data tersebut, alihkan kebijakan penskalaan ke mode perkiraan dan skala. Dalam mode ini:

- Jika perkiraan mengharapkan peningkatan beban, penskalaan prediktif akan meningkatkan kapasitas.
- Jika perkiraan mengharapkan penurunan beban, penskalaan prediktif tidak akan menskalakan untuk menghilangkan kapasitas. Ini memastikan bahwa Anda meningkatkan skala hanya ketika permintaan benar-benar turun, dan tidak hanya pada prediksi. Untuk menghapus kapasitas yang tidak lagi diperlukan, Anda harus membuat kebijakan Pelacakan Target atau Penskalaan Langkah karena mereka merespons data metrik waktu nyata.

Secara default, penskalaan prediktif menskalakan target Anda yang dapat diskalakan pada awal setiap jam berdasarkan perkiraan untuk jam tersebut. Anda dapat menentukan waktu mulai yang lebih awal secara opsional dengan menggunakan `SchedulingBufferTime` properti dalam operasi `PutScalingPolicy` API. Ini memungkinkan Anda untuk meluncurkan kapasitas yang diprediksi sebelum permintaan yang diperkirakan, yang memberikan kapasitas baru waktu yang cukup untuk siap menangani lalu lintas.

Batas kapasitas maksimum

Secara default, ketika kebijakan penskalaan ditetapkan, mereka tidak dapat meningkatkan kapasitas lebih tinggi dari kapasitas maksimumnya.

Atau, Anda dapat mengizinkan kapasitas maksimum target yang dapat diskalakan ditingkatkan secara otomatis jika kapasitas perkiraan mendekati atau melebihi kapasitas maksimum target yang dapat diskalakan. Untuk mengaktifkan perilaku ini, gunakan `MaxCapacityBuffer` properti `MaxCapacityBreachBehavior` dan dalam operasi `PutScalingPolicy` API atau setelan perilaku kapasitas Maks di Konsol Manajemen AWS.

> **Warning**
>
> Berhati-hatilah saat memungkinkan kapasitas maksimum ditingkatkan secara otomatis. Kapasitas maksimum tidak secara otomatis berkurang kembali ke maksimum asli.

Perintah yang umum digunakan untuk penskalaan pembuatan kebijakan, manajemen, dan penghapusan

Perintah yang umum digunakan untuk bekerja dengan kebijakan penskalaan prediktif meliputi:

- `register-scalable-target`untuk mendaftarkan AWS atau menyesuaikan sumber daya sebagai target yang dapat diskalakan, untuk menangguhkan penskalaan, dan untuk melanjutkan penskalaan.
- `put-scaling-policy`untuk membuat kebijakan penskalaan prediktif.
- `get-predictive-scaling-forecast`untuk mengambil data perkiraan untuk kebijakan penskalaan prediktif.
- `describe-scaling-activities`untuk mengembalikan informasi tentang aktivitas penskalaan dalam file AWS Region.
- `describe-scaling-policies`untuk mengembalikan informasi tentang kebijakan penskalaan dalam file AWS Region.
- `delete-scaling-policy`untuk menghapus kebijakan penskalaan.

Metrik-metrik kustom

Metrik khusus dapat digunakan untuk memprediksi kapasitas yang dibutuhkan untuk suatu aplikasi. Metrik kustom berguna ketika metrik yang telah ditentukan tidak cukup untuk menangkap beban pada aplikasi Anda.

Pertimbangan-pertimbangan

Pertimbangan berikut berlaku saat bekerja dengan penskalaan prediktif.

- Konfirmasikan apakah penskalaan prediktif cocok untuk aplikasi Anda. Aplikasi sangat cocok untuk penskalaan prediktif jika menunjukkan pola beban berulang yang spesifik untuk hari dalam seminggu atau waktu dalam sehari. Evaluasi prakiraan sebelum membiarkan penskalaan prediktif secara aktif menskalakan aplikasi Anda.
- Penskalaan prediktif membutuhkan setidaknya 24 jam data historis untuk memulai peramalan. Namun, perkiraan lebih efektif jika data historis mencakup dua minggu penuh.
- Pilih metrik pemuatan yang secara akurat mewakili beban penuh pada aplikasi Anda dan merupakan aspek aplikasi Anda yang paling penting untuk diskalakan.

Membuat kebijakan penskalaan prediktif untuk Application Auto Scaling

Kebijakan contoh berikut menggunakan kebijakan AWS CLI untuk mengonfigurasi kebijakan penskalaan prediktif untuk layanan Amazon ECS. Ganti masing-masing *user input placeholder* dengan informasi Anda sendiri.

Untuk informasi selengkapnya tentang CloudWatch metrik yang dapat Anda tentukan, lihat PredictiveScalingMetricSpecificationdi Referensi API Amazon EC2 Auto Scaling.

Berikut ini adalah contoh kebijakan dengan konfigurasi memori yang telah ditentukan.

```
cat policy.json
{
    "MetricSpecifications": [
        {
            "TargetValue": 40,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ECSServiceMemoryUtilization"
            }
        }
    ],
    "SchedulingBufferTime": 3600,
    "MaxCapacityBreachBehavior": "HonorMaxCapacity",
    "Mode": "ForecastOnly"
}
```

Contoh berikut mengilustrasikan pembuatan kebijakan dengan menjalankan put-scaling-policyperintah dengan file konfigurasi yang ditentukan.

```
aws aas put-scaling-policy \
--service-namespace ecs \
--region us-east-1 \
--policy-name predictive-scaling-policy-example \
--resource-id service/MyCluster/test \
--policy-type PredictiveScaling \
--scalable-dimension ecs:service:DesiredCount \
--predictive-scaling-policy-configuration file://policy.json
```

Jika berhasil, perintah ini mengembalikan ARN kebijakan.

```
{
"PolicyARN": "arn:aws:autoscaling:us-
east-1:012345678912:scalingPolicy:d1d72dfe-5fd3-464f-83cf-824f16cb88b7:resource/ecs/
service/MyCluster/test:policyName/predictive-scaling-policy-example",
"Alarms": []
}
```

Ganti nilai perkiraan menggunakan tindakan terjadwal

Terkadang, Anda mungkin memiliki informasi tambahan tentang persyaratan aplikasi future Anda yang tidak dapat diperhitungkan oleh perhitungan perkiraan. Misalnya, perhitungan perkiraan mungkin meremehkan kapasitas yang dibutuhkan untuk acara pemasaran yang akan datang. Anda dapat menggunakan tindakan terjadwal untuk mengganti perkiraan sementara selama periode waktu mendatang. Tindakan terjadwal dapat berjalan secara berulang, atau pada tanggal dan waktu tertentu ketika ada fluktuasi permintaan satu kali.

Misalnya, Anda dapat membuat tindakan terjadwal dengan kapasitas minimum yang lebih tinggi dari yang diperkirakan. Saat runtime, Application Auto Scaling memperbarui kapasitas minimum target yang dapat diskalakan. Karena penskalaan prediktif mengoptimalkan kapasitas, tindakan terjadwal dengan kapasitas minimum yang lebih tinggi dari nilai perkiraan dihormati. Ini mencegah kapasitas menjadi kurang dari yang diharapkan. Untuk berhenti mengesampingkan perkiraan, gunakan tindakan terjadwal kedua untuk mengembalikan kapasitas minimum ke pengaturan aslinya.

Prosedur berikut menguraikan langkah-langkah untuk mengesampingkan perkiraan selama periode waktu mendatang.

Topik

> **Important**
>
> Topik ini mengasumsikan bahwa Anda mencoba mengesampingkan perkiraan untuk skala ke kapasitas yang lebih tinggi daripada yang diperkirakan. Jika Anda perlu mengurangi kapasitas sementara tanpa campur tangan dari kebijakan penskalaan prediktif, gunakan mode perkiraan saja. Sementara dalam mode perkiraan saja, penskalaan prediktif akan terus menghasilkan perkiraan, tetapi tidak akan secara otomatis meningkatkan kapasitas.

Anda kemudian dapat memantau pemanfaatan sumber daya dan secara manual mengurangi ukuran grup Anda sesuai kebutuhan.

Langkah 1: (Opsional) Analisis data deret waktu

Mulailah dengan menganalisis data deret waktu perkiraan. Ini adalah langkah opsional, tetapi akan sangat membantu jika Anda ingin memahami detail perkiraan.

1. Ambil ramalan

   Setelah perkiraan dibuat, Anda dapat menanyakan periode waktu tertentu dalam perkiraan. Tujuan dari kueri ini adalah untuk mendapatkan tampilan lengkap dari data deret waktu untuk periode waktu tertentu.

   Kueri Anda dapat mencakup hingga dua hari data perkiraan masa depan. Jika Anda telah menggunakan penskalaan prediktif untuk sementara waktu, Anda juga dapat mengakses data perkiraan sebelumnya. Namun, durasi waktu maksimum antara waktu mulai dan akhir adalah 30 hari.

   Untuk mengambil ramalan, gunakan get-predictive-scaling-forecastperintah. Contoh berikut mendapatkan perkiraan penskalaan prediktif untuk layanan Amazon ECS.

   ```
   aws application-autoscaling get-predictive-scaling-forecast --service-namespace ecs
    \
       --scalable-dimension ecs:service:DesiredCount \
       --resource-id 1234567890abcdef0
       --policy-name predictive-scaling-policy \
       --start-time "2021-05-19T17:00:00Z" \
       --end-time "2021-05-19T23:00:00Z"
   ```

   Tanggapan tersebut mencakup dua prakiraan: `LoadForecast` dan`CapacityForecast`. `LoadForecast`menunjukkan perkiraan beban per jam. `CapacityForecast`menunjukkan nilai perkiraan untuk kapasitas yang dibutuhkan setiap jam untuk menangani beban yang diperkirakan sambil mempertahankan yang ditentukan. `TargetValue`

2. Identifikasi periode waktu target

   Identifikasi jam atau jam ketika fluktuasi permintaan satu kali harus terjadi. Ingat bahwa tanggal dan waktu yang ditunjukkan dalam perkiraan ada di UTC.

Langkah 2: Buat dua tindakan terjadwal

Selanjutnya, buat dua tindakan terjadwal untuk periode waktu tertentu ketika aplikasi Anda akan memiliki beban yang lebih tinggi dari perkiraan. Misalnya, jika Anda memiliki acara pemasaran yang akan mengarahkan lalu lintas ke situs Anda untuk jangka waktu terbatas, Anda dapat menjadwalkan tindakan satu kali untuk memperbarui kapasitas minimum saat dimulai. Kemudian, jadwalkan tindakan lain untuk mengembalikan kapasitas minimum ke pengaturan asli saat acara berakhir.

Untuk membuat dua tindakan terjadwal untuk acara satu kali ()AWS CLI

Untuk membuat tindakan yang dijadwalkan, gunakan put-scheduled-actionperintah.

Contoh berikut mendefinisikan jadwal untuk Amazon EC2 Auto Scaling yang mempertahankan kapasitas minimum tiga instans pada 19 Mei pukul 17:00 selama delapan jam. Perintah berikut menunjukkan bagaimana menerapkan skenario ini.

Perintah put-scheduled-update-group-action pertama menginstruksikan Amazon EC2 Auto Scaling untuk memperbarui kapasitas minimum grup Auto Scaling yang ditentukan pada pukul 17:00 UTC pada 19 Mei 2021.

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-
start \
  --auto-scaling-group-name my-asg --start-time "2021-05-19T17:00:00Z" --minimum-
capacity 3
```

Perintah kedua menginstruksikan Amazon EC2 Auto Scaling untuk mengatur kapasitas minimum grup menjadi satu pada 1:00 UTC pada 20 Mei 2021.

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-end
 \
  --auto-scaling-group-name my-asg --start-time "2021-05-20T01:00:00Z" --minimum-
capacity 1
```

Setelah Anda menambahkan tindakan terjadwal ini ke grup Auto Scaling, Amazon EC2 Auto Scaling melakukan hal berikut:

- Pada pukul 17:00 UTC pada 19 Mei 2021, aksi terjadwal pertama berjalan. Jika grup saat ini memiliki kurang dari tiga instance, grup tersebut menskalakan menjadi tiga instance. Selama waktu ini dan selama delapan jam ke depan, Amazon EC2 Auto Scaling dapat terus ditingkatkan jika

kapasitas yang diprediksi lebih tinggi dari kapasitas aktual atau jika ada kebijakan penskalaan dinamis yang berlaku.

- Pukul 1:00 UTC pada 20 Mei 2021, aksi terjadwal kedua berjalan. Ini mengembalikan kapasitas minimum ke pengaturan aslinya di akhir acara.

Penskalaan berdasarkan jadwal berulang

Untuk mengganti perkiraan untuk periode waktu yang sama setiap minggu, buat dua tindakan terjadwal dan berikan logika waktu dan tanggal menggunakan ekspresi cron.

Format ekspresi cron terdiri dari lima bidang yang dipisahkan oleh spasi: [Minute] [Hour] [Day_of_month] [Month_of_year] [day_of_week]. Bidang dapat berisi nilai apa pun yang diizinkan, termasuk karakter khusus.

Misalnya, ekspresi cron berikut menjalankan aksi setiap hari Selasa pukul 6:30 pagi. Tanda bintang digunakan sebagai wildcard untuk mencocokkan semua nilai untuk bidang.

```
30 6 * * 2
```

Kebijakan penskalaan prediktif lanjutan menggunakan metrik khusus

Dalam kebijakan penskalaan prediktif, Anda dapat menggunakan metrik yang telah ditentukan atau kustom. Metrik kustom berguna ketika metrik yang telah ditentukan tidak cukup menggambarkan pemuatan aplikasi Anda.

Saat membuat kebijakan penskalaan prediktif dengan metrik kustom, Anda dapat menentukan CloudWatch metrik lain yang disediakan oleh AWS, atau Anda dapat menentukan metrik yang Anda tentukan dan publikasikan sendiri. Anda juga dapat menggunakan matematika metrik untuk menggabungkan dan mengubah metrik yang ada menjadi deret waktu baru yang AWS tidak dilacak secara otomatis. Ketika Anda menggabungkan nilai dalam data Anda, misalnya, dengan menghitung jumlah atau rata-rata baru, itu disebut agregasi. Data yang dihasilkan disebut agregat.

Bagian berikut berisi praktik terbaik dan contoh bagaimana membangun struktur JSON untuk kebijakan tersebut.

Topik

Praktik terbaik

Praktik terbaik berikut dapat membantu Anda menggunakan metrik kustom secara lebih efektif:

- Untuk spesifikasi metrik beban, metrik yang paling berguna adalah metrik yang mewakili beban pada aplikasi Anda.
- Metrik penskalaan harus berbanding terbalik dengan kapasitas. Artinya, jika target yang dapat diskalakan meningkat, metrik penskalaan akan berkurang dengan proporsi yang kira-kira sama. Untuk memastikan bahwa penskalaan prediktif berperilaku seperti yang diharapkan, metrik beban dan metrik penskalaan juga harus berkorelasi kuat satu sama lain.
- Pemanfaatan target harus sesuai dengan jenis metrik penskalaan. Untuk konfigurasi kebijakan yang menggunakan pemanfaatan CPU, ini adalah persentase target. Untuk konfigurasi kebijakan yang menggunakan throughput, seperti jumlah permintaan atau pesan, ini adalah jumlah target permintaan atau pesan per instance selama interval satu menit.
- Jika rekomendasi ini tidak diikuti, nilai future yang diperkirakan dari deret waktu mungkin akan salah. Untuk memvalidasi bahwa data sudah benar, Anda dapat melihat nilai yang diperkirakan. Atau, setelah Anda membuat kebijakan penskalaan prediktif, periksa `LoadForecast` dan `CapacityForecast` objek yang ditampilkan oleh panggilan ke API. GetPredictiveScalingForecast
- Kami sangat menyarankan agar Anda mengonfigurasi penskalaan prediktif dalam mode hanya perkiraan sehingga Anda dapat mengevaluasi perkiraan sebelum penskalaan prediktif mulai secara aktif menskalakan kapasitas.

Prasyarat

Untuk menambahkan metrik khusus ke kebijakan penskalaan prediktif, Anda harus memiliki izin. `cloudwatch:GetMetricData`

Untuk menentukan metrik Anda sendiri, bukan metrik yang AWS disediakan, Anda harus terlebih dahulu mempublikasikan metrik Anda. CloudWatch Untuk informasi selengkapnya, lihat Menerbitkan metrik kustom di Panduan CloudWatch Pengguna Amazon.

Jika Anda mempublikasikan metrik Anda sendiri, pastikan untuk mempublikasikan titik data pada frekuensi minimum lima menit. Application Auto Scaling mengambil titik data CloudWatch berdasarkan panjang periode yang dibutuhkannya. Misalnya, spesifikasi metrik beban menggunakan metrik per jam untuk mengukur beban pada aplikasi Anda. CloudWatch menggunakan data metrik yang Anda publikasikan untuk memberikan nilai data tunggal untuk periode satu jam dengan menggabungkan semua titik data dengan stempel waktu yang termasuk dalam setiap periode satu jam.

Membangun JSON untuk metrik khusus

Bagian berikut berisi contoh cara mengonfigurasi penskalaan prediktif ke data kueri dari Amazon EC2 CloudWatch Auto Scaling. Ada dua metode berbeda untuk mengonfigurasi opsi ini, dan metode yang Anda pilih memengaruhi format mana yang Anda gunakan untuk membuat JSON untuk kebijakan penskalaan prediktif Anda. Saat Anda menggunakan matematika metrik, format JSON bervariasi lebih lanjut berdasarkan matematika metrik yang dilakukan.

1. Untuk membuat kebijakan yang mendapatkan data langsung dari CloudWatch metrik lain yang disediakan oleh AWS atau metrik yang Anda publikasikan CloudWatch, lihat. Contoh kebijakan penskalaan prediktif dengan metrik pemuatan dan penskalaan khusus ()AWS CLI
2. Untuk membuat kebijakan yang dapat menanyakan beberapa CloudWatch metrik dan menggunakan ekspresi matematika untuk membuat deret waktu baru berdasarkan metrik ini, lihat. Gunakan ekspresi matematika metrik

Contoh kebijakan penskalaan prediktif dengan metrik pemuatan dan penskalaan khusus ()AWS CLI

Untuk membuat kebijakan penskalaan prediktif dengan metrik pemuatan dan penskalaan kustom dengan AWS CLI, simpan argumen `--predictive-scaling-configuration` dalam file JSON bernama. `config.json`

Anda mulai menambahkan metrik khusus dengan mengganti nilai yang dapat diganti dalam contoh berikut dengan metrik dan pemanfaatan target Anda.

```
{
  "MetricSpecifications": [
    {
      "TargetValue": 50,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
```

```
          {
            "Id": "scaling_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyUtilizationMetric",
                "Namespace": "MyNameSpace",
                "Dimensions": [
                  {
                    "Name": "MyOptionalMetricDimensionName",
                    "Value": "MyOptionalMetricDimensionValue"
                  }
                ]
              },
              "Stat": "Average"
            }
          }
        ]
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyLoadMetric",
                "Namespace": "MyNameSpace",
                "Dimensions": [
                  {
                    "Name": "MyOptionalMetricDimensionName",
                    "Value": "MyOptionalMetricDimensionValue"
                  }
                ]
              },
              "Stat": "Sum"
            }
          }
        ]
      }
    }
  ]
}
```

Untuk informasi lebih lanjut, lihat MetricDataQuery dalam Referensi API Amazon EC2 Auto Scaling.

> **Note**
>
> Berikut adalah beberapa sumber daya tambahan yang dapat membantu Anda menemukan nama metrik, ruang nama, dimensi, dan statistik untuk CloudWatch metrik:
>
> - Untuk informasi tentang metrik yang tersedia untuk AWS layanan, lihat AWS layanan yang memublikasikan CloudWatch metrik di CloudWatch Panduan Pengguna Amazon.
> - Untuk mendapatkan nama metrik, namespace, dan dimensi yang tepat (jika ada) untuk CloudWatch metrik dengan metrik AWS CLI, lihat daftar-metrik.

Untuk membuat kebijakan ini, jalankan put-scaling-policyperintah menggunakan file JSON sebagai input, seperti yang ditunjukkan dalam contoh berikut.

```
aws autoscaling put-scaling-policy --policy-name my-predictive-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

Jika berhasil, perintah ini mengembalikan Amazon Resource Name (ARN) kebijakan.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-predictive-scaling-policy",
  "Alarms": []
}
```

Gunakan ekspresi matematika metrik

Bagian berikut memberikan informasi dan contoh kebijakan penskalaan prediktif yang menunjukkan bagaimana Anda dapat menggunakan matematika metrik dalam kebijakan Anda.

Topik

- Memahami matematika metrik
- Contoh kebijakan penskalaan prediktif untuk Amazon EC2 Auto Scaling yang menggabungkan metrik menggunakan metrik matematika ()AWS CLI
- Contoh kebijakan penskalaan prediktif yang akan digunakan dalam skenario blue/green penerapan ()AWS CLI

Memahami matematika metrik

Jika yang ingin Anda lakukan hanyalah mengumpulkan data metrik yang ada, matematika CloudWatch metrik menghemat upaya dan biaya penerbitan metrik lain. CloudWatch Anda dapat menggunakan metrik apa pun yang AWS menyediakan, dan Anda juga dapat menggunakan metrik yang Anda tetapkan sebagai bagian dari aplikasi Anda.

Untuk informasi selengkapnya, lihat Menggunakan matematika metrik di Panduan CloudWatch Pengguna Amazon.

Jika Anda memilih untuk menggunakan ekspresi matematika metrik dalam kebijakan penskalaan prediktif Anda, pertimbangkan poin-poin berikut:

- Operasi matematika metrik menggunakan titik data dari kombinasi unik nama metrik, namespace, dan kunci dimensi/pasangan nilai metrik.
- Anda dapat menggunakan operator aritmatika (+ - */^), fungsi statistik (seperti AVG atau SUM), atau fungsi lain yang mendukung. CloudWatch
- Anda dapat menggunakan metrik dan hasil ekspresi matematika lainnya dalam rumus ekspresi matematika.
- Ekspresi matematika metrik Anda dapat terdiri dari agregasi yang berbeda. Namun, ini adalah praktik terbaik untuk hasil agregasi akhir yang digunakan `Average` untuk metrik penskalaan dan `Sum` untuk metrik beban.
- Setiap ekspresi yang digunakan dalam spesifikasi metrik pada akhirnya harus mengembalikan satu deret waktu.

Untuk menggunakan matematika metrik, lakukan hal berikut:

- Pilih satu atau beberapa CloudWatch metrik. Kemudian, buat ekspresi. Untuk informasi selengkapnya, lihat Menggunakan matematika metrik di Panduan CloudWatch Pengguna Amazon.
- Verifikasi bahwa ekspresi matematika metrik valid dengan menggunakan CloudWatch konsol atau CloudWatch GetMetricDataAPI.

Contoh kebijakan penskalaan prediktif untuk Amazon EC2 Auto Scaling yang menggabungkan metrik menggunakan metrik matematika ()AWS CLI

Terkadang, alih-alih menentukan metrik secara langsung, Anda mungkin perlu terlebih dahulu memproses datanya dengan cara tertentu. Misalnya, Anda mungkin memiliki aplikasi yang menarik

pekerjaan dari antrean Amazon SQS, dan Anda mungkin ingin menggunakan jumlah item dalam antrian sebagai kriteria untuk penskalaan prediktif. Jumlah pesan dalam antrian tidak hanya menentukan jumlah instance yang Anda butuhkan. Oleh karena itu, lebih banyak pekerjaan diperlukan untuk membuat metrik yang dapat digunakan untuk menghitung backlog per instance.

Berikut ini adalah contoh kebijakan penskalaan prediktif untuk skenario ini. Ini menentukan metrik penskalaan dan pemuatan yang didasarkan pada metrik Amazon SQS`ApproximateNumberOfMessagesVisible`, yang merupakan jumlah pesan yang tersedia untuk diambil dari antrian. Ini juga menggunakan metrik Amazon EC2 `GroupInServiceInstances` Auto Scaling dan ekspresi matematika untuk menghitung backlog per instance untuk metrik penskalaan.

```
aws autoscaling put-scaling-policy --policy-name my-sqs-custom-metrics-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
{
  "MetricSpecifications": [
    {
      "TargetValue": 100,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Label": "Get the queue size (the number of messages waiting to be
 processed)",
            "Id": "queue_size",
            "MetricStat": {
              "Metric": {
                "MetricName": "ApproximateNumberOfMessagesVisible",
                "Namespace": "AWS/SQS",
                "Dimensions": [
                  {
                    "Name": "QueueName",
                    "Value": "my-queue"
                  }
                ]
              },
              "Stat": "Sum"
            },
            "ReturnData": false
          },
          {
            "Label": "Get the group size (the number of running instances)",
```

```
            "Id": "running_capacity",
            "MetricStat": {
              "Metric": {
                "MetricName": "GroupInServiceInstances",
                "Namespace": "AWS/AutoScaling",
                "Dimensions": [
                  {
                    "Name": "AutoScalingGroupName",
                    "Value": "my-asg"
                  }
                ]
              },
              "Stat": "Sum"
            },
            "ReturnData": false
          },
          {
            "Label": "Calculate the backlog per instance",
            "Id": "scaling_metric",
            "Expression": "queue_size / running_capacity",
            "ReturnData": true
          }
        ]
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "ApproximateNumberOfMessagesVisible",
                "Namespace": "AWS/SQS",
                "Dimensions": [
                  {
                    "Name": "QueueName",
                    "Value": "my-queue"
                  }
                ],
              },
              "Stat": "Sum"
            },
            "ReturnData": true
          }
        ]
```

```
      }
    }
  ]
}
```

Contoh mengembalikan ARN kebijakan.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-sqs-custom-metrics-policy",
  "Alarms": []
}
```

Contoh kebijakan penskalaan prediktif yang akan digunakan dalam skenario blue/green penerapan ()AWS CLI

Ekspresi penelusuran menyediakan opsi lanjutan di mana Anda dapat melakukan kueri metrik dari beberapa grup Auto Scaling dan melakukan ekspresi matematika pada mereka. Ini sangat berguna untuk blue/green penerapan.

> **Note**
>
> Penerapan biru/hijau adalah metode penerapan di mana Anda membuat dua grup Auto Scaling yang terpisah namun identik. Hanya satu dari kelompok yang menerima lalu lintas produksi. Lalu lintas pengguna awalnya diarahkan ke grup Auto Scaling sebelumnya (“biru”), sedangkan grup baru (“hijau”) digunakan untuk pengujian dan evaluasi versi baru aplikasi atau layanan. Lalu lintas pengguna dialihkan ke grup Auto Scaling hijau setelah penerapan baru diuji dan diterima. Anda kemudian dapat menghapus grup biru setelah penerapan berhasil.

Saat grup Auto Scaling baru dibuat sebagai bagian dari blue/green penerapan, riwayat metrik setiap grup dapat secara otomatis disertakan dalam kebijakan penskalaan prediktif tanpa Anda harus mengubah spesifikasi metriknya. Untuk informasi selengkapnya, lihat Menggunakan kebijakan penskalaan prediktif EC2 Auto Scaling dengan penerapan Biru/Hijau di Blog Komputasi. AWS

Contoh kebijakan berikut menunjukkan bagaimana hal ini dapat dilakukan. Dalam contoh ini, kebijakan menggunakan `CPUUtilization` metrik yang dipancarkan oleh Amazon EC2. Ini menggunakan metrik Amazon EC2 `GroupInServiceInstances` Auto Scaling dan ekspresi

matematika untuk menghitung nilai metrik penskalaan per instance. Ini juga menentukan spesifikasi metrik kapasitas untuk mendapatkan `GroupInServiceInstances` metrik.

Ekspresi pencarian menemukan instance di beberapa grup Auto Scaling berdasarkan kriteria pencarian yang ditentukan. `CPUUtilization` Jika nanti Anda membuat grup Auto Scaling baru yang cocok dengan kriteria penelusuran yang sama, instance di grup Auto Scaling baru akan disertakan secara otomatis. `CPUUtilization`

```
aws autoscaling put-scaling-policy --policy-name my-blue-green-predictive-scaling-
policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
{
  "MetricSpecifications": [
    {
      "TargetValue": 25,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_sum",
            "Expression": "SUM(SEARCH('{AWS/EC2,AutoScalingGroupName} MetricName=
\"CPUUtilization\" ASG-myapp', 'Sum', 300))",
            "ReturnData": false
          },
          {
            "Id": "capacity_sum",
            "Expression": "SUM(SEARCH('{AWS/AutoScaling,AutoScalingGroupName}
 MetricName=\"GroupInServiceInstances\" ASG-myapp', 'Average', 300))",
            "ReturnData": false
          },
          {
            "Id": "weighted_average",
            "Expression": "load_sum / capacity_sum",
            "ReturnData": true
          }
        ]
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_sum",
            "Expression": "SUM(SEARCH('{AWS/EC2,AutoScalingGroupName} MetricName=
\"CPUUtilization\" ASG-myapp', 'Sum', 3600))"
```

```
          }
        ]
      },
      "CustomizedCapacityMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "capacity_sum",
            "Expression": "SUM(SEARCH('{AWS/AutoScaling,AutoScalingGroupName}
 MetricName=\"GroupInServiceInstances\" ASG-myapp', 'Average', 300))"
          }
        ]
      }
    }
  ]
}
```

Contoh mengembalikan ARN kebijakan.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-blue-green-predictive-
scaling-policy",
  "Alarms": []
}
```

Pertimbangan untuk metrik kustom dalam kebijakan penskalaan prediktif

Jika terjadi masalah saat menggunakan metrik kustom, kami sarankan Anda melakukan hal berikut:

- Jika pesan kesalahan disediakan, baca pesan dan selesaikan masalah yang dilaporkan, jika memungkinkan.
- Jika Anda tidak memvalidasi ekspresi sebelumnya, put-scaling-policyperintah memvalidasinya saat Anda membuat kebijakan penskalaan Anda. Namun, ada kemungkinan bahwa perintah ini mungkin gagal mengidentifikasi penyebab pasti dari kesalahan yang terdeteksi. Untuk memperbaiki masalah, pecahkan masalah kesalahan yang Anda terima dalam respons dari permintaan ke perintah. get-metric-data Anda juga dapat memecahkan masalah ekspresi dari konsol. CloudWatch
- Anda harus menentukan `false` untuk `ReturnData` jika `MetricDataQueries` menentukan fungsi SEARCH () sendiri tanpa fungsi matematika seperti SUM (). Ini karena ekspresi penelusuran

mungkin mengembalikan beberapa deret waktu, dan spesifikasi metrik berdasarkan ekspresi hanya dapat mengembalikan satu deret waktu.

- Semua metrik yang terlibat dalam ekspresi pencarian harus memiliki resolusi yang sama.

Batasan

Pembatasan berikut berlaku pada:

- Anda dapat menanyakan titik data hingga 10 metrik dalam satu spesifikasi metrik.
- Untuk tujuan batas ini, satu ekspresi dihitung sebagai satu metrik.

Tutorial: Konfigurasikan penskalaan otomatis untuk menangani beban kerja yang berat

Dalam tutorial ini, Anda mempelajari cara skala dan berdasarkan jendela waktu ketika aplikasi Anda akan memiliki beban kerja yang lebih berat dari biasanya. Ini sangat membantu ketika Anda memiliki aplikasi yang tiba-tiba dapat memiliki sejumlah besar pengunjung pada jadwal reguler atau secara musiman.

Anda dapat menggunakan kebijakan penskalaan pelacakan target bersama dengan penskalaan terjadwal untuk menangani beban tambahan. Skala terjadwal secara otomatis memulai perubahan pada `MinCapacity` dan `MaxCapacity` atas nama Anda, berdasarkan jadwal yang Anda tentukan. Ketika kebijakan penskalaan pelacakan target aktif pada sumber daya, kebijakan tersebut dapat menskalakan secara dinamis berdasarkan pemanfaatan sumber daya saat ini, dalam rentang kapasitas minimum dan maksimum yang baru.

Setelah menyelesaikan tutorial ini, Anda akan tahu cara:

- Gunakan penskalaan terjadwal untuk menambah kapasitas ekstra untuk memenuhi beban berat sebelum tiba, lalu lepaskan kapasitas ekstra saat tidak lagi diperlukan.
- Gunakan kebijakan penskalaan pelacakan target untuk menskalakan aplikasi Anda berdasarkan pemanfaatan sumber daya saat ini.

Daftar Isi

Prasyarat

Tutorial ini mengasumsikan bahwa Anda telah melakukan hal berikut:

- Dibuat sebuah Akun AWS.

- Menginstal dan mengkonfigurasi file AWS CLI.
- Memberikan izin yang diperlukan untuk mendaftarkan dan membatalkan pendaftaran sumber daya sebagai target yang dapat diskalakan dengan Application Auto Scaling. Selain itu, diberikan izin yang diperlukan untuk membuat kebijakan penskalaan dan tindakan terjadwal. Untuk informasi selengkapnya, lihat Identity and Access Management untuk Application Auto Scaling.
- Membuat sumber daya yang didukung di lingkungan non-produksi yang tersedia untuk digunakan untuk tutorial ini. Jika Anda belum memilikinya, buat sekarang. Untuk informasi tentang AWS layanan dan sumber daya yang bekerja dengan Application Auto Scaling, lihat bagian. Layanan AWS yang dapat Anda gunakan dengan Application Auto Scaling

> **Note**
>
> Saat menyelesaikan tutorial ini, ada dua langkah di mana Anda mengatur nilai kapasitas minimum dan maksimum sumber daya Anda ke 0 untuk mengatur ulang kapasitas saat ini ke 0. Bergantung pada sumber daya yang Anda gunakan dengan Application Auto Scaling, Anda mungkin tidak dapat mengatur ulang kapasitas saat ini ke 0 selama langkah-langkah ini. Untuk membantu Anda mengatasi masalah ini, pesan dalam output akan menunjukkan bahwa kapasitas minimum tidak boleh kurang dari nilai yang ditentukan dan akan memberikan nilai kapasitas minimum yang dapat diterima AWS sumber daya.

Langkah 1: Daftarkan target yang dapat diskalakan

Mulailah dengan meregistrasikan sumber daya Anda sebagai target yang dapat diskalakan dengan Application Auto Scaling. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling.

Untuk mendaftarkan target yang dapat diskalakan dengan Application Auto Scaling

- Gunakan berikut ini register-scalable-target untuk mendaftarkan target baru yang dapat diskalakan. Tetapkan `--min-capacity` dan `--max-capacity` ke 0 untuk mengatur ulang kapasitas saat ini ke 0.

  Ganti teks sampel `--service-namespace` dengan namespace AWS layanan yang Anda gunakan dengan Application Auto Scaling`--scalable-dimension`, dengan dimensi skalabel yang terkait dengan sumber daya yang Anda daftarkan, `--resource-id` dan dengan pengenal untuk sumber daya. Nilai-nilai ini bervariasi berdasarkan sumber daya yang digunakan dan

bagaimana ID sumber daya dibangun. Lihat topik di Layanan AWS yang dapat Anda gunakan dengan Application Auto Scaling bagian untuk informasi lebih lanjut. Topik ini mencakup contoh perintah yang menunjukkan cara mendaftarkan target yang dapat diskalakan dengan Application Auto Scaling.

Linux, macOS, atau Unix

```
aws application-autoscaling register-scalable-target \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --min-capacity 0 --max-capacity 0
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace namespace
 --scalable-dimension dimension --resource-id identifier --min-capacity 0 --max-
capacity 0
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-
id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Langkah 2: Siapkan tindakan terjadwal sesuai dengan kebutuhan Anda

Anda dapat menggunakan perintah put-scheduled-action untuk membuat tindakan terjadwal yang dikonfigurasi guna memenuhi kebutuhan bisnis Anda. Dalam tutorial ini, kami fokus pada konfigurasi yang berhenti mengkonsumsi sumber daya di luar jam kerja dengan mengurangi kapasitas menjadi 0.

Untuk membuat tindakan terjadwal yang akan disesuaikan di pagi hari

1. Untuk menskalakan target yang dapat diskalakan, gunakan perintah put-scheduled-action berikut. Sertakan parameter `--schedule` dengan jadwal berulang, dalam UTC, menggunakan ekspresi cron.

   Pada jadwal yang ditentukan (setiap hari pukul 9:00 pagi UTC), Application Auto Scaling memperbarui `MinCapacity` dan `MaxCapacity` ke rentang unit kapasitas 1-5 yang diinginkan.

   Linux, macOS, atau Unix

   ```
   aws application-autoscaling put-scheduled-action \
     --service-namespace namespace \
     --scalable-dimension dimension \
     --resource-id identifier \
     --scheduled-action-name my-first-scheduled-action \
     --schedule "cron(0 9 * * ? *)" \
     --scalable-target-action MinCapacity=1,MaxCapacity=5
   ```

   Windows

   ```
   aws application-autoscaling put-scheduled-action --service-namespace namespace --
   scalable-dimension dimension --resource-id identifier --scheduled-action-name my-
   first-scheduled-action --schedule "cron(0 9 * * ? *)" --scalable-target-action
    MinCapacity=1,MaxCapacity=5
   ```

   Perintah ini tidak akan memberikan output jika berhasil.

2. Untuk mengonfirmasi bahwa tindakan terjadwal Anda tersedia, gunakan perintah describe-scheduled-actions berikut.

   Linux, macOS, atau Unix

   ```
   aws application-autoscaling describe-scheduled-actions \
     --service-namespace namespace \
     --query 'ScheduledActions[?ResourceId==`identifier`]'
   ```

   Windows

```
aws application-autoscaling describe-scheduled-actions --service-
namespace namespace --query "ScheduledActions[?ResourceId==`identifier`]"
```

Berikut ini adalah contoh output.

```
[
    {
        "ScheduledActionName": "my-first-scheduled-action",
        "ScheduledActionARN": "arn",
        "Schedule": "cron(0 9 * * ? *)",
        "ScalableTargetAction": {
            "MinCapacity": 1,
            "MaxCapacity": 5
        },
        ...
    }
]
```

Untuk membuat jadwal tindakan pembesaran skala di malam hari

1. Ulangi prosedur sebelumnya untuk membuat tindakan terjadwal lainnya yang digunakan Application Auto Scaling untuk memperbesar skala di akhir hari.

   Pada jadwal yang ditentukan (setiap hari pukul 8.00 malam UTC), Application Auto Scaling memperbarui `MinCapacity` dan `MaxCapacity` hingga 0, sebagaimana diinstruksikan oleh perintah put-scheduled-action.

   Linux, macOS, atau Unix

   ```
   aws application-autoscaling put-scheduled-action \
     --service-namespace namespace \
     --scalable-dimension dimension \
     --resource-id identifier \
     --scheduled-action-name my-second-scheduled-action \
     --schedule "cron(0 20 * * ? *)" \
     --scalable-target-action MinCapacity=0,MaxCapacity=0
   ```

   Windows

```
aws application-autoscaling put-scheduled-action --service-namespace namespace --
scalable-dimension dimension --resource-id identifier --scheduled-action-name my-
second-scheduled-action --schedule "cron(0 20 * * ? *)" --scalable-target-action
 MinCapacity=0,MaxCapacity=0
```

2. Untuk mengonfirmasi bahwa tindakan terjadwal Anda tersedia, gunakan perintah describe-scheduled-actions berikut.

   Linux, macOS, atau Unix

   ```
   aws application-autoscaling describe-scheduled-actions \
     --service-namespace namespace \
     --query 'ScheduledActions[?ResourceId==`identifier`]'
   ```

   Windows

   ```
   aws application-autoscaling describe-scheduled-actions --service-
   namespace namespace --query "ScheduledActions[?ResourceId==`identifier`]"
   ```

   Berikut ini adalah output contoh.

   ```
   [
       {
           "ScheduledActionName": "my-first-scheduled-action",
           "ScheduledActionARN": "arn",
           "Schedule": "cron(0 9 * * ? *)",
           "ScalableTargetAction": {
               "MinCapacity": 1,
               "MaxCapacity": 5
           },
           ...
       },
       {
           "ScheduledActionName": "my-second-scheduled-action",
           "ScheduledActionARN": "arn",
           "Schedule": "cron(0 20 * * ? *)",
           "ScalableTargetAction": {
               "MinCapacity": 0,
               "MaxCapacity": 0
           },
           ...
   ```

```
    }
]
```

Langkah 3: Tambahkan kebijakan penskalaan pelacakan target

Sekarang setelah Anda memiliki jadwal dasar, tambahkan kebijakan penskalaan pelacakan target ke skala berdasarkan pemanfaatan sumber daya saat ini.

Dengan pelacakan target, Application Auto Scaling membandingkan nilai target dalam kebijakan dengan nilai saat ini dari metrik yang ditentukan. Jika tidak sama untuk periode waktu tertentu, Application Auto Scaling akan menambah atau menghapus kapasitas untuk mempertahankan kinerja yang stabil. Saat beban pada aplikasi Anda dan nilai metrik meningkat, Application Auto Scaling menambah kapasitas secepat mungkin tanpa melebihi `MaxCapacity`. Ketika Application Auto Scaling menghapus kapasitas karena beban minimal, itu melakukannya tanpa di bawah `MinCapacity`. Dengan menyesuaikan kapasitas berdasarkan penggunaan, Anda hanya membayar untuk kebutuhan aplikasi Anda.

Jika metrik memiliki data yang tidak cukup karena aplikasi Anda tidak memiliki beban apa pun, Application Auto Scaling tidak menambah atau menghapus kapasitas. Dengan kata lain, Application Auto Scaling memprioritaskan ketersediaan dalam situasi di mana tidak cukup informasi yang tersedia.

Anda dapat menambahkan beberapa kebijakan penskalaan, tetapi pastikan Anda tidak menambahkan kebijakan penskalaan langkah yang bertentangan, yang dapat menyebabkan perilaku yang tidak diinginkan. Misalnya, jika kebijakan penskalaan langkah memulai aktivitas pembesaran skala sebelum kebijakan pelacakan target siap untuk memulai pembesaran skala, aktivitas pembesaran skala tidak akan diblokir. Setelah aktivitas scale-in selesai, kebijakan pelacakan target dapat menginstruksikan Application Auto Scaling untuk skala keluar lagi.

Untuk membuat kebijakan penskalaan pelacakan target

1. Gunakan perintah berikut ini put-scaling-policy untuk membuat kebijakan.

   Metrik yang paling sering digunakan untuk pelacakan target sudah ditentukan sebelumnya, dan Anda dapat menggunakannya tanpa menyediakan spesifikasi metrik lengkap dari. CloudWatch Untuk informasi selengkapnya tentang metrik standar yang tersedia, lihat. Kebijakan penskalaan pelacakan target untuk Application Auto Scaling

Sebelum menjalankan perintah ini, pastikan metrik yang sudah ditentukan sebelumnya menetapkan nilai target. Misalnya, untuk memperkecil skala ketika CPU mencapai 50% utilisasi, tentukan nilai target sebesar 50.0. Atau, untuk memperkecil skala konkurensi yang disediakan Lambda ketika penggunaan mencapai 70% pemanfaatan, tentukan nilai target sebesar 0,7. Untuk informasi tentang nilai target untuk sumber daya tertentu, lihat dokumentasi yang disediakan oleh layanan tentang cara mengonfigurasi pelacakan target. Untuk informasi selengkapnya, lihat Layanan AWS yang dapat Anda gunakan dengan Application Auto Scaling.

Linux, macOS, atau Unix

```
aws application-autoscaling put-scaling-policy \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --policy-name my-scaling-policy --policy-type TargetTrackingScaling \
  --target-tracking-scaling-policy-configuration '{ "TargetValue": 50.0,
 "PredefinedMetricSpecification": { "PredefinedMetricType": "predefinedmetric" }}'
```

Windows

```
aws application-autoscaling put-scaling-policy --service-namespace namespace --
scalable-dimension dimension --resource-id identifier --policy-name my-scaling-
policy --policy-type TargetTrackingScaling --target-tracking-scaling-policy-
configuration "{ \"TargetValue\": 50.0, \"PredefinedMetricSpecification\":
 { \"PredefinedMetricType\": \"predefinedmetric\" }}"
```

Jika berhasil, perintah ini mengembalikan ARNs dan nama dari dua CloudWatch alarm yang dibuat atas nama Anda.

2. Untuk mengonfirmasi bahwa tindakan terjadwal Anda tersedia, gunakan perintah describe-scaling-policies berikut.

Linux, macOS, atau Unix

```
aws application-autoscaling describe-scaling-policies --service-namespace namespace
 \
  --query 'ScalingPolicies[?ResourceId==`identifier`]'
```

Windows

```
aws application-autoscaling describe-scaling-policies --service-namespace namespace
 --query "ScalingPolicies[?ResourceId==`identifier`]"
```

Berikut ini adalah contoh output.

```
[
    {
        "PolicyARN": "arn",
        "TargetTrackingScalingPolicyConfiguration": {
            "PredefinedMetricSpecification": {
                "PredefinedMetricType": "predefinedmetric"
            },
            "TargetValue": 50.0
        },
        "PolicyName": "my-scaling-policy",
        "PolicyType": "TargetTrackingScaling",
        "Alarms": [],
        ...
    }
]
```

Langkah 4: Langkah selanjutnya

Saat aktivitas penskalaan terjadi, Anda akan melihat catatannya di output aktivitas penskalaan untuk target yang dapat diskalakan, misalnya:

```
Successfully set desired count to 1. Change successfully fulfilled by ecs.
```

Untuk memantau aktivitas penskalaan Anda dengan Application Auto Scaling, Anda dapat menggunakan perintah berikut. describe-scaling-activities

Linux, macOS, atau Unix

```
aws application-autoscaling describe-scaling-activities
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier
```

Windows

```
aws application-autoscaling describe-scaling-activities --service-namespace namespace
 --scalable-dimension dimension --resource-id identifier
```

Langkah 5: Bersihkan

Untuk mencegah akun Anda menyesuaikan biaya untuk sumber daya yang diciptakan saat penskalaan aktif, Anda dapat membersihkan konfigurasi penskalaan terkait sebagai berikut.

Menghapus konfigurasi penskalaan tidak menghapus sumber daya yang mendasarinya AWS . Ini juga tidak akan mengembalikannya ke kapasitas asli. Anda dapat menggunakan konsol layanan tempat Anda membuat sumber daya untuk menghapusnya atau menyesuaikan kapasitasnya.

Untuk menghapus tindakan terjadwal

Perintah berikut delete-scheduled-action menghapus tindakan terjadwal yang telah ditentukan. Anda dapat melewati langkah ini jika Anda ingin mempertahankan tindakan terjadwal yang Anda buat.

Linux, macOS, atau Unix

```
aws application-autoscaling delete-scheduled-action \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --scheduled-action-name my-second-scheduled-action
```

Windows

```
aws application-autoscaling delete-scheduled-action --service-namespace namespace
 --scalable-dimension dimension --resource-id identifier --scheduled-action-name my-
second-scheduled-action
```

Untuk menghapus kebijakan penskalaan

Perintah berikut delete-scaling-policy menghapus kebijakan penskalaan target tertentu. Anda dapat melewati langkah ini jika Anda ingin mempertahankan kebijakan penskalaan yang Anda buat.

Linux, macOS, atau Unix

```
aws application-autoscaling delete-scaling-policy \
  --service-namespace namespace \
```

```
  --scalable-dimension dimension \
  --resource-id identifier \
  --policy-name my-scaling-policy
```

Windows

```
aws application-autoscaling delete-scaling-policy --service-namespace namespace --
scalable-dimension dimension --resource-id identifier --policy-name my-scaling-policy
```

Untuk membatalkan pendaftaran target yang dapat diskalakan

Gunakan perintah deregister-scalable-target berikut untuk membatalkan pendaftaran target yang dapat diskalakan. Jika Anda memiliki kebijakan penskalaan yang Anda buat atau tindakan terjadwal yang belum dihapus, itu semua akan dihapus oleh perintah ini. Anda dapat melewati langkah ini jika Anda ingin menjaga target yang dapat diskalakan tetap terdaftar untuk penggunaan pada masa mendatang.

Linux, macOS, atau Unix

```
aws application-autoscaling deregister-scalable-target \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier
```

Windows

```
aws application-autoscaling deregister-scalable-target --service-namespace namespace --
scalable-dimension dimension --resource-id identifier
```

Menangguhkan dan melanjutkan penskalaan untuk Application Auto Scaling

Topik ini menjelaskan cara menangguhkan dan kemudian melanjutkan satu atau beberapa aktivitas penskalaan untuk target yang dapat diskalakan dalam aplikasi Anda. Fitur tangguhkan-lanjutkan digunakan untuk secara sementara menjeda aktivitas penskalaan yang dipicu oleh kebijakan penskalaan dan tindakan terjadwal Anda. Ini dapat berguna, misalnya, ketika Anda tidak ingin penskalaan otomatis berpotensi mengganggu saat Anda melakukan perubahan atau menyelidiki masalah konfigurasi. Kebijakan penskalaan dan tindakan terjadwal Anda dapat dipertahankan, dan saat Anda siap, aktivitas penskalaan dapat dilanjutkan.

Dalam contoh perintah CLI yang mengikuti, Anda meneruskan parameter berformat JSON dalam file config.json. Anda juga dapat memberikan parameter ini di baris perintah dengan menggunakan tanda petik untuk menyertakan struktur data JSON. Untuk informasi lebih lanjut, lihat Menggunakan tanda petik dengan string di AWS CLI dalam AWS Command Line Interface Panduan Pengguna.

Daftar Isi

> **Note**
>
> Untuk petunjuk penangguhan proses penskalaan saat penerapan Amazon ECS sedang berlangsung, lihat dokumentasi berikut:
> Servis penskalaan dan penerapan otomatis di Panduan Pengembang Layanan Amazon Elastic Container

Aktivitas penskalaan

Application Auto Scaling mendukung penetapan aktivitas penskalaan berikut dalam status ditangguhkan:

- Semua aktivitas penskalaan turun yang dipicu oleh kebijakan penskalaan.
- Semua aktivitas penskalaan naik yang dipicu oleh kebijakan penskalaan.

- Semua aktivitas penskalaan yang melibatkan tindakan terjadwal.

Deskripsi berikut menjelaskan apa yang terjadi saat tiap-tiap aktivitas penskalaan ditangguhkan. Tiap aktivitas dapat ditangguhkan dan dilanjutkan secara mandiri. Tergantung pada alasan untuk menangguhkan aktivitas penskalaan, Anda mungkin perlu menangguhkan beberapa aktivitas penskalaan sekaligus.

`DynamicScalingInSuspended`

- Application Auto Scaling tidak menghapus kapasitas ketika kebijakan penskalaan pelacakan target atau kebijakan penskalaan langkah dipicu. Ini memungkinkan Anda untuk menonaktifkan sementara aktivitas penskalaan turun yang terkait dengan kebijakan penskalaan tanpa menghapus kebijakan penskalaan atau alarm CloudWatch yang terkait dengannya. Ketika Anda melanjutkan penskalaan turun, Application Auto Scaling mengevaluasi kebijakan dengan ambang batas alarm yang saat ini dilanggar.

`DynamicScalingOutSuspended`

- Application Auto Scaling tidak menambah kapasitas ketika kebijakan penskalaan pelacakan target atau kebijakan penskalaan langkah dipicu. Hal ini memungkinkan Anda untuk menonaktifkan sementara aktivitas penskalaan naik yang terkait dengan kebijakan penskalaan tanpa menghapus kebijakan penskalaan atau alarm CloudWatch yang terkait dengannya. Ketika Anda melanjutkan penskalaan naik, Application Auto Scaling mengevaluasi kebijakan dengan ambang batas alarm yang saat ini dilanggar.

`ScheduledScalingSuspended`

- Application Auto Scaling tidak mengawali tindakan penskalaan yang dijadwalkan untuk dijalankan selama periode penangguhan. Ketika Anda melanjutkan penskalaan terjadwal, Application Auto Scaling hanya mengevaluasi tindakan terjadwal yang waktu pelaksanaannya belum berlalu.

Menangguhkan dan melanjutkan aktivitas penskalaan

Anda dapat menangguhkan dan melanjutkan aktivitas penskalaan individual atau semua aktivitas penskalaan untuk target Application Auto Scaling Anda yang dapat diskalakan.

> **Note**
>
> Demi keringkasan, contoh ini menggambarkan cara menangguhkan dan melanjutkan penskalaan untuk tabel DynamoDB. Untuk menentukan target yang dapat diskalakan yang berbeda, tentukan namespace-nya di `--service-namespace`, dimensinya yang dapat diskalakan di `--scalable-dimension`, dan ID sumber dayanya di `--resource-id`. Untuk informasi selengkapnya dan contoh untuk setiap layanan, lihat topik diLayanan AWS yang dapat Anda gunakan dengan Application Auto Scaling.

Untuk menangguhkan aktivitas penskalaan

Buka jendela baris perintah dan gunakan perintah register-scalable-target dengan opsi `--suspended-state` sebagai berikut.

Linux, macOS, atau Unix

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table --
suspended-state file://config.json
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Untuk hanya menangguhkan aktivitas penskalaan turun yang dipicu oleh kebijakan penskalaan, tentukan hal berikut dalam config.json.

```
{
    "DynamicScalingInSuspended":true
```

```
}
```

Untuk hanya menangguhkan aktivitas penskalaan naik yang dipicu oleh kebijakan penskalaan, tentukan hal berikut dalam config.json.

```
{
    "DynamicScalingOutSuspended":true
}
```

Untuk hanya menangguhkan aktivitas penskalaan yang melibatkan tindakan terjadwal, tentukan hal berikut dalam config.json.

```
{
    "ScheduledScalingSuspended":true
}
```

Untuk menangguhkan semua aktivitas penskalaan

Gunakan perintah register-scalable-target dengan opsi `--suspended-state` sebagai berikut.

Linux, macOS, atau Unix

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table --
suspended-state file://config.json
```

Contoh ini mengasumsikan bahwa file config.json berisi parameter yang diformat JSON berikut.

```
{
    "DynamicScalingInSuspended":true,
    "DynamicScalingOutSuspended":true,
    "ScheduledScalingSuspended":true
}
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Melihat aktivitas penskalaan yang ditangguhkan

Gunakan perintah describe-scalable-targets untuk menentukan aktivitas penskalaan mana yang berada dalam status ditangguhkan untuk target yang dapat diskalakan.

Linux, macOS, atau Unix

```
aws application-autoscaling describe-scalable-targets --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table
```

Windows

```
aws application-autoscaling describe-scalable-targets --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table
```

Berikut ini adalah contoh output.

```
{
    "ScalableTargets": [
        {
            "ServiceNamespace": "dynamodb",
            "ScalableDimension": "dynamodb:table:ReadCapacityUnits",
            "ResourceId": "table/my-table",
            "MinCapacity": 1,
            "MaxCapacity": 20,
            "SuspendedState": {
                "DynamicScalingOutSuspended": true,
                "DynamicScalingInSuspended": true,
                "ScheduledScalingSuspended": true
            },
            "CreationTime": 1558125758.957,
            "RoleARN": "arn:aws:iam::123456789012:role/aws-
service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable"
```

```
        }
    ]
}
```

Melanjutkan aktivitas penskalaan

Saat Anda siap untuk melanjutkan aktivitas penskalaan, Anda dapat melanjutkannya menggunakan perintah register-scalable-target.

Contoh perintah berikut melanjutkan semua aktivitas penskalaan untuk target yang dapat diskalakan tertentu.

Linux, macOS, atau Unix

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table --
suspended-state file://config.json
```

Contoh ini mengasumsikan bahwa file config.json berisi parameter yang diformat JSON berikut.

```
{
    "DynamicScalingInSuspended":false,
    "DynamicScalingOutSuspended":false,
    "ScheduledScalingSuspended":false
}
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Aktivitas penskalaan untuk Application Auto Scaling

Application Auto Scaling memantau CloudWatch metrik kebijakan penskalaan Anda dan memulai aktivitas penskalaan saat ambang batas terlampaui. Ini juga memulai aktivitas penskalaan saat Anda memodifikasi ukuran maksimum atau minimum target yang dapat diskalakan, baik secara manual atau mengikuti jadwal.

Ketika aktivitas penskalaan terjadi, Application Auto Scaling melakukan salah satu hal berikut:

- Meningkatkan kapasitas target yang dapat diskalakan (disebut sebagai scaling out)
- Mengurangi kapasitas target yang dapat diskalakan (disebut sebagai penskalaan)

Anda dapat mencari aktivitas penskalaan dari enam minggu terakhir.

Cari aktivitas penskalaan berdasarkan target yang dapat diskalakan

Untuk melihat aktivitas penskalaan target tertentu yang dapat diskalakan, gunakan perintah berikut describe-scaling-activities.

Linux, macOS, atau Unix

```
aws application-autoscaling describe-scaling-activities --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount --resource-id service/my-cluster/my-service
```

Windows

```
aws application-autoscaling describe-scaling-activities --service-namespace ecs --
scalable-dimension ecs:service:DesiredCount --resource-id service/my-cluster/my-service
```

Berikut ini adalah contoh respons, yang `StatusCode` berisi status aktivitas saat ini dan `StatusMessage` berisi informasi tentang status aktivitas penskalaan.

```
{
    "ScalingActivities": [
```

```
        {
            "ScalableDimension": "ecs:service:DesiredCount",
            "Description": "Setting desired count to 1.",
            "ResourceId": "service/my-cluster/my-service",
            "ActivityId": "e6c5f7d1-dbbb-4a3f-89b2-51f33e766399",
            "StartTime": 1462575838.171,
            "ServiceNamespace": "ecs",
            "EndTime": 1462575872.111,
            "Cause": "monitor alarm web-app-cpu-lt-25 in state ALARM triggered policy
 web-app-cpu-lt-25",
            "StatusMessage": "Successfully set desired count to 1. Change successfully
 fulfilled by ecs.",
            "StatusCode": "Successful"
        }
    ]
}
```

Untuk deskripsi bidang dalam respons, lihat ScalingActivitydi Referensi API Application Auto Scaling.

Kode status berikut menunjukkan kapan peristiwa penskalaan yang mengarah ke aktivitas penskalaan mencapai status selesai:

- `Successful`— Penskalaan berhasil diselesaikan
- `Overridden`— Kapasitas yang diinginkan diperbarui oleh acara penskalaan yang lebih baru
- `Unfulfilled`— Penskalaan waktunya habis atau layanan target tidak dapat memenuhi permintaan
- `Failed`— Penskalaan gagal dengan pengecualian

> **Note**
>
> Aktivitas penskalaan mungkin juga memiliki status `Pending` atau`InProgress`. Semua aktivitas penskalaan memiliki `Pending` status sebelum layanan target merespons. Setelah target merespons, status aktivitas penskalaan berubah menjadi. `InProgress`

Sertakan aktivitas yang tidak diskalakan

Secara default, aktivitas penskalaan tidak mencerminkan waktu ketika Application Auto Scaling membuat keputusan tentang apakah tidak akan menskalakan.

Misalnya, misalkan layanan Amazon ECS melebihi ambang batas maksimum metrik yang diberikan, tetapi jumlah tugas sudah mencapai jumlah maksimum tugas yang diizinkan. Dalam hal ini, Application Auto Scaling tidak mengurangi jumlah tugas yang diinginkan.

Untuk menyertakan aktivitas yang tidak diskalakan (bukan aktivitas yang diskalakan) dalam respons, tambahkan `--include-not-scaled-activities` opsi ke perintah. describe-scaling-activities

Linux, macOS, atau Unix

```
aws application-autoscaling describe-scaling-activities --include-not-scaled-activities
 \
  --service-namespace ecs --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service
```

Windows

```
aws application-autoscaling describe-scaling-activities --include-not-scaled-activities
 --service-namespace ecs --scalable-dimension ecs:service:DesiredCount --resource-
id service/my-cluster/my-service
```

> **Note**
>
> Jika perintah ini menimbulkan kesalahan, pastikan Anda telah memperbarui AWS CLI secara lokal ke versi terbaru.

Untuk mengonfirmasi bahwa respons mencakup aktivitas yang tidak diskalakan, `NotScaledReasons` elemen ditampilkan dalam output untuk beberapa, jika tidak semua, aktivitas penskalaan yang gagal.

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "ecs:service:DesiredCount",
            "Description": "Attempting to scale due to alarm triggered",
            "ResourceId": "service/my-cluster/my-service",
            "ActivityId": "4d759079-a31f-4d0c-8468-504c56e2eecf",
            "StartTime": 1664928867.915,
            "ServiceNamespace": "ecs",
            "Cause": "monitor alarm web-app-cpu-gt-75 in state ALARM triggered policy
 web-app-cpu-gt-75",
```

```
            "StatusCode": "Failed",
            "NotScaledReasons": [
                {
                    "Code": "AlreadyAtMaxCapacity",
                    "MaxCapacity": 4
                }
            ]
        }
    ]
}
```

Untuk deskripsi bidang dalam respons, lihat ScalingActivitydi Referensi API Application Auto Scaling.

Jika aktivitas yang tidak diskalakan dikembalikan, tergantung pada kode alasan yang tercantum dalam`Code`, atribut seperti `CurrentCapacity``MaxCapacity`,, dan `MinCapacity` mungkin ada dalam respons.

Untuk mencegah entri duplikat dalam jumlah besar, hanya aktivitas pertama yang tidak diskalakan yang akan dicatat dalam riwayat aktivitas penskalaan. Setiap aktivitas yang tidak diskalakan berikutnya tidak akan menghasilkan entri baru kecuali alasan untuk tidak mengubah penskalaan.

Kode alasan

Berikut ini adalah kode alasan untuk aktivitas yang tidak diskalakan.

Kode alasan	Definisi			
AutoScalingAnticipatedFlapping	Algoritma penskalaan otomatis memutuskan untuk tidak mengambil tindakan penskalaan karena akan menyebabkan flapping. Flapping adalah lingkaran penskalaan dan			

Kode alasan	Definisi			
	penskalaan yang tak terbatas. Artinya, jika tindakan penskalaan diambil, nilai metrik akan berubah untuk memulai tindakan penskalaan lain dalam arah sebaliknya.			
TargetServicePutResourceAsUnscalable	Layanan target untuk sementara menempatkan sumber daya dalam keadaan tidak dapat diskalakan. Application Auto Scaling akan mencoba untuk menskalakan lagi ketika kondisi penskalaan otomatis yang ditentukan dalam kebijakan penskalaan terpenuhi.			

Kode alasan	Definisi			
AlreadyAtMaxCapacity	Penskalaan diblokir oleh kapasitas maksimum yang Anda tentukan. Jika Anda ingin Application Auto Scaling ditingkatkan, Anda perlu meningkatkan kapasitas maksimum.			
AlreadyAtMinCapacity	Penskalaan diblokir oleh kapasitas minimum yang Anda tentukan. Jika Anda ingin Application Auto Scaling masuk, Anda perlu mengurangi kapasitas minimum.			
AlreadyAtDesiredCapacity	Algoritma penskalaan otomatis menghitung kapasitas yang direvisi menjadi sama dengan kapasitas saat ini.			

Pemantauan Application Auto Scaling

Pemantauan adalah bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja Application Auto Scaling dan solusi Anda yang lain AWS . Anda harus mengumpulkan data pemantauan dari semua bagian AWS solusi Anda sehingga Anda dapat lebih mudah men-debug kegagalan multi-titik jika terjadi. AWS menyediakan alat pemantauan untuk menonton Application Auto Scaling, melaporkan ketika ada sesuatu yang salah, dan mengambil tindakan otomatis bila perlu.

Anda dapat menggunakan fitur-fitur berikut untuk membantu Anda mengelola AWS sumber daya Anda:

AWS CloudTrail

Dengan AWS CloudTrail, Anda dapat melacak panggilan yang dilakukan ke Application Auto Scaling API oleh atau atas nama Anda. Akun AWS CloudTrail menyimpan informasi dalam file log di bucket Amazon S3 yang Anda tentukan. Anda dapat mengidentifikasi pengguna dan akun mana yang disebut Application Auto Scaling, alamat IP sumber dari mana panggilan dilakukan, dan kapan panggilan terjadi. Untuk informasi selengkapnya, lihat Log Application Auto Scaling API call menggunakan AWS CloudTrail.

> **Note**
>
> Untuk informasi tentang AWS layanan lain yang dapat membantu Anda mencatat dan mengumpulkan data tentang beban kerja Anda, lihat panduan Pencatatan dan pemantauan untuk pemilik aplikasi di Panduan AWS Preskriptif.

Amazon CloudWatch

Amazon CloudWatch membantu Anda menganalisis log dan, secara real time, memantau metrik AWS sumber daya dan aplikasi yang dihosting. Anda dapat mengumpulkan dan melacak metrik, membuat dasbor yang disesuaikan, dan mengatur alarm yang memberi tahu Anda atau mengambil tindakan saat metrik tertentu mencapai ambang batas yang ditentukan. Misalnya, Anda dapat CloudWatch melacak pemanfaatan sumber daya dan memberi tahu Anda saat pemanfaatan sangat tinggi atau ketika alarm metrik telah masuk ke status. `INSUFFICIENT_DATA` Untuk informasi selengkapnya, lihat Memantau penggunaan sumber daya yang dapat diskalakan menggunakan CloudWatch.

CloudWatch juga melacak metrik penggunaan AWS API untuk Application Auto Scaling. Anda dapat menggunakan metrik ini untuk mengonfigurasi alarm yang memberi tahu Anda saat volume panggilan API melanggar ambang batas yang Anda tentukan. Untuk informasi selengkapnya, lihat metrik AWS penggunaan di Panduan CloudWatch Pengguna Amazon.

Amazon EventBridge

Amazon EventBridge adalah layanan bus acara tanpa server yang memudahkan untuk menghubungkan aplikasi Anda dengan data dari berbagai sumber. EventBridge mengirimkan aliran data real-time dari aplikasi Anda sendiri, aplikasi Software-as-a-Service (SaaS), AWS dan layanan dan rute data tersebut ke target seperti Lambda. Ini memungkinkan Anda memantau peristiwa yang terjadi di layanan, dan membangun arsitektur berbasis peristiwa. Untuk informasi selengkapnya, lihat Memantau peristiwa Application Auto Scaling menggunakan Amazon EventBridge.

Dasbor AWS Health

Dasbor Health (PHD) menampilkan informasi, dan juga menyediakan pemberitahuan yang dipanggil oleh perubahan kesehatan sumber daya. AWS Informasi ini disajikan dalam dua cara: di dasbor yang menampilkan peristiwa terbaru dan mendatang yang diatur berdasarkan kategori, dan dalam catatan peristiwa lengkap yang menampilkan semua peristiwa dari 90 hari terakhir. Untuk informasi selengkapnya, lihat Memulai dengan Anda Dasbor Health.

Memantau penggunaan sumber daya yang dapat diskalakan menggunakan CloudWatch

Dengan Amazon CloudWatch, Anda mendapatkan visibilitas yang hampir terus menerus ke dalam aplikasi Anda di seluruh sumber daya yang dapat diskalakan. CloudWatch adalah layanan pemantauan untuk AWS sumber daya. Anda dapat menggunakannya CloudWatch untuk mengumpulkan dan melacak metrik, menyetel alarm, dan bereaksi secara otomatis terhadap perubahan sumber daya Anda AWS . Anda juga dapat membuat dasbor untuk memantau metrik atau set metrik tertentu yang Anda butuhkan.

Saat Anda berinteraksi dengan layanan yang terintegrasi dengan Application Auto Scaling, mereka akan mengirimkan metrik yang ditunjukkan pada tabel berikut. CloudWatch Dalam CloudWatch, metrik dikelompokkan terlebih dahulu oleh namespace layanan, dan kemudian oleh berbagai kombinasi dimensi dalam setiap namespace. Metrik ini dapat membantu Anda memantau penggunaan sumber daya dan kapasitas rencana untuk aplikasi Anda. Jika beban kerja aplikasi

Anda tidak konstan, ini menunjukkan bahwa Anda harus mempertimbangkan untuk menggunakan penskalaan otomatis. Untuk deskripsi mendetail tentang metrik ini, lihat dokumentasi untuk metrik yang diminati.

Daftar Isi

CloudWatch metrik untuk memantau penggunaan sumber daya

Tabel berikut mencantumkan CloudWatch metrik yang tersedia untuk mendukung pemantauan penggunaan sumber daya. Daftar ini tidak lengkap tetapi akan memberi Anda titik awal yang baik. Jika Anda tidak melihat metrik ini di CloudWatch konsol, pastikan Anda telah menyelesaikan pengaturan sumber daya. Untuk informasi selengkapnya, lihat Panduan CloudWatch Pengguna Amazon.

Sumber daya yang dapat diskalakan	Namespace	CloudWatch metrik	Tautan ke dokumentasi
WorkSpaces Aplikasi			
Armada	AWS/AppStream	Nama: AvailableCapacity Dimensi: Armada	WorkSpaces Metrik aplikasi
Armada	AWS/AppStream	Nama: CapacityUtilization Dimensi: Armada	WorkSpaces Metrik aplikasi
Aurora			

Sumber daya yang dapat diskalakan	Namespace	CloudWatch metrik	Tautan ke dokumentasi
Replika	AWS/RDS	Nama: CPUUtilization Dimensi: DBClusterPengidentifikasi, Peran (PEMBACA	Metrik tingkat klaster Aurora
Replika	AWS/RDS	Nama: DatabaseConnections Dimensi: DBClusterPengidentifikasi, Peran (PEMBACA	Metrik tingkat klaster Aurora
Amazon Comprehend			
Titik akhir klasifikasi dokumen	AWS/Comprehend	Nama: InferenceUtilization Dimensi: EndpointArn	Amazon Comprehend metrik titik akhir

Sumber daya yang dapat diskalakan	Namespace	CloudWatch metrik	Tautan ke dokumentasi
Titik akhir pengenal entitas	AWS/Comprehend	Nama: InferenceUtilization Dimensi: EndpointArn	Amazon Comprehend metrik titik akhir
DynamoDB			
Tabel dan indeks sekunder global	AWS/DynamoDB	Nama: ProvisionedReadCapacityUnits Dimensi: TableName, GlobalSecondaryIndexName	Metrik DynamoDB
Tabel dan indeks sekunder global	AWS/DynamoDB	Nama: ProvisionedWriteCapacityUnits Dimensi: TableName, GlobalSecondaryIndexName	Metrik DynamoDB

Sumber daya yang dapat diskalakan	Namespace	CloudWatch metrik	Tautan ke dokumentasi
Tabel dan indeks sekunder global	AWS/DynamoDB	Nama: ConsumedReadCapacityUnits Dimensi: TableName, GlobalSecondaryIndexName	Metrik DynamoDB
Tabel dan indeks sekunder global	AWS/DynamoDB	Nama: ConsumedWriteCapacityUnits Dimensi: TableName, GlobalSecondaryIndexName	Metrik DynamoDB
Amazon ECS			
Layanan	AWS/ECS	Nama: CPUUtilization Dimensi: ClusterName, ServiceName	Metrik Amazon ECS

Sumber daya yang dapat diskalakan	Namespace	CloudWatch metrik	Tautan ke dokumentasi
Layanan	AWS/ECS	Nama: MemoryUtilization Dimensi: ClusterName, ServiceName	Metrik Amazon ECS
Layanan	AWS/ApplicationELB	Nama: RequestCountPerTarget Dimensi: TargetGroup	Metrik Application Load Balancer
ElastiCache			
Cluster (kelompok replikasi)	AWS/ElastiCache	Nama: DatabaseMemoryUsageCountedForEvictPercentage Dimensi: ReplicationGroupId	ElastiCache Metrik Valkey dan Redis OSS

Sumber daya yang dapat diskalakan	Namespace	CloudWatch metrik	Tautan ke dokumentasi
Cluster (kelompok replikasi)	AWS/ElastiCache	Nama: DatabaseCapacityUsageCountedForEvictPercentage Dimensi: ReplicationGroupId	ElastiCache Metrik Valkey dan Redis OSS
Cluster (kelompok replikasi)	AWS/ElastiCache	Nama: MesinCPUUtilization Dimensi: ReplicationGroupId, Peran (Utama)	ElastiCache Metrik Valkey dan Redis OSS
Cluster (kelompok replikasi)	AWS/ElastiCache	Nama: MesinCPUUtilization Dimensi: ReplicationGroupId, Peran (Replika)	ElastiCache Metrik Valkey dan Redis OSS

Sumber daya yang dapat diskalakan	Namespace	CloudWatch metrik	Tautan ke dokumentasi
Cluster (cache)	AWS/ElastiCache	Nama: EngineCPUUtilization Dimensi: CacheClusterId, Node	ElastiCache Metrik memcache
Cluster (cache)	AWS/ElastiCache	Nama: DatabaseCapacityMemoryUsagePercentage Dimensi: CacheClusterId	ElastiCache Metrik memcache
Amazon EMR			
klaster	AWS/ElasticMapReduce	Nama: YARNMemoryAvailablePercentage Dimensi: ClusterId	Metrik Amazon EMR
Keyspaces Amazon			

Sumber daya yang dapat diskalakan	Namespace	CloudWatch metrik	Tautan ke dokumentasi
Tabel	AWS/Cassandra	Nama: ProvisionedReadCapacityUnits Dimensi: Keyspace, TableName	Metrik Amazon Keyspaces
Tabel	AWS/Cassandra	Nama: ProvisionedWriteCapacityUnits Dimensi: Keyspace, TableName	Metrik Amazon Keyspaces
Tabel	AWS/Cassandra	Nama: ConsumedReadCapacityUnits Dimensi: Keyspace, TableName	Metrik Amazon Keyspaces

Sumber daya yang dapat diskalakan	Namespace	CloudWatch metrik	Tautan ke dokumentasi
Tabel	AWS/Cassandra	Nama: ConsumedWriteCapacityUnits Dimensi: Keyspace, TableName	Metrik Amazon Keyspaces
Lambda			
Konkurensi yang disediakan	AWS/Lambda	Nama: ProvisionedConcurrencyUtilization Dimensi: FunctionName, Sumber	Metrik fungsi Lambda
Amazon MSK			
Penyimpanan broker	AWS/Kafka	Nama: KafkaDataLogsDiskUsed Dimensi: Nama Cluster	Metrik MSK Amazon

Sumber daya yang dapat diskalakan	Namespace	CloudWatch metrik	Tautan ke dokumentasi
Penyimpanan broker	AWS/Kafka	Nama: KafkaDataLogsDiskUsed Dimensi: Nama Cluster, ID Broker	Metrik MSK Amazon
Neptunus			
klaster	AWS/Neptunus	Nama: CPUUtilization Dimensi: DBClusterPengidentifikasi, Peran (PEMBACA	Metrik Neptunus
SageMaker AI			
Varian titik akhir	AWS/SageMaker	Nama: InvocationsPerInstance Dimensi: EndpointName, VariantName	Metrik pemanggilan

Sumber daya yang dapat diskalakan	Namespace	CloudWatch metrik	Tautan ke dokumentasi
Komponen inferensi	AWS/SageMaker	Nama: InvocationsPerCopy Dimensi: InferenceComponentName	Metrik pemanggilan
Konkurensi yang disediakan untuk titik akhir tanpa server	AWS/SageMaker	Nama: ServerlessProvisionedConcurrencyUtilization Dimensi: EndpointName, VariantName	Metrik titik akhir tanpa server
Armada Spot (Amazon EC2)			
Armada Spot	AWS/TempatEC2	Nama: CPUUtilization Dimensi: FleetRequestId	Metrik Armada Spot

Sumber daya yang dapat diskalakan	Namespace	CloudWatch metrik	Tautan ke dokumentasi
Armada Spot	AWS/EC2	Nama: NetworkIn Dimensi: FleetRequestId	Metrik Armada Spot
Armada Spot	AWS/EC2	Nama: NetworkOut Dimensi: FleetRequestId	Metrik Armada Spot
Armada Spot	AWS/ApplicationELB	Nama: RequestCountPerTarget Dimensi: TargetGroup	Metrik Application Load Balancer

Metrik yang telah ditentukan untuk kebijakan penskalaan pelacakan target

Tabel berikut mencantumkan tipe metrik yang telah ditentukan dari Referensi API Application Auto Scaling dengan nama metrik yang CloudWatch sesuai. Setiap metrik yang telah ditentukan mewakili agregasi nilai metrik yang mendasarinya CloudWatch . Hasilnya adalah penggunaan sumber daya rata-rata selama periode satu menit, berdasarkan persentase kecuali dinyatakan lain. Metrik yang telah ditentukan hanya digunakan dalam konteks pengaturan kebijakan penskalaan pelacakan target.

Anda dapat menemukan informasi selengkapnya tentang metrik ini di dokumentasi layanan yang tersedia dari tabel diCloudWatch metrik untuk memantau penggunaan sumber daya.

Jenis metrik yang telah ditentukan	CloudWatch nama metrik
WorkSpaces Aplikasi	
`AppStreamAverageCapacityUtilization`	CapacityUtilization
Aurora	
`RDSReaderAverageCPUUtilization`	CPUUtilization
`RDSReaderAverageDatabaseConnections`	DatabaseConnections[1]
Amazon Comprehend	
`ComprehendInferenceUtilization`	InferenceUtilization
DynamoDB	
`DynamoDBReadCapacityUtilization`	ProvisionedReadCapacityUnits, ConsumedReadCapacityUnits [2]
`DynamoDBWriteCapacityUtilization`	ProvisionedWriteCapacityUnits, ConsumedWriteCapacityUnits [2]
Amazon ECS	
`ECSServiceAverageCPUUtilization`	CPUUtilization
`ECSServiceAverageMemoryUtilization`	MemoryUtilization
`ALBRequestCountPerTarget`	RequestCountPerTarget[1]
ElastiCache	
`ElastiCacheDatabaseMemoryUsageCountedForEvictPercentage`	DatabaseMemoryUsageCountedForEvictPercentage

Jenis metrik yang telah ditentukan	CloudWatch nama metrik
`ElastiCacheDatabaseCapacityUsageCountedForEvictPercentage`	DatabaseCapacityUsageCountedForEvictPercentage
`ElastiCachePrimaryEngineCPUUtilization`	Mesin CPUUtilization
`ElastiCacheReplicaEngineCPUUtilization`	Mesin CPUUtilization
`ElastiCacheEngineCPUUtilization`	Mesin CPUUtilization
`ElastiCacheDatabaseMemoryUsagePercentage`	DatabaseMemoryUsagePercentage
Keyspaces Amazon	
`CassandraReadCapacityUtilization`	ProvisionedReadCapacityUnits, ConsumedReadCapacityUnits [2]
`CassandraWriteCapacityUtilization`	ProvisionedWriteCapacityUnits, ConsumedWriteCapacityUnits [2]
Lambda	
`LambdaProvisionedConcurrencyUtilization`	ProvisionedConcurrencyUtilization
Amazon MSK	
`KafkaBrokerStorageUtilization`	KafkaDataLogsDiskUsed
Neptunus	
`NeptuneReaderAverageCPUUtilization`	CPUUtilization
SageMaker AI	

Jenis metrik yang telah ditentukan	CloudWatch nama metrik
`SageMakerVariantInvocationsPerInstance`	InvocationsPerInstance[1]
`SageMakerInferenceComponentInvocationsPerCopy`	InvocationsPerCopy[1]
`SageMakerVariantProvisionedConcurrencyUtilization`	ServerlessProvisionedConcurrencyUtilization
`SageMakerInferenceComponentConcurrentRequestsPerCopyHighResolution`	ConcurrentRequestsPerCopy
`SageMakerVariantConcurrentRequestsPerModelHighResolution`	ConcurrentRequestsPerModel
Armada Spot	
`EC2SpotFleetRequestAverageCPUUtilization`	CPUUtilization[3]
`EC2SpotFleetRequestAverageNetworkIn`[3]	NetworkIn[1] [3]
`EC2SpotFleetRequestAverageNetworkOut`[3]	NetworkOut[1] [3]
`ALBRequestCountPerTarget`	RequestCountPerTarget[1]

[1] Metrik didasarkan pada hitungan, bukan persentase.

[2] Untuk DynamoDB dan Amazon Keyspaces, metrik yang telah ditentukan adalah agregasi dari dua metrik untuk mendukung penskalaan berdasarkan konsumsi CloudWatch throughput yang disediakan.

[3] Untuk kinerja penskalaan terbaik, pemantauan terperinci Amazon EC2 harus digunakan.

Metrik dan dimensi penskalaan prediktif

`AWS/ApplicationAutoScaling`Namespace menyertakan metrik berikut untuk kebijakan penskalaan prediktif. Metrik ini tersedia dengan resolusi satu jam dan dapat membantu Anda mengevaluasi akurasi perkiraan dengan membandingkan nilai yang diperkirakan dengan nilai aktual.

Metrik	Deskripsi	Dimensi
`PredictiveScalingLoadForecast`	Jumlah beban yang diantisipasi akan dihasilkan oleh aplikasi Anda. Statistik `Average`, `Minimum`, dan `Maximum` statistik berguna, tetapi `Sum` statistiknya tidak. Kriteria pelaporan: Dilaporkan setelah perkiraan awal dibuat.	`ResourceId`, `ServiceNamespace`, `PolicyName`, `ScalableDimension`, `PairIndex`
`PredictiveScalingCapacityForecast`	Jumlah kapasitas yang diantisipasi yang dibutuhkan untuk memenuhi permintaan aplikasi. Ini didasarkan pada perkiraan beban dan tingkat pemanfaatan target di mana Anda ingin mempertahankan sumber daya Application Auto Scaling Anda. Statistik `Average`, `Minimum`, dan `Maximum` statistik berguna, tetapi `Sum` statistiknya tidak. Kriteria pelaporan: Dilaporkan setelah perkiraan awal dibuat.	`ResourceId`, `ServiceNamespace`, `PolicyName`, `ScalableDimension`
`PredictiveScalingMetricPairCorrelation`	Korelasi antara metrik penskalaan dan rata-rata per instance dari metrik beban. Penskalaan prediktif mengasumsikan korelasi tinggi. Oleh karena itu, jika Anda mengamati nilai rendah untuk metrik ini, lebih baik tidak menggunakan pasangan metrik. Statistik `Average`, `Minimum`, dan `Maximum` statistik berguna, tetapi `Sum` statistiknya tidak.	`ResourceId`, `ServiceNamespace`, `PolicyName`, `ScalableDimension`, `PairIndex`

Metrik	Deskripsi	Dimensi
	Kriteria pelaporan: Dilaporkan setelah perkiraan awal dibuat.	

Log Application Auto Scaling API call menggunakan AWS CloudTrail

Application Auto Scaling terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau. Layanan AWS CloudTrail menangkap panggilan API untuk Application Auto Scaling sebagai peristiwa. Panggilan yang diambil termasuk panggilan dari panggilan Konsol Manajemen AWS dan kode ke operasi Application Auto Scaling API. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat untuk Application Auto Scaling, alamat IP dari mana permintaan dibuat, kapan dibuat, dan detail tambahan.

Setiap entri peristiwa atau log berisi informasi tentang entitas yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan berikut hal ini:

- Baik permintaan tersebut dibuat dengan kredensial pengguna root atau pengguna.
- Apakah permintaan dibuat atas nama pengguna IAM Identity Center.
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna gabungan.
- Apakah permintaan tersebut dibuat oleh Layanan AWS lain.

CloudTrail aktif di Anda Akun AWS ketika Anda membuat akun dan Anda secara otomatis memiliki akses ke riwayat CloudTrail Acara. Riwayat CloudTrail Acara menyediakan catatan yang dapat dilihat, dapat dicari, dapat diunduh, dan tidak dapat diubah dari 90 hari terakhir dari peristiwa manajemen yang direkam dalam file. AWS Region Untuk informasi selengkapnya, lihat Bekerja dengan riwayat CloudTrail Acara di Panduan AWS CloudTrail Pengguna. Tidak ada CloudTrail biaya untuk melihat riwayat Acara.

Untuk catatan peristiwa yang sedang berlangsung dalam 90 hari Akun AWS terakhir Anda, buat jejak.

CloudTrail jalan setapak

Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Semua jalur yang dibuat menggunakan Konsol Manajemen AWS Multi-region. Anda dapat membuat jalur Single-region atau Multi-region dengan menggunakan. AWS CLI Membuat jejak Multi-wilayah disarankan karena Anda menangkap aktivitas Wilayah AWS di semua akun Anda. Jika Anda membuat jejak wilayah Tunggal, Anda hanya dapat melihat peristiwa yang dicatat di jejak. AWS Region Untuk informasi selengkapnya tentang jejak, lihat Membuat jejak untuk Anda Akun AWS dan Membuat jejak untuk organisasi di Panduan AWS CloudTrail Pengguna.

Anda dapat mengirimkan satu salinan acara manajemen yang sedang berlangsung ke bucket Amazon S3 Anda tanpa biaya CloudTrail dengan membuat jejak, namun, ada biaya penyimpanan Amazon S3. Untuk informasi selengkapnya tentang CloudTrail harga, lihat AWS CloudTrail Harga. Untuk informasi tentang harga Amazon S3, lihat Harga Amazon S3.

Acara manajemen Application Auto Scaling di CloudTrail

Acara manajemen memberikan informasi tentang operasi manajemen yang dilakukan pada sumber daya di Anda Akun AWS. Ini juga dikenal sebagai operasi bidang kontrol. Secara default, CloudTrail mencatat peristiwa manajemen.

Application Auto Scaling mencatat semua operasi bidang kontrol Application Auto Scaling sebagai peristiwa manajemen. Untuk daftar operasi bidang kontrol Application Auto Scaling yang digunakan untuk log CloudTrail Application Auto Scaling, lihat Referensi API Application Auto Scaling.

Contoh acara Application Auto Scaling

Peristiwa mewakili permintaan tunggal dari sumber manapun dan mencakup informasi tentang operasi API yang diminta, tanggal dan waktu operasi, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, sehingga peristiwa tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan CloudTrail peristiwa yang menunjukkan `DescribeScalableTargets` operasi.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
```

```
        "type": "Root",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:root",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-08-21T17:05:42Z"
            }
        }
    },
    "eventTime": "2018-08-16T23:20:32Z",
    "eventSource": "autoscaling.amazonaws.com",
    "eventName": "DescribeScalableTargets",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "72.21.196.68",
    "userAgent": "EC2 Spot Console",
    "requestParameters": {
        "serviceNamespace": "ec2",
        "scalableDimension": "ec2:spot-fleet-request:TargetCapacity",
        "resourceIds": [
            "spot-fleet-request/sfr-05ceaf79-3ba2-405d-e87b-612857f1357a"
        ]
    },
    "responseElements": null,
    "additionalEventData": {
        "service": "application-autoscaling"
    },
    "requestID": "0737e2ea-fb2d-11e3-bfd8-99133058e7bb",
    "eventID": "3fcfb182-98f8-4744-bd45-b38835ab61cb",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```


Untuk informasi tentang konten CloudTrail rekaman, lihat konten CloudTrail rekaman di Panduan AWS CloudTrail Pengguna.

Application Auto Scaling memanggil RemoveAction CloudWatch

AWS CloudTrail Log Anda mungkin menunjukkan bahwa Application Auto Scaling memanggil CloudWatch `RemoveAction` API saat Application Auto Scaling CloudWatch menginstruksikan untuk menghapus tindakan penskalaan otomatis dari alarm. Ini bisa terjadi jika Anda membatalkan

pendaftaran target yang dapat diskalakan, menghapus kebijakan penskalaan, atau jika alarm memanggil kebijakan penskalaan yang tidak ada.

Memantau peristiwa Application Auto Scaling menggunakan Amazon EventBridge

Amazon EventBridge, sebelumnya disebut CloudWatch Events, membantu Anda memantau peristiwa yang khusus untuk Application Auto Scaling dan memulai tindakan target yang menggunakan lainnya. Layanan AWS Acara dari Layanan AWS dikirim ke EventBridge dalam waktu dekat.

Dengan menggunakan EventBridge, Anda dapat membuat aturan yang cocok dengan peristiwa yang masuk dan merutekkannya ke target untuk diproses.

Untuk informasi selengkapnya, lihat Memulai Amazon EventBridge di Panduan EventBridge Pengguna Amazon.

Peristiwa Application Auto Scaling

Contoh berikut menunjukkan peristiwa untuk Application Auto Scaling. Acara diproduksi atas dasar upaya terbaik.

Hanya peristiwa yang khusus untuk diskalakan ke max dan panggilan API via yang saat ini CloudTrail tersedia untuk Application Auto Scaling.

Tipe peristiwa

- Peristiwa untuk perubahan status: diskalakan ke maks
- Acara untuk panggilan API melalui CloudTrail

Peristiwa untuk perubahan status: diskalakan ke maks

Contoh peristiwa berikut menunjukkan bahwa Application Auto Scaling meningkatkan (memperkecil) kapasitas target yang dapat diskalakan hingga batas ukuran maksimumnya. Jika permintaan meningkat lagi, Application Auto Scaling akan dicegah dari penskalaan target ke ukuran yang lebih besar karena sudah diskalakan ke ukuran maksimumnya.

Dalam `detail` objek, nilai untuk, `resourceIdserviceNamespace`, dan `scalableDimension` atribut mengidentifikasi target yang dapat diskalakan. Nilai untuk `oldDesiredCapacity` atribut

`newDesiredCapacity` dan mengacu pada kapasitas baru setelah peristiwa scale-out dan kapasitas asli sebelum acara scale-out. `maxCapacity`Ini adalah batas ukuran maksimum dari target yang dapat diskalakan.

```
{
  "version": "0",
  "id": "11112222-3333-4444-5555-666677778888",
  "detail-type": "Application Auto Scaling Scaling Activity State Change",
  "source": "aws.application-autoscaling",
  "account": "123456789012",
  "time": "2019-06-12T10:23:40Z",
  "region": "us-west-2",
  "resources": [],
  "detail": {
    "startTime": "2022-06-12T10:20:43Z",
    "endTime": "2022-06-12T10:23:40Z",
    "newDesiredCapacity": 8,
    "oldDesiredCapacity": 5,
    "minCapacity": 2,
    "maxCapacity": 8,
    "resourceId": "table/my-table",
    "scalableDimension": "dynamodb:table:WriteCapacityUnits",
    "serviceNamespace": "dynamodb",
    "statusCode": "Successful",
    "scaledToMax": true,
    "direction": "scale-out"
}
```

Untuk membuat aturan yang menangkap semua peristiwa perubahan `scaledToMax` status untuk semua target yang dapat diskalakan, gunakan contoh pola peristiwa berikut.

```
{
  "source": [
    "aws.application-autoscaling"
  ],
  "detail-type": [
    "Application Auto Scaling Scaling Activity State Change"
  ],
  "detail": {
    "scaledToMax": [
      true
    ]
```

```
  }
}
```

Acara untuk panggilan API melalui CloudTrail

Trail adalah konfigurasi yang AWS CloudTrail digunakan untuk mengirimkan peristiwa sebagai file log ke bucket Amazon S3. CloudTrail file log berisi entri log. Suatu peristiwa mewakili entri log, dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, dan parameter permintaan. Untuk mempelajari cara memulai CloudTrail, lihat Membuat jejak di Panduan AWS CloudTrail Pengguna.

Acara yang disampaikan melalui CloudTrail memiliki `AWS API Call via CloudTrail` nilai untuk`detail-type`.

Contoh peristiwa berikut mewakili entri file CloudTrail log yang menunjukkan bahwa pengguna konsol disebut tindakan Application Auto Scaling. RegisterScalableTarget

```
{
  "version": "0",
  "id": "99998888-7777-6666-5555-444433332222",
  "detail-type": "AWS API Call via CloudTrail",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2022-07-13T16:50:15Z",
  "region": "us-west-2",
  "resources": [],
  "detail": {
    "eventVersion": "1.08",
    "userIdentity": {
      "type": "IAMUser",
      "principalId": "123456789012",
      "arn": "arn:aws:iam::123456789012:user/Bob",
      "accountId": "123456789012",
      "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
      "sessionContext": {
        "sessionIssuer": {
          "type": "Role",
          "principalId": "123456789012",
          "arn": "arn:aws:iam::123456789012:role/Admin",
          "accountId": "123456789012",
          "userName": "Admin"
        },
```

```
        "webIdFederationData": {},
        "attributes": {
          "creationDate": "2022-07-13T15:17:08Z",
          "mfaAuthenticated": "false"
        }
      }
    },
    "eventTime": "2022-07-13T16:50:15Z",
    "eventSource": "autoscaling.amazonaws.com",
    "eventName": "RegisterScalableTarget",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "EC2 Spot Console",
    "requestParameters": {
      "resourceId": "spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
      "serviceNamespace": "ec2",
      "scalableDimension": "ec2:spot-fleet-request:TargetCapacity",
      "minCapacity": 2,
      "maxCapacity": 10
    },
    "responseElements": null,
    "additionalEventData": {
      "service": "application-autoscaling"
    },
    "requestID": "e9caf887-8d88-11e5-a331-3332aa445952",
    "eventID": "49d14f36-6450-44a5-a501-b0fdcdfaeb98",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "sessionCredentialFromConsole": "true"
  }
}
```


Untuk membuat aturan berdasarkan semua DeleteScalingPolicydan panggilan DeregisterScalableTargetAPI untuk semua target yang dapat diskalakan, gunakan contoh pola peristiwa berikut:


```
{
  "source": [
    "aws.autoscaling"
  ],
```

```
  "detail-type": [
    "AWS API Call via CloudTrail"
  ],
  "detail": {
    "eventSource": [
      "autoscaling.amazonaws.com"
    ],
    "eventName": [
       "DeleteScalingPolicy",
       "DeregisterScalableTarget"
    ],
    "additionalEventData": {
      "service": [
        "application-autoscaling"
      ]
    }
  }
}
```

Untuk informasi selengkapnya tentang penggunaan CloudTrail, lihatLog Application Auto Scaling API call menggunakan AWS CloudTrail.

Menggunakan layanan ini dengan AWS SDK

AWS kit pengembangan perangkat lunak (SDKs) tersedia untuk banyak bahasa pemrograman populer. Setiap SDK menyediakan API, contoh kode, dan dokumentasi yang memudahkan developer untuk membangun aplikasi dalam bahasa pilihan mereka.

Dokumentasi SDK	Contoh kode
AWS SDK for C++	AWS SDK for C++ contoh kode
AWS CLI	AWS CLI contoh kode
AWS SDK for Go	AWS SDK for Go contoh kode
AWS SDK for Java	AWS SDK for Java contoh kode
AWS SDK for JavaScript	AWS SDK for JavaScript contoh kode
AWS SDK for Kotlin	AWS SDK for Kotlin contoh kode
AWS SDK for .NET	AWS SDK for .NET contoh kode
AWS SDK for PHP	AWS SDK for PHP contoh kode
AWS Tools for PowerShell	AWS Tools for PowerShell contoh kode
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) contoh kode
AWS SDK for Ruby	AWS SDK for Ruby contoh kode
AWS SDK for Rust	AWS SDK for Rust contoh kode
AWS SDK for SAP ABAP	AWS SDK for SAP ABAP contoh kode
AWS SDK for Swift	AWS SDK for Swift contoh kode

Ketersediaan contoh

Tidak dapat menemukan apa yang Anda butuhkan? Minta contoh kode menggunakan tautan Berikan umpan balik di bagian bawah halaman ini.

Contoh kode untuk Application Auto Scaling menggunakan AWS SDKs

Contoh kode berikut menunjukkan cara menggunakan Application Auto Scaling dengan AWS software development kit (SDK).

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Contoh kode

Contoh dasar untuk Application Auto Scaling menggunakan AWS SDKs

Contoh kode berikut menunjukkan cara menggunakan dasar-dasar Application Auto Scaling dengan. AWS SDKs

Contoh

Tindakan untuk Application Auto Scaling menggunakan AWS SDKs

Contoh kode berikut menunjukkan cara melakukan tindakan Application Auto Scaling individual dengan. AWS SDKs Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan instruksi untuk mengatur dan menjalankan kode.

Contoh berikut hanya mencakup tindakan yang paling umum digunakan. Untuk daftar lengkapnya, lihat Referensi API Application Auto Scaling.

Contoh

- Gunakan DeleteScalingPolicy dengan AWS SDK atau CLI
- Gunakan DeleteScheduledAction dengan CLI
- Gunakan DeregisterScalableTarget dengan CLI
- Gunakan DescribeScalableTargets dengan CLI
- Gunakan DescribeScalingActivities dengan CLI
- Gunakan DescribeScalingPolicies dengan AWS SDK atau CLI
- Gunakan DescribeScheduledActions dengan CLI
- Gunakan PutScalingPolicy dengan CLI
- Gunakan PutScheduledAction dengan CLI
- Gunakan RegisterScalableTarget dengan AWS SDK atau CLI

Gunakan **`DeleteScalingPolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteScalingPolicy`.

CLI

AWS CLI

Untuk menghapus kebijakan penskalaan

Contoh ini menghapus kebijakan penskalaan untuk aplikasi web layanan Amazon ECS yang berjalan di kluster default.

Perintah:

```
aws application-autoscaling delete-scaling-policy --policy-name web-app-cpu-lt-25
 --scalable-dimension ecs:service:DesiredCount --resource-id service/default/web-
app --service-namespace ecs
```

- Untuk detail API, lihat DeleteScalingPolicydi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.applicationautoscaling.ApplicationAutoScalingClient;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ApplicationAutoScalingExcep
import
 software.amazon.awssdk.services.applicationautoscaling.model.DeleteScalingPolicyRequest;
import
 software.amazon.awssdk.services.applicationautoscaling.model.DeregisterScalableTargetReq
```

```
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalableTargetsRequ
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalableTargetsResp
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalingPoliciesRequ
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalingPoliciesResp
import
 software.amazon.awssdk.services.applicationautoscaling.model.ScalableDimension;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ServiceNamespace;

/**
 * Before running this Java V2 code example, set up your development environment,
 including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class DisableDynamoDBAutoscaling {
    public static void main(String[] args) {
        final String usage = """

            Usage:
               <tableId> <policyName>\s

            Where:
               tableId - The table Id value (for example, table/Music).\s
               policyName - The name of the policy (for example, $Music5-scaling-
policy).

            """;
        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        ApplicationAutoScalingClient appAutoScalingClient =
 ApplicationAutoScalingClient.builder()
            .region(Region.US_EAST_1)
```

```
            .build();

        ServiceNamespace ns = ServiceNamespace.DYNAMODB;
        ScalableDimension tableWCUs =
ScalableDimension.DYNAMODB_TABLE_WRITE_CAPACITY_UNITS;
        String tableId = args[0];
        String policyName = args[1];

        deletePolicy(appAutoScalingClient, policyName, tableWCUs, ns, tableId);
        verifyScalingPolicies(appAutoScalingClient, tableId, ns, tableWCUs);
        deregisterScalableTarget(appAutoScalingClient, tableId, ns, tableWCUs);
        verifyTarget(appAutoScalingClient, tableId, ns, tableWCUs);
    }

    public static void deletePolicy(ApplicationAutoScalingClient
appAutoScalingClient, String policyName, ScalableDimension tableWCUs,
ServiceNamespace ns, String tableId) {
        try {
            DeleteScalingPolicyRequest delSPRequest =
DeleteScalingPolicyRequest.builder()
                .policyName(policyName)
                .scalableDimension(tableWCUs)
                .serviceNamespace(ns)
                .resourceId(tableId)
                .build();

            appAutoScalingClient.deleteScalingPolicy(delSPRequest);
            System.out.println(policyName +" was deleted successfully.");

        } catch (ApplicationAutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
    }

    // Verify that the scaling policy was deleted
    public static void verifyScalingPolicies(ApplicationAutoScalingClient
appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
tableWCUs) {
        DescribeScalingPoliciesRequest dscRequest =
DescribeScalingPoliciesRequest.builder()
            .scalableDimension(tableWCUs)
            .serviceNamespace(ns)
            .resourceId(tableId)
            .build();
```

```
        DescribeScalingPoliciesResponse response =
 appAutoScalingClient.describeScalingPolicies(dscRequest);
        System.out.println("DescribeScalableTargets result: ");
        System.out.println(response);
    }

    public static void deregisterScalableTarget(ApplicationAutoScalingClient
 appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
 tableWCUs) {
        try {
            DeregisterScalableTargetRequest targetRequest =
 DeregisterScalableTargetRequest.builder()
                .scalableDimension(tableWCUs)
                .serviceNamespace(ns)
                .resourceId(tableId)
                .build();

            appAutoScalingClient.deregisterScalableTarget(targetRequest);
            System.out.println("The scalable target was deregistered.");

        } catch (ApplicationAutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
    }

    public static void verifyTarget(ApplicationAutoScalingClient
 appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
 tableWCUs) {
        DescribeScalableTargetsRequest dscRequest =
 DescribeScalableTargetsRequest.builder()
            .scalableDimension(tableWCUs)
            .serviceNamespace(ns)
            .resourceIds(tableId)
            .build();

        DescribeScalableTargetsResponse response =
 appAutoScalingClient.describeScalableTargets(dscRequest);
        System.out.println("DescribeScalableTargets result: ");
        System.out.println(response);
    }
}
```

- Untuk detail API, lihat DeleteScalingPolicydi Referensi AWS SDK for Java 2.x API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini menghapus kebijakan penskalaan yang ditentukan untuk target scalable Application Auto Scaling.

```
Remove-AASScalingPolicy -ServiceNamespace AppStream -PolicyName "default-scale-
out" -ResourceId fleet/Test -ScalableDimension appstream:fleet:DesiredCapacity
```

- Untuk detail API, lihat DeleteScalingPolicydi Referensi AWS Tools for PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Cmdlet ini menghapus kebijakan penskalaan yang ditentukan untuk target scalable Application Auto Scaling.

```
Remove-AASScalingPolicy -ServiceNamespace AppStream -PolicyName "default-scale-
out" -ResourceId fleet/Test -ScalableDimension appstream:fleet:DesiredCapacity
```

- Untuk detail API, lihat DeleteScalingPolicydi Referensi AWS Tools for PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteScheduledAction`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteScheduledAction`.

CLI

AWS CLI

Untuk menghapus tindakan terjadwal

`delete-scheduled-action`Contoh berikut menghapus tindakan terjadwal yang ditentukan dari armada Amazon AppStream 2.0 yang ditentukan:

```
aws application-autoscaling delete-scheduled-action \
    --service-namespace appstream \
    --scalable-dimension appstream:fleet:DesiredCapacity \
    --resource-id fleet/sample-fleet \
    --scheduled-action-name my-recurring-action
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Penskalaan Terjadwal di Panduan Pengguna Application Auto Scaling.

- Untuk detail API, lihat DeleteScheduledActiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini menghapus tindakan terjadwal yang ditentukan untuk target scalable Application Auto Scaling.

```
Remove-AASScheduledAction -ServiceNamespace AppStream -ScheduledActionName
 WeekDaysFleetScaling -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScheduledAction (DeleteScheduledAction)" on
 target "WeekDaysFleetScaling".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- Untuk detail API, lihat DeleteScheduledActiondi Referensi AWS Tools for PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Cmdlet ini menghapus tindakan terjadwal yang ditentukan untuk target scalable Application Auto Scaling.

```
Remove-AASScheduledAction -ServiceNamespace AppStream -ScheduledActionName
 WeekDaysFleetScaling -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScheduledAction (DeleteScheduledAction)" on
 target "WeekDaysFleetScaling".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- Untuk detail API, lihat DeleteScheduledActiondi Referensi AWS Tools for PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeregisterScalableTarget`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeregisterScalableTarget`.

CLI

AWS CLI

Untuk membatalkan pendaftaran target yang dapat diskalakan

Contoh ini membatalkan pendaftaran target yang dapat diskalakan untuk layanan Amazon ECS yang disebut web-app yang berjalan di klaster default.

Perintah:

```
aws application-autoscaling deregister-scalable-target --service-namespace ecs --
scalable-dimension ecs:service:DesiredCount --resource-id service/default/web-app
```

Contoh ini membatalkan pendaftaran target yang dapat diskalakan untuk sumber daya khusus. custom-resource-idFile.txt berisi string yang mengidentifikasi ID Sumber Daya, yang, untuk sumber daya kustom, adalah jalur ke sumber daya kustom melalui titik akhir Amazon API Gateway Anda.

Perintah:

```
aws application-autoscaling deregister-scalable-target --
service-namespace custom-resource --scalable-dimension custom-
resource:ResourceType:Property --resource-id file://~/custom-resource-id.txt
```

Isi custom-resource-id file.txt:

```
https://example.execute-api.us-west-2.amazonaws.com/prod/
scalableTargetDimensions/1-23456789
```

- Untuk detail API, lihat DeregisterScalableTargetdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini membatalkan pendaftaran target yang dapat diskalakan Application Auto Scaling. Membatalkan pendaftaran target yang dapat diskalakan akan menghapus kebijakan penskalaan yang terkait dengannya.

```
Remove-AASScalableTarget -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity -ServiceNamespace AppStream
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScalableTarget (DeregisterScalableTarget)" on
 target "fleet/MyFleet".
```

```
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- Untuk detail API, lihat DeregisterScalableTargetdi Referensi AWS Tools for PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Cmdlet ini membatalkan pendaftaran target yang dapat diskalakan Application Auto Scaling. Membatalkan pendaftaran target yang dapat diskalakan akan menghapus kebijakan penskalaan yang terkait dengannya.

```
Remove-AASScalableTarget -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity -ServiceNamespace AppStream
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScalableTarget (DeregisterScalableTarget)" on
 target "fleet/MyFleet".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- Untuk detail API, lihat DeregisterScalableTargetdi Referensi AWS Tools for PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DescribeScalableTargets`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DescribeScalableTargets`.

CLI

AWS CLI

Untuk menggambarkan target yang dapat diskalakan

`describe-scalable-targets`Contoh berikut menjelaskan target yang dapat diskalakan untuk namespace `ecs` layanan.

```
aws application-autoscaling describe-scalable-targets \
    --service-namespace ecs
```

Output:

```
{
    "ScalableTargets": [
        {
            "ServiceNamespace": "ecs",
            "ScalableDimension": "ecs:service:DesiredCount",
            "ResourceId": "service/default/web-app",
            "MinCapacity": 1,
            "MaxCapacity": 10,
            "RoleARN": "arn:aws:iam::123456789012:role/
aws-service-role/ecs.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ECSService",
            "CreationTime": 1462558906.199,
            "SuspendedState": {
                "DynamicScalingOutSuspended": false,
                "ScheduledScalingSuspended": false,
                "DynamicScalingInSuspended": false
            },
            "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat AWS layanan yang dapat Anda gunakan dengan Application Auto Scaling di Panduan Pengguna Application Auto Scaling.

- Untuk detail API, lihat DescribeScalableTargetsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini akan memberikan informasi tentang target Application Autoscaling Scaling Scalable di namespace yang ditentukan.

```
Get-AASScalableTarget -ServiceNamespace "AppStream"
```

Output:

```
CreationTime      : 11/7/2019 2:30:03 AM
MaxCapacity       : 5
MinCapacity       : 1
ResourceId        : fleet/Test
RoleARN           : arn:aws:iam::012345678912:role/aws-
service-role/appstream.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_AppStreamFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace  : appstream
SuspendedState    : Amazon.ApplicationAutoScaling.Model.SuspendedState
```

- Untuk detail API, lihat DescribeScalableTargetsdi Referensi AWS Tools for PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini akan memberikan informasi tentang target Application Autoscaling Scaling Scalable di namespace yang ditentukan.

```
Get-AASScalableTarget -ServiceNamespace "AppStream"
```

Output:

```
CreationTime      : 11/7/2019 2:30:03 AM
MaxCapacity       : 5
MinCapacity       : 1
ResourceId        : fleet/Test
RoleARN           : arn:aws:iam::012345678912:role/aws-
service-role/appstream.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_AppStreamFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace  : appstream
SuspendedState    : Amazon.ApplicationAutoScaling.Model.SuspendedState
```

- Untuk detail API, lihat DescribeScalableTargetsdi Referensi AWS Tools for PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DescribeScalingActivities`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DescribeScalingActivities`.

CLI

AWS CLI

Contoh 1: Untuk menjelaskan aktivitas penskalaan untuk layanan Amazon ECS yang ditentukan

`describe-scaling-activities`Contoh berikut menjelaskan aktivitas penskalaan untuk layanan Amazon ECS `web-app` yang disebut yang berjalan di `default` cluster. Output menunjukkan aktivitas penskalaan yang diprakarsai oleh kebijakan penskalaan.

```
aws application-autoscaling describe-scaling-activities \
    --service-namespace ecs \
    --resource-id service/default/web-app
```

Output:

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "ecs:service:DesiredCount",
            "Description": "Setting desired count to 1.",
            "ResourceId": "service/default/web-app",
            "ActivityId": "e6c5f7d1-dbbb-4a3f-89b2-51f33e766399",
            "StartTime": 1462575838.171,
            "ServiceNamespace": "ecs",
            "EndTime": 1462575872.111,
            "Cause": "monitor alarm web-app-cpu-lt-25 in state ALARM triggered
 policy web-app-cpu-lt-25",
            "StatusMessage": "Successfully set desired count to 1. Change
 successfully fulfilled by ecs.",
            "StatusCode": "Successful"
        }
    ]
```

```
}
```

Untuk informasi selengkapnya, lihat Aktivitas penskalaan untuk Application Auto Scaling di Panduan Pengguna Application Auto Scaling.

Contoh 2: Untuk menggambarkan aktivitas penskalaan untuk tabel DynamoDB yang ditentukan

`describe-scaling-activities`Contoh berikut menjelaskan aktivitas penskalaan untuk tabel DynamoDB yang disebut. `TestTable` Output menunjukkan aktivitas penskalaan yang diprakarsai oleh dua tindakan terjadwal yang berbeda.

```
aws application-autoscaling describe-scaling-activities \
    --service-namespace dynamodb \
    --resource-id table/TestTable
```

Output:

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 10.",
            "ResourceId": "table/my-table",
            "ActivityId": "4d1308c0-bbcf-4514-a673-b0220ae38547",
            "StartTime": 1561574415.086,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574449.51,
            "Cause": "maximum capacity was set to 10",
            "StatusMessage": "Successfully set write capacity units to 10. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 5 and max capacity to 10",
            "ResourceId": "table/my-table",
            "ActivityId": "f2b7847b-721d-4e01-8ef0-0c8d3bacc1c7",
            "StartTime": 1561574414.644,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-second-scheduled-action was
 triggered",
```

```
            "StatusMessage": "Successfully set min capacity to 5 and max capacity
 to 10",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 15.",
            "ResourceId": "table/my-table",
            "ActivityId": "d8ea4de6-9eaa-499f-b466-2cc5e681ba8b",
            "StartTime": 1561574108.904,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574140.255,
            "Cause": "minimum capacity was set to 15",
            "StatusMessage": "Successfully set write capacity units to 15. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 15 and max capacity to 20",
            "ResourceId": "table/my-table",
            "ActivityId": "3250fd06-6940-4e8e-bb1f-d494db7554d2",
            "StartTime": 1561574108.512,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-first-scheduled-action was
 triggered",
            "StatusMessage": "Successfully set min capacity to 15 and max
 capacity to 20",
            "StatusCode": "Successful"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Aktivitas penskalaan untuk Application Auto Scaling di Panduan Pengguna Application Auto Scaling.

- Untuk detail API, lihat DescribeScalingActivitiesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Memberikan informasi deskriptif tentang aktivitas penskalaan di namespace yang ditentukan dari enam minggu sebelumnya.

```
Get-AASScalingActivity -ServiceNamespace AppStream
```

Output:

```
ActivityId        : 2827409f-b639-4cdb-a957-8055d5d07434
Cause             : monitor alarm Appstream2-MyFleet-default-scale-in-Alarm in
 state ALARM triggered policy default-scale-in
Description       : Setting desired capacity to 2.
Details           :
EndTime           : 12/14/2019 11:32:49 AM
ResourceId        : fleet/MyFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace  : appstream
StartTime         : 12/14/2019 11:32:14 AM
StatusCode        : Successful
StatusMessage     : Successfully set desired capacity to 2. Change successfully
 fulfilled by appstream.
```

- Untuk detail API, lihat DescribeScalingActivitiesdi Referensi AWS Tools for PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Memberikan informasi deskriptif tentang aktivitas penskalaan di namespace yang ditentukan dari enam minggu sebelumnya.

```
Get-AASScalingActivity -ServiceNamespace AppStream
```

Output:

```
ActivityId        : 2827409f-b639-4cdb-a957-8055d5d07434
Cause             : monitor alarm Appstream2-MyFleet-default-scale-in-Alarm in
 state ALARM triggered policy default-scale-in
Description       : Setting desired capacity to 2.
Details           :
EndTime           : 12/14/2019 11:32:49 AM
```

```
ResourceId        : fleet/MyFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace  : appstream
StartTime         : 12/14/2019 11:32:14 AM
StatusCode        : Successful
StatusMessage     : Successfully set desired capacity to 2. Change successfully
 fulfilled by appstream.
```

- Untuk detail API, lihat DescribeScalingActivitiesdi Referensi AWS Tools for PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DescribeScalingPolicies`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DescribeScalingPolicies`.

CLI

AWS CLI

Untuk menggambarkan kebijakan penskalaan

Perintah contoh ini menjelaskan kebijakan penskalaan untuk namespace layanan ecs.

Perintah:

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs
```

Output:

```
{
    "ScalingPolicies": [
        {
            "PolicyName": "web-app-cpu-gt-75",
            "ScalableDimension": "ecs:service:DesiredCount",
            "ResourceId": "service/default/web-app",
            "CreationTime": 1462561899.23,
            "StepScalingPolicyConfiguration": {
                "Cooldown": 60,
```

```
                "StepAdjustments": [
                    {
                        "ScalingAdjustment": 200,
                        "MetricIntervalLowerBound": 0.0
                    }
                ],
                "AdjustmentType": "PercentChangeInCapacity"
            },
            "PolicyARN": "arn:aws:autoscaling:us-
west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/
ecs/service/default/web-app:policyName/web-app-cpu-gt-75",
            "PolicyType": "StepScaling",
            "Alarms": [
                {
                    "AlarmName": "web-app-cpu-gt-75",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:web-app-cpu-gt-75"
                }
            ],
            "ServiceNamespace": "ecs"
        },
        {
            "PolicyName": "web-app-cpu-lt-25",
            "ScalableDimension": "ecs:service:DesiredCount",
            "ResourceId": "service/default/web-app",
            "CreationTime": 1462562575.099,
            "StepScalingPolicyConfiguration": {
                "Cooldown": 1,
                "StepAdjustments": [
                    {
                        "ScalingAdjustment": -50,
                        "MetricIntervalUpperBound": 0.0
                    }
                ],
                "AdjustmentType": "PercentChangeInCapacity"
            },
            "PolicyARN": "arn:aws:autoscaling:us-
west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/
ecs/service/default/web-app:policyName/web-app-cpu-lt-25",
            "PolicyType": "StepScaling",
            "Alarms": [
                {
                    "AlarmName": "web-app-cpu-lt-25",
```

```
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:web-app-cpu-lt-25"
                }
            ],
            "ServiceNamespace": "ecs"
        }
    ]
}
```

- Untuk detail API, lihat DescribeScalingPoliciesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini menjelaskan kebijakan penskalaan Application Auto Scaling untuk namespace layanan yang ditentukan.

```
Get-AASScalingPolicy -ServiceNamespace AppStream
```

Output:

```
Alarms                                    : {Appstream2-LabFleet-default-scale-
out-Alarm}
CreationTime                              : 9/3/2019 2:48:15 AM
PolicyARN                                 : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                                            policyName/default-scale-out
PolicyName                                : default-scale-out
PolicyType                                : StepScaling
ResourceId                                : fleet/LabFleet
ScalableDimension                         : appstream:fleet:DesiredCapacity
ServiceNamespace                          : appstream
StepScalingPolicyConfiguration            :
 Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :

Alarms                                    : {Appstream2-LabFleet-default-scale-in-
Alarm}
CreationTime                              : 9/3/2019 2:48:15 AM
```

```
PolicyARN                                   : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                                              policyName/default-scale-in
PolicyName                                  : default-scale-in
PolicyType                                  : StepScaling
ResourceId                                  : fleet/LabFleet
ScalableDimension                           : appstream:fleet:DesiredCapacity
ServiceNamespace                            : appstream
StepScalingPolicyConfiguration              :
 Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :
```

- Untuk detail API, lihat DescribeScalingPoliciesdi Referensi AWS Tools for PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Cmdlet ini menjelaskan kebijakan penskalaan Application Auto Scaling untuk namespace layanan yang ditentukan.

```
Get-AASScalingPolicy -ServiceNamespace AppStream
```

Output:

```
Alarms                                      : {Appstream2-LabFleet-default-scale-
out-Alarm}
CreationTime                                : 9/3/2019 2:48:15 AM
PolicyARN                                   : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                                              policyName/default-scale-out
PolicyName                                  : default-scale-out
PolicyType                                  : StepScaling
ResourceId                                  : fleet/LabFleet
ScalableDimension                           : appstream:fleet:DesiredCapacity
ServiceNamespace                            : appstream
StepScalingPolicyConfiguration              :
 Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :

Alarms                                      : {Appstream2-LabFleet-default-scale-in-
Alarm}
```

```
CreationTime                                : 9/3/2019 2:48:15 AM
PolicyARN                                   : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                                              policyName/default-scale-in
PolicyName                                  : default-scale-in
PolicyType                                  : StepScaling
ResourceId                                  : fleet/LabFleet
ScalableDimension                           : appstream:fleet:DesiredCapacity
ServiceNamespace                            : appstream
StepScalingPolicyConfiguration              :
 Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :
```

- Untuk detail API, lihat DescribeScalingPoliciesdi Referensi AWS Tools for PowerShell Cmdlet (V5).

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
async fn show_policies(client: &Client) -> Result<(), Error> {
    let response = client
        .describe_scaling_policies()
        .service_namespace(ServiceNamespace::Ec2)
        .send()
        .await?;
    println!("Auto Scaling Policies:");
    for policy in response.scaling_policies() {
        println!("{:?}\n", policy);
    }
    println!("Next token: {:?}", response.next_token());

    Ok(())
}
```

- Untuk detail API, lihat DescribeScalingPoliciesreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DescribeScheduledActions`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DescribeScheduledActions`.

CLI

AWS CLI

Untuk menggambarkan tindakan terjadwal

`describe-scheduled-actions`Contoh berikut menampilkan rincian untuk tindakan terjadwal untuk namespace layanan tertentu:

```
aws application-autoscaling describe-scheduled-actions \
    --service-namespace dynamodb
```

Output:

```
{
    "ScheduledActions": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Schedule": "at(2019-05-20T18:35:00)",
            "ResourceId": "table/my-table",
            "CreationTime": 1561571888.361,
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:2d36aa3b-cdf9-4565-
b290-81db519b227d:resource/dynamodb/table/my-table:scheduledActionName/my-first-
scheduled-action",
            "ScalableTargetAction": {
                "MinCapacity": 15,
                "MaxCapacity": 20
            },
            "ScheduledActionName": "my-first-scheduled-action",
```

```
            "ServiceNamespace": "dynamodb"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Schedule": "at(2019-05-20T18:40:00)",
            "ResourceId": "table/my-table",
            "CreationTime": 1561571946.021,
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:2d36aa3b-cdf9-4565-
b290-81db519b227d:resource/dynamodb/table/my-table:scheduledActionName/my-second-
scheduled-action",
            "ScalableTargetAction": {
                "MinCapacity": 5,
                "MaxCapacity": 10
            },
            "ScheduledActionName": "my-second-scheduled-action",
            "ServiceNamespace": "dynamodb"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Penskalaan Terjadwal di Panduan Pengguna Application Auto Scaling.

- Untuk detail API, lihat DescribeScheduledActionsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini mencantumkan tindakan yang dijadwalkan untuk grup Auto Scaling Anda yang belum berjalan atau yang belum mencapai waktu akhirnya.

```
Get-AASScheduledAction -ServiceNamespace AppStream
```

Output:

```
CreationTime         : 12/22/2019 9:25:52 AM
EndTime              : 1/1/0001 12:00:00 AM
ResourceId           : fleet/MyFleet
ScalableDimension    : appstream:fleet:DesiredCapacity
ScalableTargetAction : Amazon.ApplicationAutoScaling.Model.ScalableTargetAction
```

```
Schedule             : cron(0 0 8 ? * MON-FRI *)
ScheduledActionARN   : arn:aws:autoscaling:us-
west-2:012345678912:scheduledAction:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:scheduledActionName
                       /WeekDaysFleetScaling
ScheduledActionName  : WeekDaysFleetScaling
ServiceNamespace     : appstream
StartTime            : 1/1/0001 12:00:00 AM
```

- Untuk detail API, lihat DescribeScheduledActionsdi Referensi AWS Tools for PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Cmdlet ini mencantumkan tindakan yang dijadwalkan untuk grup Auto Scaling Anda yang belum berjalan atau yang belum mencapai waktu akhirnya.

```
Get-AASScheduledAction -ServiceNamespace AppStream
```

Output:

```
CreationTime         : 12/22/2019 9:25:52 AM
EndTime              : 1/1/0001 12:00:00 AM
ResourceId           : fleet/MyFleet
ScalableDimension    : appstream:fleet:DesiredCapacity
ScalableTargetAction : Amazon.ApplicationAutoScaling.Model.ScalableTargetAction
Schedule             : cron(0 0 8 ? * MON-FRI *)
ScheduledActionARN   : arn:aws:autoscaling:us-
west-2:012345678912:scheduledAction:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:scheduledActionName
                       /WeekDaysFleetScaling
ScheduledActionName  : WeekDaysFleetScaling
ServiceNamespace     : appstream
StartTime            : 1/1/0001 12:00:00 AM
```

- Untuk detail API, lihat DescribeScheduledActionsdi Referensi AWS Tools for PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutScalingPolicy`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`PutScalingPolicy`.

CLI

AWS CLI

Contoh 1: Untuk menerapkan kebijakan penskalaan pelacakan target dengan spesifikasi metrik yang telah ditentukan

`put-scaling-policy`Contoh berikut menerapkan kebijakan penskalaan pelacakan target dengan spesifikasi metrik yang telah ditentukan sebelumnya ke layanan Amazon ECS yang disebut web-app di kluster default. Kebijakan tersebut menjaga pemanfaatan CPU rata-rata layanan pada 75 persen, dengan periode cooldown scale-out dan scale-in 60 detik. Output berisi ARNs dan nama dari dua CloudWatch alarm yang dibuat atas nama Anda.

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
--scalable-dimension ecs:service:DesiredCount \
--resource-id service/default/web-app \
--policy-name cpu75-target-tracking-scaling-policy --policy-
type TargetTrackingScaling \
--target-tracking-scaling-policy-configuration file://config.json
```

Contoh ini mengasumsikan bahwa Anda memiliki file config.json di direktori saat ini dengan konten berikut:

```
{
     "TargetValue": 75.0,
     "PredefinedMetricSpecification": {
         "PredefinedMetricType": "ECSServiceAverageCPUUtilization"
     },
     "ScaleOutCooldown": 60,
    "ScaleInCooldown": 60
}
```

Output:

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/
ecs/service/default/web-app:policyName/cpu75-target-tracking-scaling-policy",
```

```
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-AlarmHigh-
d4f0770c-b46e-434a-a60f-3b36d653feca",
            "AlarmName": "TargetTracking-service/default/web-app-AlarmHigh-
d4f0770c-b46e-434a-a60f-3b36d653feca"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-
AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d",
            "AlarmName": "TargetTracking-service/default/web-app-
AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
        }
    ]
}
```

Contoh 2: Untuk menerapkan kebijakan penskalaan pelacakan target dengan spesifikasi metrik yang disesuaikan

`put-scaling-policy`Contoh berikut menerapkan kebijakan penskalaan pelacakan target dengan spesifikasi metrik yang disesuaikan ke layanan Amazon ECS yang disebut web-app di kluster default. Kebijakan tersebut menjaga pemanfaatan rata-rata layanan pada 75 persen, dengan periode cooldown scale-out dan scale-in 60 detik. Output berisi ARNs dan nama dari dua CloudWatch alarm yang dibuat atas nama Anda.

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
--scalable-dimension ecs:service:DesiredCount \
--resource-id service/default/web-app \
--policy-name cms75-target-tracking-scaling-policy
--policy-type TargetTrackingScaling \
--target-tracking-scaling-policy-configuration file://config.json
```

Contoh ini mengasumsikan bahwa Anda memiliki file config.json di direktori saat ini dengan konten berikut:

```
{
    "TargetValue":75.0,
    "CustomizedMetricSpecification":{
        "MetricName":"MyUtilizationMetric",
```

```
        "Namespace":"MyNamespace",
        "Dimensions": [
            {
                "Name":"MyOptionalMetricDimensionName",
                "Value":"MyOptionalMetricDimensionValue"
            }
        ],
        "Statistic":"Average",
        "Unit":"Percent"
    },
    "ScaleOutCooldown": 60,
    "ScaleInCooldown": 60
}
```

Output:

```
{
    "PolicyARN": "arn:aws:autoscaling:us-west-2:012345678910:scalingPolicy:
 8784a896-b2ba-47a1-b08c-27301cc499a1:resource/ecs/service/default/web-
app:policyName/cms75-target-tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0",
            "AlarmName": "TargetTracking-service/default/web-app-
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4",
            "AlarmName": "TargetTracking-service/default/web-app-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4"
        }
    ]
}
```

Contoh 3: Untuk menerapkan kebijakan penskalaan pelacakan target hanya untuk skala keluar

`put-scaling-policy`Contoh berikut menerapkan kebijakan penskalaan pelacakan target ke layanan Amazon ECS yang dipanggil `web-app` di klaster default. Kebijakan ini digunakan untuk meningkatkan skala layanan ECS ketika `RequestCountPerTarget` metrik

dari Application Load Balancer melebihi ambang batas. Outputnya berisi ARN dan nama CloudWatch alarm yang dibuat atas nama Anda.

```
aws application-autoscaling put-scaling-policy \
    --service-namespace ecs \
    --scalable-dimension ecs:service:DesiredCount \
    --resource-id service/default/web-app \
    --policy-name alb-scale-out-target-tracking-scaling-policy \
    --policy-type TargetTrackingScaling \
    --target-tracking-scaling-policy-configuration file://config.json
```

Isi dari `config.json`:

```
{
     "TargetValue": 1000.0,
     "PredefinedMetricSpecification": {
         "PredefinedMetricType": "ALBRequestCountPerTarget",
         "ResourceLabel": "app/EC2Co-EcsEl-1TKLTMITMM0EO/f37c06a68c1748aa/
targetgroup/EC2Co-Defau-LDNM7Q3ZH1ZN/6d4ea56ca2d6a18d"
     },
     "ScaleOutCooldown": 60,
    "ScaleInCooldown": 60,
    "DisableScaleIn": true
}
```

Output:

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
west-2:123456789012:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/
ecs/service/default/web-app:policyName/alb-scale-out-target-tracking-scaling-
policy",
    "Alarms": [
        {
            "AlarmName": "TargetTracking-service/default/web-app-AlarmHigh-
d4f0770c-b46e-434a-a60f-3b36d653feca",
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-service/default/web-app-AlarmHigh-
d4f0770c-b46e-434a-a60f-3b36d653feca"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Kebijakan Penskalaan Pelacakan Target untuk Application Auto Scaling di AWS Panduan Pengguna Application Auto Scaling.

- Untuk detail API, lihat PutScalingPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini membuat atau memperbarui kebijakan untuk target yang dapat diskalakan Application Auto Scaling. Setiap target yang dapat diskalakan diidentifikasi oleh namespace layanan, ID sumber daya, dan dimensi yang dapat diskalakan.

```
Set-AASScalingPolicy -ServiceNamespace AppStream -PolicyName ASFleetScaleInPolicy
 -PolicyType StepScaling  -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity -StepScalingPolicyConfiguration_AdjustmentType
 ChangeInCapacity -StepScalingPolicyConfiguration_Cooldown 360
 -StepScalingPolicyConfiguration_MetricAggregationType Average -
StepScalingPolicyConfiguration_StepAdjustments @{ScalingAdjustment = -1;
 MetricIntervalUpperBound = 0}
```

Output:

```
Alarms    PolicyARN
------    ---------
{}        arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:policyName/ASFleetScaleInPolicy
```

- Untuk detail API, lihat PutScalingPolicydi Referensi AWS Tools for PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Cmdlet ini membuat atau memperbarui kebijakan untuk target yang dapat diskalakan Application Auto Scaling. Setiap target yang dapat diskalakan diidentifikasi oleh namespace layanan, ID sumber daya, dan dimensi yang dapat diskalakan.

```
Set-AASScalingPolicy -ServiceNamespace AppStream -PolicyName ASFleetScaleInPolicy
 -PolicyType StepScaling  -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity -StepScalingPolicyConfiguration_AdjustmentType
 ChangeInCapacity -StepScalingPolicyConfiguration_Cooldown 360
```

```
 -StepScalingPolicyConfiguration_MetricAggregationType Average -
StepScalingPolicyConfiguration_StepAdjustments @{ScalingAdjustment = -1;
 MetricIntervalUpperBound = 0}
```

Output:

```
Alarms    PolicyARN
------    ---------
{}        arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:policyName/ASFleetScaleInPolicy
```

- Untuk detail API, lihat PutScalingPolicydi Referensi AWS Tools for PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutScheduledAction`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`PutScheduledAction`.

CLI

AWS CLI

Untuk menambahkan tindakan terjadwal ke tabel DynamoDB

Contoh ini menambahkan tindakan terjadwal ke tabel DynamoDB TestTable dipanggil untuk skala keluar pada jadwal berulang. Pada jadwal yang ditentukan (setiap hari pukul 12:15 WIB), jika kapasitas saat ini di bawah nilai yang ditentukan MinCapacity, Application Auto Scaling sesuai dengan nilai yang ditentukan oleh. MinCapacity

Perintah:

```
aws application-autoscaling put-scheduled-action --service-
namespace dynamodb --scheduled-action-name my-recurring-action --
schedule "cron(15 12 * * ? *)" --resource-id table/TestTable --
scalable-dimension dynamodb:table:WriteCapacityUnits --scalable-target-
action MinCapacity=6
```

Untuk informasi selengkapnya, lihat Penskalaan Terjadwal di Panduan Pengguna Application Auto Scaling.

- Untuk detail API, lihat PutScheduledActiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini membuat atau memperbarui tindakan terjadwal untuk target scalable Application Auto Scaling. Setiap target yang dapat diskalakan diidentifikasi oleh namespace layanan, ID sumber daya, dan dimensi yang dapat diskalakan.

```
Set-AASScheduledAction -ServiceNamespace AppStream -ResourceId fleet/
MyFleet -Schedule "cron(0 0 8 ? * MON-FRI *)" -ScalableDimension
 appstream:fleet:DesiredCapacity -ScheduledActionName WeekDaysFleetScaling -
ScalableTargetAction_MinCapacity 5 -ScalableTargetAction_MaxCapacity 10
```

- Untuk detail API, lihat PutScheduledActiondi Referensi AWS Tools for PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Cmdlet ini membuat atau memperbarui tindakan terjadwal untuk target scalable Application Auto Scaling. Setiap target yang dapat diskalakan diidentifikasi oleh namespace layanan, ID sumber daya, dan dimensi yang dapat diskalakan.

```
Set-AASScheduledAction -ServiceNamespace AppStream -ResourceId fleet/
MyFleet -Schedule "cron(0 0 8 ? * MON-FRI *)" -ScalableDimension
 appstream:fleet:DesiredCapacity -ScheduledActionName WeekDaysFleetScaling -
ScalableTargetAction_MinCapacity 5 -ScalableTargetAction_MaxCapacity 10
```

- Untuk detail API, lihat PutScheduledActiondi Referensi AWS Tools for PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan `RegisterScalableTarget` dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`RegisterScalableTarget`.

CLI

AWS CLI

Contoh 1: Untuk mendaftarkan layanan ECS sebagai target yang dapat diskalakan

`register-scalable-target`Contoh berikut mendaftarkan layanan Amazon ECS dengan Application Auto Scaling. Ini juga menambahkan tag dengan nama kunci `environment` dan nilai `production` ke target yang dapat diskalakan.

```
aws application-autoscaling register-scalable-target \
    --service-namespace ecs \
    --scalable-dimension ecs:service:DesiredCount \
    --resource-id service/default/web-app \
    --min-capacity 1 --max-capacity 10 \
    --tags environment=production
```

Output:

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Untuk contoh untuk AWS layanan lain dan sumber daya khusus, lihat topik dalam AWS layanan yang dapat Anda gunakan dengan Application Auto Scaling di Panduan Pengguna Application Auto Scaling.

Contoh 2: Untuk menangguhkan aktivitas penskalaan untuk target yang dapat diskalakan

`register-scalable-target`Contoh berikut menangguhkan aktivitas penskalaan untuk target skalabel yang ada.

```
aws application-autoscaling register-scalable-target \
    --service-namespace dynamodb \
    --scalable-dimension dynamodb:table:ReadCapacityUnits \
    --resource-id table/my-table \
```

```
    --suspended-
state DynamicScalingInSuspended=true,DynamicScalingOutSuspended=true,ScheduledScalingSusp
```

Output:

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Untuk informasi selengkapnya, lihat Menangguhkan dan melanjutkan penskalaan untuk Application Auto Scaling di Panduan Pengguna Application Auto Scaling.

Contoh 3: Untuk melanjutkan aktivitas penskalaan untuk target yang dapat diskalakan

`register-scalable-target`Contoh berikut melanjutkan aktivitas penskalaan untuk target skalabel yang ada.

```
aws application-autoscaling register-scalable-target \
    --service-namespace dynamodb \
    --scalable-dimension dynamodb:table:ReadCapacityUnits \
    --resource-id table/my-table \
    --suspended-
state DynamicScalingInSuspended=false,DynamicScalingOutSuspended=false,ScheduledScalingSu
```

Output:

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Untuk informasi selengkapnya, lihat Menangguhkan dan melanjutkan penskalaan untuk Application Auto Scaling di Panduan Pengguna Application Auto Scaling.

- Untuk detail API, lihat RegisterScalableTargetdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.applicationautoscaling.ApplicationAutoScalingClient;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ApplicationAutoScalingExcep
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalableTargetsRequ
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalableTargetsResp
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalingPoliciesRequ
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalingPoliciesResp
import software.amazon.awssdk.services.applicationautoscaling.model.PolicyType;
import
 software.amazon.awssdk.services.applicationautoscaling.model.PredefinedMetricSpecificati
import
 software.amazon.awssdk.services.applicationautoscaling.model.PutScalingPolicyRequest;
import
 software.amazon.awssdk.services.applicationautoscaling.model.RegisterScalableTargetReque
import
 software.amazon.awssdk.services.applicationautoscaling.model.ScalingPolicy;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ServiceNamespace;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ScalableDimension;
import software.amazon.awssdk.services.applicationautoscaling.model.MetricType;
import
 software.amazon.awssdk.services.applicationautoscaling.model.TargetTrackingScalingPolicy
import java.util.List;

/**
```

```
 * Before running this Java V2 code example, set up your development environment,
 including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class EnableDynamoDBAutoscaling {
    public static void main(String[] args) {
        final String usage = """

            Usage:
               <tableId> <roleARN> <policyName>\s

            Where:
               tableId - The table Id value (for example, table/Music).
               roleARN - The ARN of the role that has ApplicationAutoScaling
 permissions.
               policyName - The name of the policy to create.

            """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        System.out.println("This example registers an Amazon DynamoDB table,
 which is the resource to scale.");
        String tableId = args[0];
        String roleARN = args[1];
        String policyName = args[2];
        ServiceNamespace ns = ServiceNamespace.DYNAMODB;
        ScalableDimension tableWCUs =
 ScalableDimension.DYNAMODB_TABLE_WRITE_CAPACITY_UNITS;
        ApplicationAutoScalingClient appAutoScalingClient =
 ApplicationAutoScalingClient.builder()
            .region(Region.US_EAST_1)
            .build();

        registerScalableTarget(appAutoScalingClient, tableId, roleARN, ns,
 tableWCUs);
        verifyTarget(appAutoScalingClient, tableId, ns, tableWCUs);
```

```
        configureScalingPolicy(appAutoScalingClient, tableId, ns, tableWCUs,
policyName);
    }

    public static void registerScalableTarget(ApplicationAutoScalingClient
appAutoScalingClient, String tableId, String roleARN, ServiceNamespace ns,
ScalableDimension tableWCUs) {
        try {
            RegisterScalableTargetRequest targetRequest =
RegisterScalableTargetRequest.builder()
                .serviceNamespace(ns)
                .scalableDimension(tableWCUs)
                .resourceId(tableId)
                .roleARN(roleARN)
                .minCapacity(5)
                .maxCapacity(10)
                .build();

            appAutoScalingClient.registerScalableTarget(targetRequest);
            System.out.println("You have registered " + tableId);

        } catch (ApplicationAutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
    }

    // Verify that the target was created.
    public static void verifyTarget(ApplicationAutoScalingClient
appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
tableWCUs) {
        DescribeScalableTargetsRequest dscRequest =
DescribeScalableTargetsRequest.builder()
            .scalableDimension(tableWCUs)
            .serviceNamespace(ns)
            .resourceIds(tableId)
            .build();

        DescribeScalableTargetsResponse response =
appAutoScalingClient.describeScalableTargets(dscRequest);
        System.out.println("DescribeScalableTargets result: ");
        System.out.println(response);
    }

    // Configure a scaling policy.
```

```
   public static void configureScalingPolicy(ApplicationAutoScalingClient
appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
tableWCUs, String policyName) {
       // Check if the policy exists before creating a new one.
       DescribeScalingPoliciesResponse describeScalingPoliciesResponse =
appAutoScalingClient.describeScalingPolicies(DescribeScalingPoliciesRequest.builder()
           .serviceNamespace(ns)
           .resourceId(tableId)
           .scalableDimension(tableWCUs)
           .build());

       if (!describeScalingPoliciesResponse.scalingPolicies().isEmpty()) {
           // If policies exist, consider updating an existing policy instead of
creating a new one.
           System.out.println("Policy already exists. Consider updating it
instead.");
           List<ScalingPolicy> polList =
describeScalingPoliciesResponse.scalingPolicies();
           for (ScalingPolicy pol : polList) {
               System.out.println("Policy name:" +pol.policyName());
           }
       } else {
           // If no policies exist, proceed with creating a new policy.
           PredefinedMetricSpecification specification =
PredefinedMetricSpecification.builder()

.predefinedMetricType(MetricType.DYNAMO_DB_WRITE_CAPACITY_UTILIZATION)
               .build();

           TargetTrackingScalingPolicyConfiguration policyConfiguration =
TargetTrackingScalingPolicyConfiguration.builder()
               .predefinedMetricSpecification(specification)
               .targetValue(50.0)
               .scaleInCooldown(60)
               .scaleOutCooldown(60)
               .build();

           PutScalingPolicyRequest putScalingPolicyRequest =
PutScalingPolicyRequest.builder()
               .targetTrackingScalingPolicyConfiguration(policyConfiguration)
               .serviceNamespace(ns)
               .scalableDimension(tableWCUs)
               .resourceId(tableId)
               .policyName(policyName)
```

```
                .policyType(PolicyType.TARGET_TRACKING_SCALING)
                .build();

            try {
                appAutoScalingClient.putScalingPolicy(putScalingPolicyRequest);
                System.out.println("You have successfully created a scaling
 policy for an Application Auto Scaling scalable target");
            } catch (ApplicationAutoScalingException e) {
                System.err.println("Error: " +
 e.awsErrorDetails().errorMessage());
            }
        }
    }
}
```

- Untuk detail API, lihat RegisterScalableTargetdi Referensi AWS SDK for Java 2.x API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini mendaftarkan atau memperbarui target yang dapat diskalakan. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling.

```
Add-AASScalableTarget -ServiceNamespace AppStream -ResourceId fleet/MyFleet -
ScalableDimension appstream:fleet:DesiredCapacity -MinCapacity 2 -MaxCapacity 10
```

- Untuk detail API, lihat RegisterScalableTargetdi Referensi AWS Tools for PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Cmdlet ini mendaftarkan atau memperbarui target yang dapat diskalakan. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling.

```
Add-AASScalableTarget -ServiceNamespace AppStream -ResourceId fleet/MyFleet -
ScalableDimension appstream:fleet:DesiredCapacity -MinCapacity 2 -MaxCapacity 10
```

- Untuk detail API, lihat RegisterScalableTargetdi Referensi AWS Tools for PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Dukungan penandaan untuk Application Auto Scaling

Anda dapat menggunakan AWS CLI atau SDK untuk menandai target Application Auto Scaling yang dapat diskalakan. Target yang dapat diskalakan adalah entitas yang mewakili AWS atau sumber daya kustom yang dapat diskalakan oleh Application Auto Scaling.

Setiap tag adalah label yang terdiri dari kunci dan nilai yang ditentukan pengguna menggunakan Application Auto Scaling API. Tag dapat membantu Anda mengonfigurasi akses terperinci ke target tertentu yang dapat diskalakan sesuai dengan kebutuhan organisasi Anda. Untuk informasi selengkapnya, lihat ABAC dengan Application Auto Scaling.

Anda dapat menambahkan tag ke target baru yang dapat diskalakan saat Anda mendaftarkannya, atau Anda dapat menambahkannya ke target skalabel yang ada.

Perintah yang umum digunakan untuk mengelola tag meliputi:

- register-scalable-targetuntuk menandai target baru yang dapat diskalakan saat Anda mendaftarkannya.
- tag-resource untuk menambahkan tag ke target skalabel yang ada.
- list-tags-for-resourceuntuk mengembalikan tag pada target yang dapat diskalakan.
- untag-resource untuk menghapus tag.

Contoh penandaan

Gunakan register-scalable-targetperintah berikut dengan `--tags` opsi. Contoh ini menandai target yang dapat diskalakan dengan dua tag: kunci tag bernama **`environment`** dengan nilai tag dari**`production`**, dan kunci tag bernama **`iscontainerbased`** dengan nilai **`true`** tag.

Ganti nilai sampel untuk `--min-capacity` dan `--max-capacity` dan contoh teks `--service-namespace` dengan namespace AWS layanan yang Anda gunakan dengan Application Auto Scaling`--scalable-dimension`, dengan dimensi skalabel yang terkait dengan sumber daya yang Anda daftarkan, `--resource-id` dan dengan pengenal untuk sumber daya. Untuk informasi selengkapnya dan contoh untuk setiap layanan, lihat topik diLayanan AWS yang dapat Anda gunakan dengan Application Auto Scaling.

```
aws application-autoscaling register-scalable-target \
```

```
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --min-capacity 1 --max-capacity 10 \
  --tags environment=production,iscontainerbased=true
```

Jika berhasil, perintah ini mengembalikan ARN dari target yang dapat diskalakan.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

> **Note**
>
> Jika perintah ini menimbulkan kesalahan, pastikan Anda telah memperbarui AWS CLI secara lokal ke versi terbaru.

Tag untuk keamanan

Gunakan tag untuk memverifikasi bahwa pemohon (seperti pengguna atau peran IAM) memiliki izin untuk melakukan tindakan tertentu. Berikan informasi tag dalam elemen kondisi kebijakan IAM dengan menggunakan satu atau beberapa kunci kondisi berikut:

- Gunakan `aws:ResourceTag/tag-key: tag-value` untuk mengizinkan (atau menolak) tindakan pengguna pada target yang dapat diskalakan dengan tag tertentu.
- Gunakan `aws:RequestTag/tag-key: tag-value` untuk meminta tag khusus dapat ada (atau tidak ada) dalam permintaan.
- Gunakan `aws:TagKeys [tag-key, ...]` untuk meminta kunci khusus dapat ada (atau tidak ada) dalam permintaan.

Misalnya, kebijakan IAM berikut memberikan izin untuk menggunakan`DeregisterScalableTarget`,`DeleteScalingPolicy`, dan tindakan. `DeleteScheduledAction` Namun, itu juga menyangkal tindakan jika target yang dapat diskalakan yang ditindaklanjuti memiliki tag **`environment`** =. **`production`**

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:DeregisterScalableTarget",
                "application-autoscaling:DeleteScalingPolicy",
                "application-autoscaling:DeleteScheduledAction"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "application-autoscaling:DeregisterScalableTarget",
                "application-autoscaling:DeleteScalingPolicy",
                "application-autoscaling:DeleteScheduledAction"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/environment": "production"
                }
            }
        }
    ]
}
```

Kontrol akses ke tag

Gunakan tag untuk memverifikasi bahwa pemohon (seperti pengguna atau peran IAM) memiliki izin untuk menambahkan, memodifikasi, atau menghapus tag untuk target yang dapat diskalakan.

Misalnya, Anda dapat membuat kebijakan IAM yang memungkinkan penghapusan hanya tag dengan **`temporary`** kunci dari target yang dapat diskalakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "application-autoscaling:UntagResource",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": { "aws:TagKeys": ["temporary"] }
            }
        }
    ]
}
```

Keamanan dalam Application Auto Scaling

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. Model tanggung jawab bersama menjelaskan hal ini sebagai keamanan dari cloud dan keamanan dalam cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari program AWS kepatuhan program AWS . Untuk mempelajari tentang program kepatuhan yang berlaku untuk Application Auto Scaling, lihat AWS layanan dalam cakupan berdasarkan AWS layanan program kepatuhan .
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain termasuk sensitivitas data Anda, persyaratan perusahaan Anda, serta undang-undang dan peraturan yang berlaku.

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan Application Auto Scaling. Topik berikut menunjukkan kepada Anda cara mengonfigurasi Application Auto Scaling untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga mempelajari cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan sumber daya Application Auto Scaling Anda.

Daftar Isi

Perlindungan data dalam Application Auto Scaling

Model tanggung jawab AWS bersama model berlaku untuk perlindungan data dalam Application Auto Scaling. Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk Layanan AWS yang Anda gunakan. Lihat informasi yang lebih lengkap tentang privasi data dalam Pertanyaan Umum Privasi Data. Lihat informasi tentang perlindungan data di Eropa di pos blog Model Tanggung Jawab Bersama dan GDPR AWS di Blog Keamanan AWS .

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi Akun AWS kredensil dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan AWS sumber daya. Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail. Untuk informasi tentang penggunaan CloudTrail jejak untuk menangkap AWS aktivitas, lihat Bekerja dengan CloudTrail jejak di AWS CloudTrail Panduan Pengguna.
- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola tingkat lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-3 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi selengkapnya tentang titik akhir FIPS yang tersedia di Standar Pemrosesan Informasi Federal (FIPS) 140-3.

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk saat Anda bekerja dengan Application Auto Scaling atau lainnya Layanan AWS menggunakan konsol, API AWS CLI, atau. AWS SDKs Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log

penagihan atau log diagnostik. Saat Anda memberikan URL ke server eksternal, kami sangat menganjurkan supaya Anda tidak menyertakan informasi kredensial di dalam URL untuk memvalidasi permintaan Anda ke server itu.

Identity and Access Management untuk Application Auto Scaling

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk menggunakan sumber daya Application Auto Scaling. IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Untuk dokumentasi lengkap IAM, lihat Panduan Pengguna IAM.

Kontrol akses

Anda dapat memiliki kredensial yang valid untuk mengautentikasi permintaan Anda, kecuali jika Anda memiliki izin, Anda tidak dapat membuat atau mengakses sumber daya Application Auto Scaling. Misalnya, Anda harus memiliki izin untuk membuat kebijakan penskalaan, mengonfigurasi penskalaan terjadwal, dan sebagainya.

Bagian berikut memberikan rincian tentang bagaimana administrator IAM dapat menggunakan IAM untuk membantu mengamankan AWS sumber daya Anda, dengan mengontrol siapa yang dapat melakukan tindakan Application Auto Scaling API.

Daftar Isi

Cara kerja Application Auto Scaling dengan IAM

> **Note**
>
> Pada bulan Desember 2017, terdapat pembaruan untuk Application Auto Scaling, yang memungkinkan beberapa peran yang ditautkan dengan layanan untuk layanan terpadu Application Auto Scaling. Izin IAM spesifik dan peran yang ditautkan dengan layanan Application Auto Scaling (atau peran layanan untuk penskalaan otomatis Amazon EMR) diperlukan sehingga pengguna dapat mengonfigurasi penskalaan.

Sebelum Anda menggunakan IAM untuk mengelola akses ke Application Auto Scaling, pelajari fitur IAM apa saja yang tersedia untuk digunakan dengan Application Auto Scaling.

Fitur IAM yang dapat Anda gunakan dengan Application Auto Scaling

Fitur IAM	Dukungan Application Auto Scaling
Kebijakan berbasis identitas	Ya
Tindakan kebijakan	Ya
Sumber daya kebijakan	Ya
kunci-kunci persyaratan kebijakan (spesifik layanan)	Ya
Kebijakan berbasis sumber daya	Tidak
ACLs	Tidak
ABAC (tanda dalam kebijakan)	Parsial
Kredensial sementara	Ya
Peran layanan	Ya
Peran terkait layanan	Ya

Untuk mendapatkan tampilan tingkat tinggi tentang bagaimana Application Auto Scaling dan Layanan AWS lainnya bekerja dengan sebagian besar fitur IAM, Layanan AWS lihat bahwa bekerja dengan IAM di Panduan Pengguna IAM.

Kebijakan berbasis identitas Application Auto Scaling

Mendukung kebijakan berbasis identitas: Ya

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Tentukan izin IAM kustom dengan kebijakan terkelola pelanggan dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan secara spesifik apakah tindakan dan sumber daya diizinkan atau ditolak, serta kondisi yang menjadi dasar dikabulkan atau ditolaknya tindakan tersebut. Untuk mempelajari semua elemen yang dapat Anda gunakan dalam kebijakan JSON, lihat Referensi elemen kebijakan JSON IAM dalam Panduan Pengguna IAM.

Contoh kebijakan berbasis identitas untuk Application Auto Scaling

Untuk melihat contoh kebijakan berbasis identitas Application Auto Scaling, lihat Contoh kebijakan berbasis identitas Application Auto Scaling.

Tindakan

Mendukung tindakan kebijakan: Ya

Dalam pernyataan kebijakan IAM, Anda dapat menentukan tindakan API apa pun dari layanan apa pun yang mendukung IAM. Untuk Application Auto Scaling, gunakan awalan berikut dengan nama aksi API:. `application-autoscaling:` Misalnya:`application-autoscaling:RegisterScalableTarget`,`application-autoscaling:PutScalingPolicy`, dan`application-autoscaling:DeregisterScalableTarget`.

Untuk menetapkan beberapa tindakan dalam satu pernyataan, pisahkan dengan koma seperti yang ditunjukkan dalam contoh berikut.

```
"Action": [
      "application-autoscaling:DescribeScalingPolicies",
```

```
      "application-autoscaling:DescribeScalingActivities"
```

Anda dapat menentukan beberapa tindakan menggunakan wildcard (*). Misalnya, untuk menentukan semua tindakan yang dimulai dengan kata `Describe`, sertakan tindakan berikut.

```
"Action": "application-autoscaling:Describe*"
```

Untuk daftar tindakan Application Auto Scaling, lihat Tindakan yang ditentukan oleh AWS Application Auto Scaling di Referensi Otorisasi Layanan.

Sumber daya

Mendukung sumber daya kebijakan: Ya

Dalam pernyataan kebijakan IAM, `Resource` elemen menentukan objek atau objek yang dicakup oleh pernyataan tersebut. Untuk Application Auto Scaling, setiap pernyataan kebijakan IAM berlaku untuk target yang dapat diskalakan yang Anda tentukan menggunakan Nama Sumber Daya Amazon (). ARNs

Format sumber daya ARN untuk target yang dapat diskalakan:

```
arn:aws:application-autoscaling:region:account-id:scalable-target/unique-identifier
```

Misalnya, Anda dapat menunjukkan target terukur tertentu dalam pernyataan Anda menggunakan ARN sebagai berikut. ID unik (1234abcd56ab78cd901ef1234567890ab123) adalah nilai yang ditetapkan oleh Application Auto Scaling ke target yang dapat diskalakan.

```
"Resource": "arn:aws:application-autoscaling:us-east-1:123456789012:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
```

Anda dapat menentukan semua instance milik akun tertentu dengan mengganti pengenal unik dengan wildcard (*) sebagai berikut.

```
"Resource": "arn:aws:application-autoscaling:us-east-1:123456789012:scalable-target/*"
```

Untuk menentukan semua sumber daya, atau jika tindakan API tertentu tidak mendukung ARNs, gunakan wildcard (*) sebagai `Resource` elemen sebagai berikut.

```
"Resource": "*"
```

Untuk informasi selengkapnya, lihat Jenis sumber daya yang ditentukan oleh AWS Application Auto Scaling di Referensi Otorisasi Layanan.

Kunci syarat

Mendukung kunci kondisi kebijakan khusus layanan: Yes

Anda dapat menentukan kondisi dalam kebijakan IAM yang mengontrol akses ke sumber daya Application Auto Scaling. Pernyataan kebijakan hanya efektif jika syaratnya benar.

Application Auto Scaling mendukung kunci kondisi yang ditentukan layanan berikut yang dapat Anda gunakan dalam kebijakan berbasis identitas untuk menentukan siapa yang dapat melakukan tindakan Application Auto Scaling API.

- `application-autoscaling:scalable-dimension`
- `application-autoscaling:service-namespace`

Untuk mempelajari tindakan Application Auto Scaling API yang dapat Anda gunakan dengan kunci kondisi, lihat Tindakan yang ditentukan oleh AWS Application Auto Scaling di Referensi Otorisasi Layanan. Untuk informasi selengkapnya tentang penggunaan tombol kondisi Application Auto Scaling, lihat tombol Kondisi untuk AWS Application Auto Scaling.

Untuk melihat kunci kondisi global yang tersedia untuk semua layanan, lihat kunci konteks kondisi AWS global di Panduan Pengguna IAM.

Kebijakan berbasis sumber daya

Mendukung kebijakan berbasis sumber daya: Tidak

AWS Layanan lain, seperti Amazon Simple Storage Service, mendukung kebijakan izin berbasis sumber daya. Misalnya, Anda dapat melampirkan kebijakan izin ke bucket S3 untuk mengelola izin akses ke bucket tersebut.

Application Auto Scaling tidak mendukung kebijakan berbasis sumber daya.

Daftar Kontrol Akses (ACLs)

Mendukung ACLs: Tidak

Application Auto Scaling tidak mendukung Access Control Lists ()ACLs.

ABAC dengan Application Auto Scaling

Mendukung ABAC (tag dalam kebijakan): Sebagian

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang menentukan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke entitas IAM (pengguna atau peran) dan ke banyak AWS sumber daya. Penandaan ke entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian rancanglah kebijakan ABAC untuk mengizinkan operasi ketika tanda milik principal cocok dengan tanda yang ada di sumber daya yang ingin diakses.

ABAC sangat berguna di lingkungan yang berkembang dengan cepat dan berguna di situasi saat manajemen kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tanda, berikan informasi tentang tanda di elemen kondisi dari kebijakan menggunakan kunci kondisi `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, atau `aws:TagKeys`.

ABAC dimungkinkan untuk sumber daya yang mendukung tag, tetapi tidak semuanya mendukung tag. Tindakan terjadwal dan kebijakan penskalaan tidak mendukung tag, tetapi target yang dapat diskalakan mendukung tag. Untuk informasi selengkapnya, lihat Dukungan penandaan untuk Application Auto Scaling.

Untuk informasi selengkapnya tentang ABAC, lihat Apa itu ABAC? dalam Panduan Pengguna IAM. Untuk melihat tutorial yang menguraikan langkah-langkah pengaturan ABAC, lihat Menggunakan kontrol akses berbasis atribut (ABAC) dalam Panduan Pengguna IAM.

Menggunakan kredensial sementara dengan Application Auto Scaling

Mendukung kredensial sementara: Ya

Kredensi sementara menyediakan akses jangka pendek ke AWS sumber daya dan secara otomatis dibuat saat Anda menggunakan federasi atau beralih peran. AWS merekomendasikan agar Anda secara dinamis menghasilkan kredensi sementara alih-alih menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, lihat Kredensial keamanan sementara di IAM dan Layanan AWS yang berfungsi dengan IAM dalam Panduan Pengguna IAM.

Peran layanan

Mendukung peran layanan: Ya

Jika kluster Amazon EMR Anda menggunakan penskalaan otomatis, fitur ini memungkinkan Application Auto Scaling untuk mengasumsikan peran layanan atas nama Anda. Serupa dengan peran yang ditautkan dengan layanan, peran layanan memungkinkan layanan mengakses sumber daya dalam layanan lain untuk menyelesaikan tindakan atas nama Anda. Peran layanan muncul di akun IAM Anda dan dimiliki oleh akun tersebut. Ini berarti administrator IAM dapat mengubah izin untuk peran ini. Namun, melakukannya mungkin merusak fungsi layanan.

Application Auto Scaling mendukung peran layanan hanya untuk Amazon EMR. Untuk dokumentasi peran layanan EMR, lihat Menggunakan penskalaan otomatis dengan kebijakan kustom untuk grup instans dalam Panduan Manajemen EMR Amazon.

> **Note**
>
> Dengan diperkenalkannya peran terkait layanan, beberapa peran layanan lama tidak lagi diperlukan, misalnya, untuk Amazon ECS dan Spot Fleet.

Peran terkait layanan

Mendukung peran terkait layanan: Ya

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Untuk informasi tentang peran terkait layanan Application Auto Scaling, lihat. Peran yang ditautkan dengan layanan untuk Application Auto Scaling

AWS kebijakan terkelola untuk Application Auto Scaling

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. AWS Kebijakan terkelola dirancang untuk memberikan izin bagi banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan semua pelanggan. AWS Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. Jika AWS memperbarui izin yang ditentukan dalam kebijakan AWS terkelola, pembaruan akan memengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan AWS terkelola saat baru Layanan AWS diluncurkan atau operasi API baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat Kebijakan terkelola AWS dalam Panduan Pengguna IAM.

AWS kebijakan terkelola: WorkSpaces Aplikasi dan CloudWatch

Nama kebijakan: AWSApplicationAutoscalingAppStreamFleetPolicy

Kebijakan ini dilampirkan pada peran terkait layanan yang diberi nama AWSServiceRoleForApplicationAutoScaling_AppStreamFleetuntuk memungkinkan Application Auto Scaling memanggil AppStream Amazon CloudWatch dan dan melakukan penskalaan atas nama Anda.

Detail izin

Kebijakan izin memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada semua sumber daya terkait (“Resource”: “*”):

- Tindakan: `appstream:DescribeFleets`
- Tindakan: `appstream:UpdateFleet`
- Tindakan: `cloudwatch:DescribeAlarms`
- Tindakan: `cloudwatch:PutMetricAlarm`
- Tindakan: `cloudwatch:DeleteAlarms`

AWS kebijakan terkelola: Aurora dan CloudWatch

Nama kebijakan: Kebijakan AWSApplicationPenskalaan Otomatis RDSCluster

Kebijakan ini dilampirkan pada peran terkait layanan yang diberi nama AWSServiceRoleForApplicationAutoScaling_RDSClusteruntuk memungkinkan Application Auto Scaling memanggil Aurora CloudWatch dan dan melakukan penskalaan atas nama Anda.

Detail izin

Kebijakan izin memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada semua sumber daya terkait (“Resource”: “*”):

- Tindakan: `rds:AddTagsToResource`
- Tindakan: `rds:CreateDBInstance`
- Tindakan: `rds:DeleteDBInstance`
- Tindakan: `rds:DescribeDBClusters`
- Tindakan: `rds:DescribeDBInstance`
- Tindakan: `cloudwatch:DescribeAlarms`
- Tindakan: `cloudwatch:PutMetricAlarm`
- Tindakan: `cloudwatch:DeleteAlarms`

AWS kebijakan terkelola: Amazon Comprehend dan CloudWatch

Nama kebijakan: AWSApplicationAutoscalingComprehendEndpointPolicy

Kebijakan ini dilampirkan pada peran terkait layanan yang diberi nama AWSServiceRoleForApplicationAutoScaling_ComprehendEndpointuntuk memungkinkan Application Auto Scaling memanggil Amazon Comprehend dan serta melakukan penskalaan atas nama Anda. CloudWatch

Detail izin

Kebijakan izin memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada semua sumber daya terkait (“Resource”: “*”):

- Tindakan: `comprehend:UpdateEndpoint`
- Tindakan: `comprehend:DescribeEndpoint`
- Tindakan: `cloudwatch:DescribeAlarms`
- Tindakan: `cloudwatch:PutMetricAlarm`
- Tindakan: `cloudwatch:DeleteAlarms`

AWS kebijakan terkelola: DynamoDB dan CloudWatch

Nama kebijakan: AWSApplicationAutoscalingDynamoDBTableKebijakan

Kebijakan ini dilampirkan pada peran terkait layanan yang diberi nama AWSServiceRoleForApplicationAutoScaling_DynamoDBTableuntuk memungkinkan Application Auto Scaling memanggil DBand CloudWatch Dynamo dan melakukan penskalaan atas nama Anda.

Detail izin

Kebijakan izin memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada semua sumber daya terkait (“Resource”: “*”):

- Tindakan: `dynamodb:DescribeTable`
- Tindakan: `dynamodb:UpdateTable`
- Tindakan: `cloudwatch:DescribeAlarms`
- Tindakan: `cloudwatch:PutMetricAlarm`
- Tindakan: `cloudwatch:DeleteAlarms`

AWS kebijakan terkelola: Amazon ECS dan CloudWatch

Nama kebijakan: Kebijakan AWSApplicationPenskalaan Otomatis ECSService

Kebijakan ini dilampirkan pada peran terkait layanan yang diberi nama AWSServiceRoleForApplicationAutoScaling_ECSServiceuntuk memungkinkan Application Auto Scaling memanggil Amazon ECS CloudWatch dan serta melakukan penskalaan atas nama Anda.

Detail izin

Kebijakan izin memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada semua sumber daya terkait (“Resource”: “*”):

- Tindakan: `ecs:DescribeServices`
- Tindakan: `ecs:UpdateService`
- Tindakan: `cloudwatch:PutMetricAlarm`
- Tindakan: `cloudwatch:DescribeAlarms`
- Tindakan: `cloudwatch:GetMetricData`
- Tindakan: `cloudwatch:DeleteAlarms`

AWS kebijakan terkelola: ElastiCache dan CloudWatch

Nama kebijakan: AWSApplicationAutoscalingElastiCacheRGPolicy

Kebijakan ini dilampirkan pada peran terkait layanan yang diberi nama AWSServiceRoleForApplicationAutoScaling_ElastiCacheRGuntuk memungkinkan Application Auto

Scaling ElastiCache memanggil CloudWatch dan dan melakukan penskalaan atas nama Anda. Peran terkait layanan ini dapat digunakan untuk ElastiCache Memcached, Redis OSS, dan Valkey.

Detail izin

Kebijakan izin memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- Tindakan: `elasticache:DescribeReplicationGroups` pada semua sumber daya
- Tindakan: `elasticache:ModifyReplicationGroupShardConfiguration` pada semua sumber daya
- Tindakan: `elasticache:IncreaseReplicaCount` pada semua sumber daya
- Tindakan: `elasticache:DecreaseReplicaCount` pada semua sumber daya
- Tindakan: `elasticache:DescribeCacheClusters` pada semua sumber daya
- Tindakan: `elasticache:DescribeCacheParameters` pada semua sumber daya
- Tindakan: `elasticache:ModifyCacheCluster` pada semua sumber daya
- Tindakan: `cloudwatch:DescribeAlarms` pada sumber daya `arn:aws:cloudwatch:*:*:alarm:*`
- Tindakan: `cloudwatch:PutMetricAlarm` pada sumber daya `arn:aws:cloudwatch:*:*:alarm:TargetTracking*`
- Tindakan: `cloudwatch:DeleteAlarms` pada sumber daya `arn:aws:cloudwatch:*:*:alarm:TargetTracking*`

AWS kebijakan terkelola: Amazon Keyspaces dan CloudWatch

Nama kebijakan: AWSApplicationAutoscalingCassandraTablePolicy

Kebijakan ini dilampirkan ke peran terkait layanan yang diberi nama AWSServiceRoleForApplicationAutoScaling_CassandraTableuntuk memungkinkan Application Auto Scaling memanggil Amazon Keyspaces CloudWatch dan melakukan penskalaan atas nama Anda.

Detail izin

Kebijakan izin memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- Tindakan: `cassandra:Select` pada sumber daya berikut:

  - `arn:*:cassandra:*:*:/keyspace/system/table/*`
  - `arn:*:cassandra:*:*:/keyspace/system_schema/table/*`
  - `arn:*:cassandra:*:*:/keyspace/system_schema_mcs/table/*`
- Tindakan: `cassandra:Alter` pada semua sumber daya
- Tindakan: `cloudwatch:DescribeAlarms` pada semua sumber daya
- Tindakan: `cloudwatch:PutMetricAlarm` pada semua sumber daya
- Tindakan: `cloudwatch:DeleteAlarms` pada semua sumber daya

AWS kebijakan terkelola: Lambda dan CloudWatch

Nama kebijakan: AWSApplicationAutoscalingLambdaConcurrencyPolicy

Kebijakan ini dilampirkan pada peran terkait layanan yang diberi nama AWSServiceRoleForApplicationAutoScaling_LambdaConcurrencyuntuk memungkinkan Application Auto Scaling memanggil Lambda CloudWatch dan serta melakukan penskalaan atas nama Anda.

Detail izin

Kebijakan izin memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada semua sumber daya terkait (“Resource”: “*”):

- Tindakan: `lambda:PutProvisionedConcurrencyConfig`
- Tindakan: `lambda:GetProvisionedConcurrencyConfig`
- Tindakan: `lambda:DeleteProvisionedConcurrencyConfig`
- Tindakan: `cloudwatch:DescribeAlarms`
- Tindakan: `cloudwatch:PutMetricAlarm`
- Tindakan: `cloudwatch:DeleteAlarms`

AWS kebijakan terkelola: Amazon MSK dan CloudWatch

Nama kebijakan: AWSApplicationAutoscalingKafkaClusterPolicy

Kebijakan ini dilampirkan pada peran terkait layanan yang diberi nama AWSServiceRoleForApplicationAutoScaling_KafkaClusteruntuk memungkinkan Application Auto Scaling memanggil Amazon MSK CloudWatch dan serta melakukan penskalaan atas nama Anda.

Detail izin

Kebijakan izin memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada semua sumber daya terkait (“Resource”: “*”):

- Tindakan: `kafka:DescribeCluster`
- Tindakan: `kafka:DescribeClusterOperation`
- Tindakan: `kafka:UpdateBrokerStorage`
- Tindakan: `cloudwatch:DescribeAlarms`
- Tindakan: `cloudwatch:PutMetricAlarm`
- Tindakan: `cloudwatch:DeleteAlarms`

AWS kebijakan terkelola: Neptunus dan CloudWatch

Nama kebijakan: AWSApplicationAutoscalingNeptuneClusterPolicy

Kebijakan ini dilampirkan ke peran terkait layanan yang diberi nama AWSServiceRoleForApplicationAutoScaling_NeptuneClusteruntuk memungkinkan Application Auto Scaling memanggil CloudWatch Neptunus dan melakukan penskalaan atas nama Anda.

Detail izin

Kebijakan izin memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- Tindakan: `rds:ListTagsForResource` pada semua sumber daya
- Tindakan: `rds:DescribeDBInstances` pada semua sumber daya
- Tindakan: `rds:DescribeDBClusters` pada semua sumber daya
- Tindakan: `rds:DescribeDBClusterParameters` pada semua sumber daya
- Tindakan: `cloudwatch:DescribeAlarms` pada semua sumber daya
- Tindakan: `rds:AddTagsToResource` pada sumber daya dengan awalan pembaca berskala otomatis di mesin database Amazon Neptunus () `"Condition":{"StringEquals":{"rds:DatabaseEngine":"neptune"}`
- Tindakan: `rds:CreateDBInstance` pada sumber daya dengan awalan pembaca berskala otomatis di semua cluster DB () di mesin database Amazon Neptunus () `"Resource":"arn:*:rds:*:*:db:autoscaled-reader*",`

`"arn:aws:rds:*:*:cluster:*" "Condition":{"StringEquals":{"rds:DatabaseEngine":"neptune"}`

- Tindakan: `rds:DeleteDBInstance` pada sumber daya `arn:aws:rds:*:*:db:autoscaled-reader*`
- Tindakan: `cloudwatch:PutMetricAlarm` pada sumber daya `arn:aws:cloudwatch:*:*:alarm:TargetTracking*`
- Tindakan: `cloudwatch:DeleteAlarms` pada sumber daya `arn:aws:cloudwatch:*:*:alarm:TargetTracking*`

AWS kebijakan terkelola: SageMaker AI dan CloudWatch

Nama kebijakan: AWSApplicationAutoscalingSageMakerEndpointPolicy

Kebijakan ini dilampirkan pada peran terkait layanan yang diberi nama AWSServiceRoleForApplicationAutoScaling_SageMakerEndpointuntuk memungkinkan Application Auto Scaling SageMaker memanggil AI CloudWatch dan dan melakukan penskalaan atas nama Anda.

Detail izin

Kebijakan izin memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- Tindakan: `sagemaker:DescribeEndpoint` pada semua sumber daya
- Tindakan: `sagemaker:DescribeEndpointConfig` pada semua sumber daya
- Tindakan: `sagemaker:DescribeInferenceComponent` pada semua sumber daya
- Tindakan: `sagemaker:UpdateEndpointWeightsAndCapacities` pada semua sumber daya
- Tindakan: `sagemaker:UpdateInferenceComponentRuntimeConfig` pada semua sumber daya
- Tindakan: `cloudwatch:DescribeAlarms` pada semua sumber daya
- Tindakan: `cloudwatch:GetMetricData` pada semua sumber daya
- Tindakan: `cloudwatch:PutMetricAlarm` pada sumber daya `arn:aws:cloudwatch:*:*:alarm:TargetTracking*`
- Tindakan: `cloudwatch:DeleteAlarms` pada sumber daya `arn:aws:cloudwatch:*:*:alarm:TargetTracking*`

AWS kebijakan terkelola: Armada Spot EC2 dan CloudWatch

Nama kebijakan: AWSApplicationAutoscaling EC2 SpotFleetRequestPolicy

Kebijakan ini dilampirkan pada peran terkait layanan bernama AWSServiceRoleForApplicationAutoScaling_EC2 SpotFleetRequest untuk memungkinkan Application Auto Scaling memanggil Amazon EC2 CloudWatch dan melakukan penskalaan atas nama Anda.

Detail izin

Kebijakan izin memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada semua sumber daya terkait (“Resource”: “*”):

- Tindakan: `ec2:DescribeSpotFleetRequests`
- Tindakan: `ec2:ModifySpotFleetRequest`
- Tindakan: `cloudwatch:DescribeAlarms`
- Tindakan: `cloudwatch:PutMetricAlarm`
- Tindakan: `cloudwatch:DeleteAlarms`

AWS kebijakan terkelola: WorkSpaces dan CloudWatch

Nama kebijakan: AWSApplicationAutoscalingWorkSpacesPoolPolicy

Kebijakan ini dilampirkan pada peran terkait layanan yang diberi nama AWSServiceRoleForApplicationAutoScaling_WorkSpacesPooluntuk memungkinkan Application Auto Scaling WorkSpaces memanggil CloudWatch dan dan melakukan penskalaan atas nama Anda.

Detail izin

Kebijakan izin memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- Tindakan: `workspaces:DescribeWorkspacesPools` pada semua sumber daya dari akun yang sama dengan SLR
- Tindakan: `workspaces:UpdateWorkspacesPool` pada semua sumber daya dari akun yang sama dengan SLR

- Tindakan: `cloudwatch:DescribeAlarms` pada semua alarm dari akun yang sama dengan SLR
- Tindakan: `cloudwatch:PutMetricAlarm` pada semua alarm dari akun yang sama dengan SLR, di mana nama alarm dimulai dengan TargetTracking
- Tindakan: `cloudwatch:DeleteAlarms` pada semua alarm dari akun yang sama dengan SLR, di mana nama alarm dimulai dengan TargetTracking

AWS kebijakan terkelola: sumber daya khusus dan CloudWatch

Nama kebijakan: AWSApplicationAutoScalingCustomResourcePolicy

Kebijakan ini dilampirkan ke peran terkait layanan yang diberi nama AWSServiceRoleForApplicationAutoScaling_CustomResourceuntuk memungkinkan Application Auto Scaling memanggil sumber daya kustom Anda yang tersedia melalui API Gateway CloudWatch dan dan melakukan penskalaan atas nama Anda.

Detail izin

Kebijakan izin memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada semua sumber daya terkait (“Resource”: “*”):

- Tindakan: `execute-api:Invoke`
- Tindakan: `cloudwatch:DescribeAlarms`
- Tindakan: `cloudwatch:PutMetricAlarm`
- Tindakan: `cloudwatch:DeleteAlarms`

Pembaruan Application Auto Scaling ke AWS kebijakan terkelola

Lihat detail tentang pembaruan kebijakan AWS terkelola untuk Application Auto Scaling sejak layanan ini mulai melacak perubahan ini. Untuk peringatan otomatis tentang perubahan pada halaman ini, berlangganan umpan RSS di halaman riwayat Dokumen Penskalaan Otomatis Aplikasi.

Ubah	Deskripsi	Date
AWSApplicationAutoscalingElastiCacheRGPolicy— Perbarui kebijakan yang ada	Menambahkan izin untuk memanggil tindakan ElastiCache `ModifyCacheCluster` API untuk mendukung	April 10, 2025

Ubah	Deskripsi	Date
	penskalaan otomatis Memcached.	
AWSApplicationKebijakan Penskalaan Otomatis - Memperbarui ECSService kebijakan yang ada	Menambahkan izin untuk memanggil tindakan CloudWatch `GetMetricData` API guna mendukung penskalaan prediktif.	November 21, 2024
AWSApplicationAutoscalingWorkSpacesPoolPolicy – Kebijakan baru	Menambahkan kebijakan terkelola untuk Amazon WorkSpaces. Kebijakan ini dilampirkan pada peran terkait layanan yang memungkinkan Application Auto Scaling untuk WorkSpaces memanggil CloudWatch dan dan melakukan penskalaan atas nama Anda.	24 Juni 2024

Ubah	Deskripsi	Date
AWSApplicationAutoscalingSageMakerEndpointPolicy – Pembaruan ke kebijakan yang ada	Menambahkan izin untuk memanggil tindakan SageMaker AI `DescribeInferenceComponent` dan `UpdateInferenceComponentRuntimeConfig` API untuk mendukung kompatibilitas penskalaan otomatis sumber daya SageMaker AI untuk integrasi yang akan datang. Kebijakan ini juga sekarang membatasi tindakan CloudWatch `PutMetricAlarm` dan `DeleteAlarms` API ke CloudWatch alarm yang digunakan dengan kebijakan penskalaan pelacakan target.	13 November 2023
AWSApplicationAutoscalingNeptuneClusterPolicy – Kebijakan baru	Menambahkan kebijakan terkelola untuk Neptunus. Kebijakan ini dilampirkan pada peran terkait layanan yang memungkinkan Application Auto Scaling memanggil CloudWatch Neptunus dan melakukan penskalaan atas nama Anda.	Oktober 6, 2021

Ubah	Deskripsi	Date
AWSApplicationKebijakan Penskalaan Otomatis — RDSCluster Kebijakan baru	Menambahkan kebijakan terkelola untuk ElastiCache. Kebijakan ini dilampirkan pada peran terkait layanan yang memungkinkan Application Auto Scaling untuk ElastiCac he memanggil CloudWatch dan dan melakukan penskalaa n atas nama Anda.	19 Agustus 2021
Application Auto Scaling mulai melacak perubahan	Application Auto Scaling mulai melacak perubahan untuk kebijakan yang AWS dikelola.	19 Agustus 2021

Peran yang ditautkan dengan layanan untuk Application Auto Scaling

Application Auto Scaling menggunakan peran terkait layanan untuk izin yang diperlukan untuk memanggil layanan lain AWS atas nama Anda. Peran terkait layanan adalah jenis peran unik AWS Identity and Access Management (IAM) yang ditautkan langsung ke layanan. AWS Peran terkait layanan menyediakan cara aman untuk mendelegasikan izin ke AWS layanan karena hanya layanan tertaut yang dapat mengambil peran terkait layanan.

Untuk layanan yang terintegrasi dengan Application Auto Scaling, Application Auto Scaling membuat peran terkait layanan untuk Anda. Ada satu peran terkait layanan untuk setiap layanan. Setiap peran yang ditautkan dengan layanan memercayai prinsipal layanan yang ditentukan untuk mengasumsikannya. Untuk informasi selengkapnya, lihat Referensi ARN peran yang ditautkan dengan layanan.

Application Auto Scaling mencakup semua izin yang diperlukan untuk setiap peran terkait layanan. Izin terkelola ini dibuat dan dikelola oleh Application Auto Scaling, dan mereka menentukan tindakan yang diizinkan untuk setiap jenis sumber daya. Untuk detail tentang izin yang diberikan setiap peran, lihat. AWS kebijakan terkelola untuk Application Auto Scaling

Daftar Isi

- Buat peran yang ditautkan dengan layanan (otomatis)
- Buat peran yang ditautkan dengan layanan (manual)
- Edit peran yang ditautkan dengan layanan
- Hapus peran yang ditautkan dengan layanan
- Wilayah yang Didukung untuk peran terkait layanan Application Auto Scaling
- Referensi ARN peran yang ditautkan dengan layanan

Izin yang diperlukan untuk membuat peran yang ditautkan dengan layanan

Application Auto Scaling memerlukan izin untuk membuat peran terkait layanan saat pertama kali setiap pengguna dalam Akun AWS panggilan `RegisterScalableTarget` Anda untuk layanan tertentu. Application Auto Scaling membuat peran tertaut layanan untuk layanan target di akun Anda, jika peran tersebut belum ada. Peran yang ditautkan dengan layanan memberikan izin untuk Application Auto Scaling sehingga dapat menghubungi layanan target atas nama Anda.

Agar pembuatan peran otomatis berhasil, pengguna harus memiliki izin untuk tindakan `iam:CreateServiceLinkedRole` nyata.

```
"Action": "iam:CreateServiceLinkedRole"
```

Berikut ini adalah kebijakan berbasis identitas yang memberikan izin untuk membuat peran terkait layanan untuk Armada Spot. Anda dapat menentukan peran tertaut layanan di bidang kebijakan `Resource` sebagai ARN, dan prinsip layanan untuk peran tertaut layanan sebagai ketentuan, seperti yang ditunjukkan. Untuk ARN untuk setiap layanan, lihat. Referensi ARN peran yang ditautkan dengan layanan

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-
service-role/ec2.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest",
```

```
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName":"ec2.application-
autoscaling.amazonaws.com"
                }
            }
        }
    ]
}
```

> **Note**
>
> Kunci kondisi IAM `iam:AWSServiceName` menentukan layanan utama yang peran dilampirkan, yang ditunjukkan dalam kebijakan contoh ini sebagai *`ec2.application-autoscaling`*`.amazonaws.com`. Jangan mencoba menebak prinsip utama layanan. Untuk melihat prinsip layanan untuk suatu layanan, lihatLayanan AWS yang dapat Anda gunakan dengan Application Auto Scaling.

Buat peran yang ditautkan dengan layanan (otomatis)

Anda tidak perlu membuat peran yang ditautkan dengan layanan secara manual. Application Auto Scaling membuat peran yang ditautkan dengan layanan sesuai untuk Anda saat Anda menelepon `RegisterScalableTarget`. Misalnya, jika Anda menyiapkan penskalaan otomatis untuk layanan Amazon ECS, Application Auto Scaling membuat peran `AWSServiceRoleForApplicationAutoScaling_ECSService`.

Buat peran yang ditautkan dengan layanan (manual)

Untuk membuat peran terkait layanan, Anda dapat menggunakan konsol IAM AWS CLI, atau IAM API. Untuk informasi selengkapnya, lihat Membuat peran terkait layanan di Panduan Pengguna IAM.

Untuk membuat peran terkait layanan (AWS CLI)

Gunakan create-service-linked-roleperintah berikut untuk membuat peran terkait layanan Application Auto Scaling. Dalam permintaan, tentukan nama layanan “awalan”.

Untuk menemukan awalan nama layanan, lihat informasi tentang prinsip layanan untuk peran terkait layanan untuk setiap layanan di bagian. Layanan AWS yang dapat Anda gunakan dengan

Application Auto Scaling Nama layanan dan kepala layanan berbagi awalan yang sama. Misalnya, untuk membuat peran AWS Lambda terkait layanan, gunakan. `lambda.application-autoscaling.amazonaws.com`

```
aws iam create-service-linked-role --aws-service-name prefix.application-autoscaling.amazonaws.com
```

Edit peran yang ditautkan dengan layanan

Dengan peran yang ditautkan layanan yang dibuat oleh Application Auto Scaling, Anda dapat mengedit deskripsi mereka saja. Untuk informasi selengkapnya, lihat Mengedit deskripsi peran terkait layanan di Panduan Pengguna IAM.

Hapus peran yang ditautkan dengan layanan

Jika Anda tidak lagi menggunakan Application Auto Scaling dengan layanan yang didukung, kami sarankan Anda menghapus peran yang ditautkan dengan layanan.

Anda dapat menghapus peran terhubung layanan hanya setelah pertama kali menghapus sumber daya AWS terkait. Hal ini melindungi Anda agar tidak merusak izin Application Auto Scaling ke sumber daya Anda secara tidak sengaja. Untuk informasi lebih lanjut, lihat dokumentasi sumber daya yang dapat diskalakan. Misalnya, untuk menghapus layanan Amazon ECS, lihat Menghapus layanan Amazon ECS di Panduan Pengembang Layanan Amazon Elastic Container.

Anda dapat menggunakan IAM untuk menghapus peran yang ditautkan dengan layanan. Untuk informasi selengkapnya, lihat Menghapus peran terkait layanan di Panduan Pengguna IAM.

Setelah Anda menghapus peran yang ditautkan dengan layanan, Application Auto Scaling akan membuat peran itu lagi saat Anda memanggil `RegisterScalableTarget`.

Wilayah yang Didukung untuk peran terkait layanan Application Auto Scaling

Application Auto Scaling mendukung penggunaan peran terkait layanan di semua AWS Wilayah tempat layanan tersedia.

Referensi ARN peran yang ditautkan dengan layanan

Tabel berikut mencantumkan Nama Sumber Daya Amazon (ARN) dari peran terkait layanan untuk masing-masing Layanan AWS yang bekerja dengan Application Auto Scaling.

Layanan	ARN
AppStream 2.0	arn:aws:iam:: *012345678910* :role/aws-service-role/appstream.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_AppStreamFleet
Aurora	arn:aws:iam:: *012345678910* :role/aws-service-role/rds.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_RDSCluster
Comprehend	arn:aws:iam:: *012345678910* :role/aws-service-role/comprehend.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_ComprehendEndpoint
DynamoDB	arn:aws:iam:: *012345678910* :role/aws-service-role/dynamodb.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_DynamoDBTable
ECS	arn:aws:iam:: *012345678910* :role/aws-service-role/ecs.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_ECSService
ElastiCache	arn:aws:iam:: *012345678910* :role/aws-service-role/elasticache.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG
Keyspaces	arn:aws:iam:: *012345678910* :role/aws-service-role/cassandra.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_CassandraTable
Lambda	arn:aws:iam:: *012345678910* :role/aws-service-role/lambda.application-autoscaling.amazonaws.com/AWSS

Layanan	ARN
	erviceRoleForApplicationAutoScaling_LambdaCon currency
MSK	arn:aws:iam:: *012345678910* :role/aws-service-role/ kafka.application-autoscaling.amazonaws.com/AWSSe rviceRoleForApplicationAutoScaling_KafkaCluster
Neptune	arn:aws:iam:: *012345678910* :role/aws-service-role/ neptune.application-autoscaling.amazonaws.com/ AWSServiceRoleForApplicationAutoScaling_NeptuneC luster
SageMaker AI	arn:aws:iam:: *012345678910* :role/aws-service-role/ sagemaker.application-autoscaling.amazonaws.com/ AWSServiceRoleForApplicationAutoScaling_SageMa kerEndpoint
Armada Spot	arn:aws:iam:: *012345678910* :role/aws-service-role/ ec2.application-autoscaling.amazonaws.com/AWSServ iceRoleForApplicationAutoScaling_EC2SpotFleet Request
WorkSpaces	arn:aws:iam:: *012345678910* :role/aws-service-role/ workspaces.application-autoscaling.amazonaws.com/ AWSServiceRoleForApplicationAutoScaling_WorkS pacesPool
Sumber daya khusus	arn:aws:iam:: *012345678910* :role/aws-service-role/cust om-resource.application-autoscaling.amazonaws.com/ AWSServiceRoleForApplicationAutoScaling_CustomRes ource

Note

Anda dapat menentukan ARN peran yang ditautkan layanan untuk `RoleARN` properti [AWS::ApplicationAutoScaling::ScalableTarget](#)sumber daya di templat CloudFormation

tumpukan, meskipun peran terkait layanan yang ditentukan belum ada. Application Auto Scaling secara otomatis menciptakan peran untuk Anda.

Contoh kebijakan berbasis identitas Application Auto Scaling

Secara default, pengguna baru di Anda tidak Akun AWS memiliki izin untuk melakukan apa pun. Administrator IAM harus membuat dan menetapkan kebijakan IAM yang memberikan izin identitas IAM (seperti pengguna atau peran) untuk melakukan tindakan Application Auto Scaling API.

Untuk mempelajari cara membuat kebijakan IAM dengan menggunakan contoh dokumen kebijakan JSON berikut, lihat Membuat kebijakan di tab JSON dalam Panduan Pengguna IAM.

Daftar Isi

Izin yang diperlukan untuk tindakan Application Auto Scaling API

Kebijakan berikut memberikan izin untuk kasus penggunaan umum saat memanggil Application Auto Scaling API. Lihat bagian ini saat menulis kebijakan berbasis identitas. Setiap kebijakan memberikan izin untuk semua atau beberapa tindakan Application Auto Scaling API. Anda juga perlu memastikan bahwa pengguna akhir memiliki izin untuk layanan target dan CloudWatch (lihat bagian selanjutnya untuk detailnya).

Kebijakan berbasis identitas berikut memberikan izin untuk semua tindakan Application Auto Scaling API.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
            "application-autoscaling:*"
        ],
        "Resource": "*"
    }
  ]
}
```

Kebijakan berbasis identitas berikut memberikan izin untuk semua tindakan Application Auto Scaling API yang diperlukan untuk mengonfigurasi kebijakan penskalaan dan bukan tindakan terjadwal.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "application-autoscaling:RegisterScalableTarget",
              "application-autoscaling:DescribeScalableTargets",
              "application-autoscaling:DeregisterScalableTarget",
              "application-autoscaling:PutScalingPolicy",
              "application-autoscaling:DescribeScalingPolicies",
              "application-autoscaling:DescribeScalingActivities",
              "application-autoscaling:DeleteScalingPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

Kebijakan berbasis identitas berikut memberikan izin untuk semua tindakan Application Auto Scaling API yang diperlukan untuk mengonfigurasi tindakan terjadwal dan bukan kebijakan penskalaan.

JSON

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

```
        {
            "Effect": "Allow",
            "Action": [
              "application-autoscaling:RegisterScalableTarget",
              "application-autoscaling:DescribeScalableTargets",
              "application-autoscaling:DeregisterScalableTarget",
              "application-autoscaling:PutScheduledAction",
              "application-autoscaling:DescribeScheduledActions",
              "application-autoscaling:DescribeScalingActivities",
              "application-autoscaling:DeleteScheduledAction"
            ],
            "Resource": "*"
        }
    ]
}
```

Izin diperlukan untuk tindakan API pada layanan target dan CloudWatch

Agar berhasil mengonfigurasi dan menggunakan Application Auto Scaling dengan layanan target, pengguna akhir harus diberikan izin untuk Amazon CloudWatch dan untuk setiap layanan target yang akan mereka konfigurasi penskalaannya. Gunakan kebijakan berikut untuk memberikan izin minimum yang diperlukan untuk bekerja dengan layanan target dan CloudWatch.

Daftar Isi

- Armada Spot (Amazon EC2)
- Sumber daya khusus

AppStream 2.0 armada

Kebijakan berbasis identitas berikut memberikan izin untuk semua tindakan AppStream 2.0 dan CloudWatch API yang diperlukan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "appstream:DescribeFleets",
              "appstream:UpdateFleet",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Replika Aurora

Kebijakan berbasis identitas berikut memberikan izin untuk semua tindakan Aurora CloudWatch dan API yang diperlukan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
              "rds:AddTagsToResource",
              "rds:CreateDBInstance",
              "rds:DeleteDBInstance",
              "rds:DescribeDBClusters",
              "rds:DescribeDBInstances",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Klasifikasi dokumen dan titik akhir pengenal entitas Amazon Comprehend

Kebijakan berbasis identitas berikut memberikan izin ke semua Amazon Comprehend dan tindakan API yang diperlukan. CloudWatch

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "comprehend:UpdateEndpoint",
              "comprehend:DescribeEndpoint",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Tabel DynamoDB dan indeks sekunder global

Kebijakan berbasis identitas berikut memberikan izin untuk semua tindakan DynamoDB dan API yang diperlukan. CloudWatch

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "dynamodb:DescribeTable",
              "dynamodb:UpdateTable",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Layanan ECS

Kebijakan berbasis identitas berikut memberikan izin untuk semua tindakan ECS dan CloudWatch API yang diperlukan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "ecs:DescribeServices",
              "ecs:UpdateService",
              "cloudwatch:DescribeAlarms",
```

```
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

ElastiCache kelompok replikasi

Kebijakan berbasis identitas berikut memberikan izin untuk semua ElastiCache dan tindakan CloudWatch API yang diperlukan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "elasticache:ModifyReplicationGroupShardConfiguration",
              "elasticache:IncreaseReplicaCount",
              "elasticache:DecreaseReplicaCount",
              "elasticache:DescribeReplicationGroups",
              "elasticache:DescribeCacheClusters",
              "elasticache:DescribeCacheParameters",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Kluster Amazon EMR

Kebijakan berbasis identitas berikut memberikan izin ke semua EMR CloudWatch Amazon dan tindakan API yang diperlukan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "elasticmapreduce:ModifyInstanceGroups",
              "elasticmapreduce:ListInstanceGroups",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Tabel Amazon Keyspaces

Kebijakan berbasis identitas berikut memberikan izin ke semua Keyspaces CloudWatch Amazon dan tindakan API yang diperlukan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "cassandra:Select",
              "cassandra:Alter",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

Fungsi Lambda

Kebijakan berbasis identitas berikut memberikan izin untuk semua tindakan Lambda CloudWatch dan API yang diperlukan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "lambda:PutProvisionedConcurrencyConfig",
              "lambda:GetProvisionedConcurrencyConfig",
              "lambda:DeleteProvisionedConcurrencyConfig",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Penyimpanan broker Amazon Managed Streaming untuk Apache Kafka (MSK)

Kebijakan berbasis identitas berikut memberikan izin ke semua tindakan MSK CloudWatch dan API Amazon yang diperlukan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
              "kafka:DescribeCluster",
              "kafka:DescribeClusterOperation",
              "kafka:UpdateBrokerStorage",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Cluster Neptunus

Kebijakan berbasis identitas berikut memberikan izin untuk semua tindakan CloudWatch Neptunus dan API yang diperlukan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "rds:AddTagsToResource",
              "rds:CreateDBInstance",
              "rds:DescribeDBInstances",
              "rds:DescribeDBClusters",
              "rds:DescribeDBClusterParameters",
              "rds:DeleteDBInstance",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

SageMaker Titik akhir AI

Kebijakan berbasis identitas berikut memberikan izin untuk semua tindakan SageMaker AI dan CloudWatch API yang diperlukan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "sagemaker:DescribeEndpoint",
              "sagemaker:DescribeEndpointConfig",
              "sagemaker:DescribeInferenceComponent",
              "sagemaker:UpdateEndpointWeightsAndCapacities",
              "sagemaker:UpdateInferenceComponentRuntimeConfig",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Armada Spot (Amazon EC2)

Kebijakan berbasis identitas berikut memberikan izin untuk semua tindakan Armada Spot dan CloudWatch API yang diperlukan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "ec2:DescribeSpotFleetRequests",
                "ec2:ModifySpotFleetRequest",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Sumber daya khusus

Kebijakan berbasis identitas berikut memberikan izin untuk tindakan eksekusi API Gateway API. Kebijakan ini juga memberikan izin untuk semua CloudWatch tindakan yang diperlukan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "execute-api:Invoke",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Izin untuk bekerja di Konsol Manajemen AWS

Tidak ada konsol Application Auto Scaling mandiri. Sebagian besar layanan yang terintegrasi dengan Application Auto Scaling memiliki fitur yang dikhususkan untuk membantu Anda mengonfigurasi penskalaan dengan konsol mereka.

Dalam kebanyakan kasus, setiap layanan menyediakan kebijakan IAM AWS terkelola (yang telah ditentukan sebelumnya) yang menentukan akses ke konsol mereka, yang mencakup izin untuk tindakan Application Auto Scaling API. Untuk informasi lebih lanjut, lihat dokumentasi untuk layanan yang konsolnya ingin Anda gunakan.

Anda juga dapat membuat kebijakan IAM kustom Anda sendiri untuk memberi pengguna izin halus untuk melihat dan bekerja dengan tindakan Application Auto Scaling API tertentu di. Konsol Manajemen AWS Anda dapat menggunakan contoh kebijakan di bagian sebelumnya; namun, kebijakan tersebut dirancang untuk permintaan yang dibuat dengan AWS CLI atau SDK. Konsol tersebut menggunakan tindakan-tindakan API tambahan untuk fitur-fiturnya, sehingga kebijakan-kebijakan ini mungkin tidak berjalan sesuai yang diharapkan. Misalnya, untuk mengonfigurasi penskalaan langkah, pengguna mungkin memerlukan izin tambahan untuk membuat dan mengelola CloudWatch alarm.

> **Tip**
>
> Untuk membantu Anda mengetahui tindakan API mana yang dibutuhkan untuk melakukan tugas-tugas dalam konsol, Anda dapat menggunakan layanan seperti AWS CloudTrail. Untuk informasi selengkapnya, silakan lihat Panduan Pengguna AWS CloudTrail.

Kebijakan berbasis identitas berikut memberikan izin untuk mengonfigurasi kebijakan penskalaan untuk Armada Spot. Selain izin IAM untuk Spot Fleet, pengguna konsol yang mengakses pengaturan penskalaan armada dari konsol Amazon EC2 harus memiliki izin yang sesuai untuk layanan yang mendukung penskalaan dinamis.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:*",
                "ec2:DescribeSpotFleetRequests",
                "ec2:ModifySpotFleetRequest",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarmHistory",
```

```
                "cloudwatch:DescribeAlarms",
                "cloudwatch:DescribeAlarmsForMetric",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DisableAlarmActions",
                "cloudwatch:EnableAlarmActions",
                "sns:CreateTopic",
                "sns:Subscribe",
                "sns:Get*",
                "sns:List*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-
service-role/ec2.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName":"ec2.application-
autoscaling.amazonaws.com"
                }
            }
        }
    ]
}
```

Kebijakan ini memungkinkan pengguna konsol untuk melihat dan memodifikasi kebijakan penskalaan di konsol Amazon EC2, serta membuat dan CloudWatch mengelola alarm di konsol. CloudWatch

Anda dapat menyesuaikan tindakan API untuk membatasi akses pengguna. Misalnya, mengganti `application-autoscaling:*` dengan `application-autoscaling:Describe*` berarti pengguna memiliki akses hanya-baca.

Anda juga dapat menyesuaikan CloudWatch izin yang diperlukan untuk membatasi akses pengguna ke CloudWatch fitur. Untuk informasi selengkapnya, lihat Izin yang diperlukan untuk CloudWatch konsol di Panduan CloudWatch Pengguna Amazon.

Pemecahan masalah akses Application Auto Scaling

Jika Anda menemukan `AccessDeniedException` atau kesulitan serupa saat bekerja dengan Application Auto Scaling, konsultasikan informasi di bagian ini.

Saya tidak diotorisasi untuk melakukan tindakan dalam Application Auto Scaling

Jika Anda menerima `AccessDeniedException` saat memanggil operasi AWS API, itu berarti kredenal AWS Identity and Access Management (IAM) yang Anda gunakan tidak memiliki izin yang diperlukan untuk melakukan panggilan tersebut.

Contoh kesalahan berikut terjadi ketika `mateojackson` pengguna mencoba melihat detail tentang target yang dapat diskalakan, tetapi tidak memiliki `application-autoscaling:DescribeScalableTargets` izin.

```
An error occurred (AccessDeniedException) when calling the DescribeScalableTargets
 operation: User: arn:aws:iam::123456789012:user/mateojackson is not authorized to
 perform: application-autoscaling:DescribeScalableTargets
```

Jika Anda menerima kesalahan ini atau yang serupa, Anda harus menghubungi administrator untuk mendapatkan bantuan.

Administrator untuk akun Anda harus memastikan bahwa Anda memiliki izin untuk mengakses semua tindakan API yang digunakan Application Auto Scaling untuk mengakses sumber daya di layanan target dan. CloudWatch Ada berbagai izin yang diperlukan tergantung pada sumber daya yang Anda gunakan. Application Auto Scaling juga memerlukan izin untuk membuat peran terkait layanan saat pertama kali pengguna mengonfigurasi penskalaan untuk sumber daya tertentu.

Saya adalah administrator dan kebijakan IAM saya mengembalikan kesalahan atau tidak berfungsi seperti yang diharapkan

Selain tindakan Application Auto Scaling, kebijakan IAM Anda harus memberikan izin untuk memanggil layanan target dan. CloudWatch Jika pengguna atau aplikasi tidak memiliki izin tambahan ini, akses mereka mungkin tiba-tiba ditolak. Untuk menulis kebijakan IAM untuk pengguna dan aplikasi di akun Anda, lihat informasi di[Contoh kebijakan berbasis identitas Application Auto Scaling](#).

Untuk informasi tentang bagaimana validasi dilakukan, lihat [Validasi izin untuk panggilan Application Auto Scaling API pada sumber daya target](#).

Perhatikan bahwa beberapa masalah izin juga dapat disebabkan oleh masalah dengan membuat peran yang terkait layanan yang digunakan oleh Application Auto Scaling. Untuk informasi tentang peran terkait layanan, lihat Peran yang ditautkan dengan layanan untuk Application Auto Scaling.

Validasi izin untuk panggilan Application Auto Scaling API pada sumber daya target

Membuat permintaan resmi untuk tindakan Application Auto Scaling API mengharuskan pemanggil API memiliki izin untuk mengakses AWS sumber daya di layanan target dan di. CloudWatch Application Auto Scaling memvalidasi izin untuk permintaan yang terkait dengan layanan target dan CloudWatch sebelum melanjutkan dengan permintaan. Untuk mencapai hal ini, kami mengeluarkan serangkaian panggilan untuk memvalidasi izin IAM pada sumber daya target. Ketika respons dikembalikan, itu dibaca oleh Application Auto Scaling. Jika izin IAM tidak mengizinkan tindakan tertentu, Application Auto Scaling menggagalkan permintaan dan mengembalikan kesalahan untuk pengguna yang berisi informasi tentang izin yang hilang. Ini memastikan bahwa konfigurasi penskalaan yang ingin diterapkan pengguna berfungsi sebagaimana dimaksud, dan kesalahan yang berguna akan ditampilkan jika permintaan gagal.

Sebagai contoh cara kerjanya, informasi berikut memberikan rincian tentang bagaimana Application Auto Scaling melakukan validasi izin dengan Aurora dan. CloudWatch

Ketika pengguna memanggil `RegisterScalableTarget` API terhadap cluster Aurora DB, Application Auto Scaling melakukan semua pemeriksaan berikut untuk memverifikasi bahwa pengguna memiliki izin yang diperlukan (dalam huruf tebal).

- RDS:create DBInstance: Untuk menentukan apakah pengguna memiliki izin ini, kami mengirim permintaan ke operasi `CreateDBInstance` API, mencoba membuat instance DB dengan parameter tidak valid (ID instance kosong) di cluster Aurora DB yang ditentukan pengguna. Untuk pengguna resmi, API mengembalikan sebuah respons kode kesalahan `InvalidParameterValue` setelah API mengaudit permintaan. Namun, untuk pengguna yang tidak sah, kami mendapatkan kesalahan `AccessDenied` dan menggagalkan permintaan Application Auto Scaling dengan kesalahan `ValidationException` untuk pengguna yang mencantumkan izin yang hilang.
- RDS:delete DBInstance: Kami mengirim ID instance kosong ke operasi API. `DeleteDBInstance` Untuk pengguna yang berwenang, permintaan ini menghasilkan kesalahan `InvalidParameterValue`. Untuk pengguna yang tidak sah, menghasilkan `AccessDenied` dan mengirimkan pengecualian validasi kepada pengguna (perlakuan yang sama seperti yang dijelaskan dalam titik peluru pertama).

- rds: AddTagsToResource: Karena operasi `AddTagsToResource` API memerlukan Nama Sumber Daya Amazon (ARN), maka perlu untuk menentukan sumber daya “dummy” menggunakan ID akun yang tidak valid (12345) dan ID instance dummy () untuk membuat ARN (non-existing-db). `arn:aws:rds:us-east-1:12345:db:non-existing-db` Untuk pengguna yang berwenang, permintaan ini menghasilkan kesalahan `InvalidParameterValue`. Untuk pengguna yang tidak sah, menghasilkan `AccessDenied` dan mengirimkan pengecualian validasi kepada pengguna.
- RDS:describe DBClusters: Kami menjelaskan nama cluster untuk sumber daya yang terdaftar untuk penskalaan otomatis. Untuk pengguna yang berwenang, kami mendapatkan hasil menggambarkan valid. Untuk pengguna yang tidak sah, menghasilkan `AccessDenied` dan mengirimkan pengecualian validasi kepada pengguna.
- RDS:describe DBInstances: Kami memanggil `DescribeDBInstances` API dengan `db-cluster-id` filter yang memfilter nama cluster yang disediakan oleh pengguna untuk mendaftarkan target yang dapat diskalakan. Untuk pengguna yang berwenang, kami diizinkan untuk menjelaskan semua instans DB dalam klaster DB. Untuk pengguna yang tidak sah, panggilan ini menghasilkan `AccessDenied` dan mengirimkan pengecualian validasi kepada pengguna.
- cloudwatch:PutMetricAlarm: Kami memanggil `PutMetricAlarm` API tanpa parameter apa pun. Karena nama alarm hilang, permintaan mengakibatkan `ValidationError` untuk pengguna yang berwenang. Untuk pengguna yang tidak sah, menghasilkan `AccessDenied` dan mengirimkan pengecualian validasi kepada pengguna.
- cloudwatch: DescribeAlarms: Kami memanggil `DescribeAlarms` API dengan jumlah maksimum nilai rekaman yang disetel ke 1. Untuk pengguna yang berwenang, kami mengharapkan informasi tentang satu alarm dalam respons. Untuk pengguna yang tidak sah, panggilan ini menghasilkan `AccessDenied` dan mengirimkan pengecualian validasi kepada pengguna.
- cloudwatch:DeleteAlarms: Mirip dengan `PutMetricAlarm` di atas, kami tidak menyediakan parameter untuk `DeleteAlarms` diminta. Karena nama alarm hilang dari permintaan, panggilan ini gagal dengan `ValidationError` untuk pengguna yang berwenang. Untuk pengguna yang tidak sah, menghasilkan `AccessDenied` dan mengirimkan pengecualian validasi kepada pengguna.

Setiap kali salah satu dari pengecualian validasi ini terjadi, itu dicatat. Anda dapat mengambil langkah-langkah untuk secara manual mengidentifikasi panggilan mana yang gagal validasi dengan menggunakan AWS CloudTrail. Untuk informasi selengkapnya, silakan lihat Panduan Pengguna AWS CloudTrail.

> **Note**
>
> Jika Anda menerima peringatan untuk acara Application Auto Scaling yang CloudTrail digunakan, peringatan ini akan menyertakan panggilan Application Auto Scaling untuk memvalidasi izin pengguna secara default. Untuk memfilter peringatan ini, gunakan `invokedBy` bidang, yang akan berisi `application-autoscaling.amazonaws.com` untuk pemeriksaan validasi ini.

Akses Application Auto Scaling menggunakan titik akhir VPC antarmuka

Anda dapat menggunakan AWS PrivateLink untuk membuat koneksi pribadi antara VPC Anda dan Application Auto Scaling. Anda dapat mengakses Application Auto Scaling seolah-olah berada di VPC Anda, tanpa menggunakan gateway internet, perangkat NAT, koneksi VPN, atau koneksi. Direct Connect Instans di VPC Anda tidak memerlukan alamat IP publik untuk mengakses Application Auto Scaling.

Anda membuat koneksi pribadi ini dengan membuat titik akhir antarmuka, didukung oleh AWS PrivateLink. Kami membuat antarmuka jaringan endpoint di setiap subnet yang Anda aktifkan untuk titik akhir antarmuka. Ini adalah antarmuka jaringan yang dikelola pemohon yang berfungsi sebagai titik masuk untuk lalu lintas yang ditujukan untuk Application Auto Scaling.

Untuk informasi selengkapnya, lihat Akses Layanan AWS melalui AWS PrivateLink di AWS PrivateLink Panduan.

Daftar Isi

Membuat titik akhir VPC antarmuka

Buat titik akhir untuk Application Auto Scaling menggunakan nama layanan berikut:

```
com.amazonaws.region.application-autoscaling
```

Untuk informasi selengkapnya, lihat Mengakses AWS layanan menggunakan titik akhir VPC antarmuka di Panduan.AWS PrivateLink

Anda tidak perlu mengubah pengaturan lainnya. Application Auto Scaling memanggil AWS layanan lain menggunakan titik akhir layanan atau titik akhir VPC antarmuka pribadi, mana saja yang digunakan.

Buat kebijakan titik akhir VPC

Anda dapat melampirkan kebijakan ke titik akhir VPC Anda untuk mengontrol akses ke API Application Auto Scaling. Kebijakan menentukan:

- Prinsipal yang dapat melakukan tindakan.
- Tindakan yang dapat dilakukan.
- Sumber daya yang dapat digunakan untuk melakukan tindakan.

Contoh berikut menunjukkan kebijakan endpoint VPC yang menolak izin setiap orang untuk menghapus kebijakan penskalaan melalui endpoint. Kebijakan contoh juga memberikan izin kepada semua orang untuk melakukan semua tindakan lainnya.

```
{
   "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        },
        {
            "Action": "application-autoscaling:DeleteScalingPolicy",
            "Effect": "Deny",
            "Resource": "*",
            "Principal": "*"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat kebijakan titik akhir VPC di Panduan.AWS PrivateLink

Ketahanan dalam Application Auto Scaling

Infrastruktur AWS global dibangun di sekitar AWS Wilayah dan Zona Ketersediaan.

AWS Wilayah menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang sangat redundan.

Dengan Availability Zone, Anda dapat mendesain dan mengoperasikan aplikasi dan basis data yang secara otomatis mengalami kegagalan di antara zona tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur pusat data tunggal atau multi tradisional.

Untuk informasi selengkapnya tentang AWS Wilayah dan Availability Zone, lihat infrastruktur AWS global.

Keamanan infrastruktur dalam Application Auto Scaling

Sebagai layanan terkelola, Application Auto Scaling dilindungi oleh keamanan jaringan AWS global. Untuk informasi tentang layanan AWS keamanan dan cara AWS melindungi infrastruktur, lihat Keamanan AWS Cloud. Untuk mendesain AWS lingkungan Anda menggunakan praktik terbaik untuk keamanan infrastruktur, lihat Perlindungan Infrastruktur dalam Kerangka Kerja yang AWS Diarsiteksikan dengan Baik Pilar Keamanan.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses Application Auto Scaling melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Pengangkutan (TLS). Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Sandi cocok dengan sistem kerahasiaan maju sempurna (perfect forward secrecy, PFS) seperti DHE (Ephemeral Diffie-Hellman) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Sebagian besar sistem modern seperti Java 7 dan versi lebih baru mendukung mode-mode ini.

Validasi kepatuhan untuk Application Auto Scaling

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat Layanan AWS di Lingkup oleh Program Kepatuhan Layanan AWS dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat Program AWS Kepatuhan Program AWS .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat Mengunduh Laporan di AWS Artifact .

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. Untuk informasi selengkapnya tentang tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS, lihat Dokumentasi AWS Keamanan.

Kuota untuk Application Auto Scaling

Anda Akun AWS memiliki kuota default, sebelumnya disebut sebagai batas, untuk masing-masing. Layanan AWS Kecuali dinyatakan lain, setiap kuota bersifat khusus per Wilayah. Anda dapat meminta peningkatan untuk beberapa kuota dan kuota lainnya tidak dapat ditingkatkan.

Untuk melihat kuota Application Auto Scaling, buka konsol Service Quotas. Di panel navigasi, pilih AWS layanan dan pilih Application Auto Scaling.

Untuk meminta peningkatan kuota, lihat Meminta Peningkatan Kuota dalam Panduan Pengguna Service Quotas.

Anda Akun AWS memiliki kuota berikut yang terkait dengan Application Auto Scaling.

Nama	Default	Dapat disesuaikan
Target yang dapat diskalakan per jenis sumber daya	Amazon DynamoDB: 5.000 | Amazon ECS: 3.000 | Amazon Keyspaces: 1.500 | Jenis sumber daya lainnya: 500	Ya
Kebijakan penskalaan per target yang dapat diskalakan (baik kebijakan penskalaan langkah maupun pelacakan target)	50	Tidak
Tindakan terjadwal per target yang dapat diskalakan	200	Tidak
Penyesuaian langkah per kebijakan penskalaan langkah	20	Ya

Selalu ingat kuota layanan saat Anda meningkatkan beban kerja Anda. Misalnya, ketika Anda mencapai jumlah maksimum unit kapasitas yang diizinkan oleh layanan, penskalaan akan berhenti. Jika permintaan dan kapasitas saat ini menurun, Application Auto Scaling dapat ditingkatkan lagi. Untuk menghindari mencapai batas kapasitas ini lagi, Anda dapat meminta peningkatan. Setiap

layanan memiliki kuota default sendiri untuk kapasitas maksimum sumber daya. Untuk informasi tentang kuota default untuk Amazon Web Services lainnya, lihat Titik akhir dan kuota layanan di. Referensi Umum Amazon Web Services

Riwayat dokumen untuk Application Auto Scaling

Tabel berikut menjelaskan tambahan penting pada dokumentasi Application Auto Scaling, yang dimulai pada Januari 2018. Untuk pemberitahuan tentang pembaruan dokumentasi ini, Anda dapat berlangganan ke feed RSS.

Perubahan	Deskripsi	Tanggal
Tambahkan dukungan untuk cluster ElastiCache Memcached	Gunakan Application Auto Scaling untuk menskalakan jumlah node secara horizontal untuk cluster Memcached. Untuk informasi selengkapnya, lihat ElastiCache dan Application Auto Scaling.	April 10, 2025
AWS pembaruan kebijakan terkelola	Application Auto Scaling memperbarui kebijakan. `AWSApplicationAutoscalingElastiCacheRGPolicy`	April 10, 2025
Perubahan panduan	Topik baru dalam Panduan Pengguna Application Auto Scaling membantu Anda memulai menggunakan penskalaan prediktif dengan Application Auto Scaling. Lihat Penskalaan prediktif Application Auto Scaling.	November 21, 2024
AWS pembaruan kebijakan terkelola	Application Auto Scaling memperbarui kebijakan. `AWSApplicationAutoscalingECSServicePolicy`	November 21, 2024

Tambahkan dukungan untuk kumpulan WorkSpaces	Gunakan Application Auto Scaling untuk menskalakan kumpulan. WorkSpaces Untuk informasi selengkapnya, lihat Amazon WorkSpaces dan Application Auto Scaling. Topik Pembaruan Application Auto Scaling ke kebijakan AWS terkelola telah diperbarui untuk mencantumkan kebijakan terkelola baru untuk integrasi. WorkSpaces	27 Juni 2024
Perubahan panduan	Memperbarui jumlah maksimum target yang dapat diskalakan per entri jenis sumber daya dalam dokumentasi kuota. Lihat Kuota untuk Application Auto Scaling.	Januari 16, 2024
Support untuk SageMaker komponen inferensi AI	Gunakan Application Auto Scaling untuk menskalakan salinan komponen inferensi.	November 29, 2023
AWS pembaruan kebijakan terkelola	Application Auto Scaling memperbarui kebijakan. `AWSApplicationAutoscalingSageMakerEndpointPolicy`	13 November 2023
Support untuk konkurensi yang disediakan tanpa server SageMaker AI	Gunakan Application Auto Scaling untuk menskalakan konkurensi yang disediakan dari titik akhir tanpa server.	9 Mei 2023

Kategorikan target skalabel Anda menggunakan tag	Anda sekarang dapat menetapkan metadata ke target scalable Application Auto Scaling Anda dalam bentuk tag. Lihat Dukungan penandaan untuk Application Auto Scaling.	20 Maret 2023
Support untuk matematika CloudWatch metrik	Sekarang Anda dapat menggunakan matematik a metrik saat membuat kebijakan penskalaan pelacakan target. Dengan matematika metrik, Anda dapat menanyakan beberapa CloudWatch metrik dan menggunakan ekspresi matematika untuk membuat deret waktu baru berdasark an metrik ini. Lihat Membuat kebijakan penskalaan pelacakan target untuk Application Auto Scaling menggunakan matematika metrik.	14 Maret 2023
Alasan untuk tidak menskalak an	Anda sekarang dapat mengambil alasan yang dapat dibaca mesin untuk Application Auto Scaling tidak menskalakan sumber daya Anda menggunakan Applicati on Auto Scaling API. Lihat Aktivitas penskalaan untuk Application Auto Scaling.	4 Januari 2023

Perubahan panduan	Memperbarui jumlah maksimum target yang dapat diskalakan per entri jenis sumber daya dalam dokumentasi kuota. Lihat Kuota untuk Application Auto Scaling.	6 Mei 2022
Tambahkan dukungan untuk cluster Amazon Neptunus	Gunakan Application Auto Scaling untuk menskalakan jumlah replika di kluster Amazon Neptunus DB. Untuk informasi selengkapnya, lihat Amazon Neptunus dan Application Auto Scaling. Topik Pembaruan Application Auto Scaling ke kebijakan AWS terkelola telah diperbarui untuk mencantumkan kebijakan terkelola baru untuk integrasi dengan Neptunus.	Oktober 6, 2021
Application Auto Scaling sekarang melaporkan perubahan pada kebijakan yang dikelola AWS	Mulai 19 Agustus 2021, perubahan kebijakan terkelola dilaporkan dalam topik Pembaruan Application Auto Scaling ke kebijakan AWS terkelola. Perubahan pertama yang tercantum adalah penambahan izin yang diperlukan untuk ElastiCache (Redis OSS).	19 Agustus 2021

Tambahkan dukungan untuk grup ElastiCache replikasi (Redis OSS)	Gunakan Application Auto Scaling untuk menskalakan jumlah grup node dan jumlah replika per grup node untuk grup replikasi ElastiCache (Redis OSS) (cluster). Untuk informasi selengkapnya, lihat ElastiCache (Redis OSS) dan Application Auto Scaling.	19 Agustus 2021
Perubahan panduan	Topik IAM baru dalam Panduan Pengguna Application Auto Scaling membantu Anda memecahkan masalah akses ke Application Auto Scaling. Untuk informasi selengkapnya, lihat Identity and Access Management untuk Application Auto Scaling. Juga menambahkan contoh baru kebijakan izin IAM untuk tindakan pada layanan target dan Amazon. CloudWatch Untuk informasi selengkapnya, lihat Contoh kebijakan untuk bekerja dengan AWS CLI atau SDK.	23 Februari 2021

Tambahkan dukungan untuk zona waktu lokal	Anda sekarang dapat membuat tindakan terjadwal di zona waktu lokal. Jika zona waktu Anda mengamati daylight saving time, maka secara otomatis menyesuaikan Daylight Saving Time (DST). Untuk informasi selengkapnya, lihat Penskalaan terjadwal.	2 Februari 2021
Perubahan panduan	Tutorial baru dalam Panduan Pengguna Application Auto Scaling membantu Anda memahami cara menggunakan kebijakan penskalaan pelacakan target dan penskalaan terjadwal untuk meningkatkan ketersediaan aplikasi Anda saat menggunakan Application Auto Scaling.	15 Oktober 2020
Tambahkan dukungan untuk Amazon Managed Streaming untuk Apache Kafka Kafka Cluster Storage	Menggunakan kebijakan penskalaan pelacakan target untuk menskalakan jumlah penyimpanan broker yang terkait dengan kluster Amazon MSK.	Rabu, 30 September 2020
Tambahkan dukungan untuk titik akhir pengenal entitas Amazon Comprehend	Menggunakan Application Auto Scaling untuk menskalakan jumlah unit inferensi yang disediakan untuk titik akhir pengenal entitas Amazon Comprehend Anda.	Senin, 28 September 2020

Tambahkan dukungan untuk Amazon Keyspaces (untuk Apache Cassandra) tabel	Menggunakan Application Auto Scaling untuk menskalakan hasil yang disediakan (kapasitas baca dan tulis) dari tabel Amazon Keyspaces.	Kamis, 23 April 2020
Bab “Keamanan” baru	Bab Keamanan baru di Panduan Pengguna Application Auto Scaling membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan Application Auto Scaling. Sebagai bagian dari pembaruan ini, bab panduan pengguna "Autentikasi dan Kontrol Akses" telah diganti dengan bagian baru yang lebih berguna, Identity and Access Management untuk Application Auto Scaling.	Kamis, 16 Januari 2020
Pembaruan kecil	Berbagai perbaikan dan koreksi.	Rabu, 15 Januari 2020
Tambahkan fungsionalitas pemberitahuan	Application Auto Scaling sekarang mengirimkan peristiwa ke Amazon EventBridge dan pemberitahuan kepada Anda Dasbor AWS Health ketika tindakan tertentu terjadi. Untuk informasi lebih lanjut, lihat pemantauan Application Auto Scaling.	20 Desember 2019

Tambahkan dukungan untuk AWS Lambda fungsi	Gunakan Application Auto Scaling untuk menskalakan ketersediaan konkurensi fungsi Lambda.	3 Desember 2019
Tambahkan dukungan untuk titik akhir klasifikasi dokumen Amazon Comprehend	Gunakan Application Auto Scaling untuk menskalakan kapasitas throughput titik akhir klasifikasi dokumen Amazon Comprehend.	Senin, 25 November 2019
Tambahkan dukungan WorkSpaces Aplikasi untuk kebijakan penskalaan pelacakan target	Gunakan kebijakan penskalaan pelacakan target untuk menskalakan ukuran armada WorkSpaces Aplikasi.	25 November 2019
Dukungan untuk titik akhir VPC Amazon VPC	Sekarang Anda dapat membuat koneksi pribadi antara VPC dan Application Auto Scaling. Untuk pertimbangan dan instruksi migrasi, lihat Application Auto Scaling dan titik akhir VPC antarmuka.	Jumat, 22 November 2019
Tangguhkan dan lanjutkan penskalaan	Dukungan tambahan untuk menangguhkan dan melanjutkan penskalaan. Untuk informasi lebih lanjut, lihat Menangguhkan dan melanjutkan penskalaan untuk Application Auto Scaling.	Kamis, 29 Agustus 2019

Perubahan panduan	Meningkatkan dokumentasi Application Auto Scaling dalam bagian Penskalaan terjadwal, Kebijakan penskalaan langkah, dan Kebijakan penskalaan pelacakan target.	Senin, 11 Maret 2019
Tambahkan dukungan untuk sumber daya khusus	Menggunakan Application Auto Scaling untuk menskalakan sumber daya kustom yang disediakan oleh aplikasi atau layanan Anda sendiri. Untuk informasi lebih lanjut, lihat repositori GitHub kami.	9 Juli 2018
Tambahkan dukungan untuk varian titik akhir SageMaker AI	Gunakan Application Auto Scaling untuk menskalakan jumlah instans titik akhir yang disediakan untuk varian.	Rabu, 28 Februari 2018

Tabel berikut menjelaskan perubahan penting pada dokumentasi Application Auto Scaling sebelum Januari 2018.

Perubahan	Deskripsi	Tanggal
Menambahkan dukungan untuk Replika Aurora	Menggunakan Application Auto Scaling untuk menskalakan jumlah yang diinginkan. Untuk informasi lebih lanjut, lihat Menggunakan Amazon Aurora Auto Scaling dengan replika Aurora dalam Panduan Pengguna Amazon RDS.	Jumat, 17 November 2017
Menambahkan dukungan untuk penskalaan terjadwal	Menggunakan penskalaan terjadwal untuk menskalakan	Rabu, 08 November 2017

Perubahan	Deskripsi	Tanggal
	sumber daya pada waktu atau interval yang telah ditentukan sebelumnya. Untuk informasi lebih lanjut, lihat Penskalaan terjadwal untuk Application Auto Scaling.	
Menambahkan dukungan untuk kebijakan penskalaan pelacakan target	Menggunakan kebijakan penskalaan pelacakan target untuk mengatur penskalaan dinamis untuk aplikasi Anda hanya dalam beberapa langkah sederhana. Untuk informasi lebih lanjut, lihat Kebijakan penskalaan pelacakan target untuk Application Auto Scaling.	Rabu, 12 Juli 2017
Menambahkan dukungan untuk kapasitas baca dan tulis yang disediakan untuk tabel DynamoDB dan indeks sekunder global	Menggunakan Application Auto Scaling untuk menskalakan hasil yang disediakan (kapasitas baca dan tulis). Untuk informasi lebih lanjut, lihat Mengelola kapasitas throughput dengan DynamoDB Auto Scaling di Panduan Developer Amazon DynamoDB.	Rabu, 14 Juni 2017

Perubahan	Deskripsi	Tanggal
Tambahkan dukungan untuk armada WorkSpaces Aplikasi	Menggunakan Application Auto Scaling untuk menskalakan ukuran armada. Untuk informasi selengkapnya, lihat Auto Scaling Armada untuk WorkSpaces Aplikasi di Panduan Administrasi WorkSpaces Aplikasi Amazon.	Kamis, 23 Maret 2017
Menambahkan dukungan untuk kluster Amazon EMR	Menggunakan Application Auto Scaling untuk menskalakan node inti dan tugas. Untuk informasi lebih lanjut, lihat Menggunakan penskalaan otomatis di Amazon EMR dalam Panduan Manajemen EMR Amazon.	Jumat, 18 November 2016
Menambahkan dukungan untuk Spot Fleet	Gunakan Application Auto Scaling untuk menskalakan kapasitas target. Untuk informasi selengkapnya, lihat Penskalaan otomatis untuk armada Spot di Panduan Pengguna Amazon EC2.	Kamis, 01 September 2016
Menambahkan dukungan untuk layanan ECS Amazon	Menggunakan Application Auto Scaling untuk menskalakan jumlah yang diinginkan. Untuk informasi lebih lanjut, lihat Service Auto Scaling dalam Panduan Pengembang Layanan Amazon Elastic Container.	Selasa, 09 Agustus 2016

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.