aws

Panduan Developer

Amazon Managed Streaming untuk Apache Kafka

Amazon Managed Streaming untuk Apache Kafka: Panduan Developer

Table of Contents

Selamat datang di Panduan Pengembang MSK Amazon

Selamat datang di Panduan Pengembang MSK Amazon. Topik berikut dapat membantu Anda mulai menggunakan panduan ini, berdasarkan apa yang Anda coba lakukan.

- Buat cluster MSK Amazon dengan mengikuti Memulai menggunakan Amazon MSK tutorial.
- Selami lebih dalam fungsionalitas Amazon MSK diAmazon MSK: Cara kerjanya.
- Jalankan Apache Kafka tanpa harus mengelola dan menskalakan kapasitas cluster dengan. MSK Tanpa Server
- Gunakan MSK Connect untuk mengalirkan data ke dan dari cluster Apache Kafka Anda.
- Gunakan Replikator MSK untuk mereplikasi data secara andal di seluruh kluster MSK Amazon di AWS wilayah yang berbeda atau sama.

Untuk sorotan, detail produk, dan harga, lihat halaman layanan untuk Amazon MSK.

Apa itu Amazon MSK?

Amazon Managed Streaming for Apache Kafka (Amazon MSK) adalah layanan yang dikelola sepenuhnya yang memungkinkan Anda membangun dan menjalankan aplikasi yang menggunakan Apache Kafka untuk memproses data streaming. Amazon MSK menyediakan operasi bidang kontrol, seperti untuk membuat, memperbarui, dan menghapus cluster. Ini memungkinkan Anda menggunakan operasi data-plane Apache Kafka, seperti untuk memproduksi dan mengkonsumsi data. Ini menjalankan versi open-source Apache Kafka. Ini berarti aplikasi, perkakas, dan plugin yang ada dari mitra dan komunitas Apache Kafka didukung tanpa memerlukan perubahan pada kode aplikasi. Anda dapat menggunakan Amazon MSK untuk membuat cluster yang menggunakan salah satu versi Apache Kafka yang tercantum di bawah. the section called “Versi Apache Kafka yang didukung”

Komponen-komponen ini menggambarkan arsitektur Amazon MSK:

- Node broker — Saat membuat klaster MSK Amazon, Anda menentukan berapa banyak node broker yang ingin dibuat Amazon MSK di setiap Availability Zone. Minimal adalah satu broker per Availability Zone. Setiap Availability Zone memiliki subnet virtual private cloud (VPC) sendiri.
- ZooKeeper node — Amazon MSK juga membuat ZooKeeper node Apache untuk Anda. Apache ZooKeeper adalah server open-source yang memungkinkan koordinasi terdistribusi yang sangat andal.

- Pengontrol Kraft —Komunitas Apache Kafka mengembangkan kRAFT untuk menggantikan Apache untuk manajemen metadata di cluster Apache ZooKeeper Kafka. Dalam mode kRAFT, metadata cluster disebarkan dalam sekelompok pengendali Kafka, yang merupakan bagian dari cluster Kafka, bukan di seluruh node. ZooKeeper Pengontrol Kraft disertakan tanpa biaya tambahan untuk Anda, dan tidak memerlukan pengaturan atau manajemen tambahan dari Anda.

  > **Note**
  >
  > Dari Apache Kafka versi 3.7.x di MSK, Anda dapat membuat cluster yang menggunakan mode Kraft alih-alih mode. ZooKeeper

- Produsen, konsumen, dan pembuat topik — Amazon MSK memungkinkan Anda menggunakan operasi pesawat data Apache Kafka untuk membuat topik dan memproduksi serta mengkonsumsi data.
- Operasi Cluster Anda dapat menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau API di SDK untuk melakukan operasi bidang kontrol. Misalnya, Anda dapat membuat atau menghapus kluster MSK Amazon, mencantumkan semua cluster di akun, melihat properti cluster, dan memperbarui jumlah dan jenis broker dalam sebuah cluster.

Amazon MSK mendeteksi dan secara otomatis pulih dari skenario kegagalan yang paling umum untuk cluster sehingga produsen dan aplikasi konsumen Anda dapat melanjutkan operasi tulis dan baca mereka dengan dampak minimal. Ketika Amazon MSK mendeteksi kegagalan broker, itu mengurangi kegagalan atau mengganti broker yang tidak sehat atau tidak terjangkau dengan yang baru. Selain itu, jika memungkinkan, ia menggunakan kembali penyimpanan dari broker yang lebih tua untuk mengurangi data yang perlu ditiru Apache Kafka. Dampak ketersediaan Anda terbatas pada waktu yang diperlukan Amazon MSK untuk menyelesaikan deteksi dan pemulihan. Setelah pemulihan, aplikasi produsen dan konsumen Anda dapat terus berkomunikasi dengan alamat IP broker yang sama yang mereka gunakan sebelum kegagalan.

Menyiapkan Amazon MSK

Sebelum Anda menggunakan Amazon MSK untuk pertama kalinya, selesaikan tugas-tugas berikut.

Tugas

- Mendaftar untuk AWS
- Unduh perpustakaan dan alat

Mendaftar untuk AWS

Ketika Anda mendaftar AWS, akun Amazon Web Services Anda secara otomatis mendaftar untuk semua layanan di AWS, termasuk Amazon MSK. Anda hanya membayar biaya layanan yang Anda gunakan.

Jika Anda sudah memiliki AWS akun, lompat ke tugas berikutnya. Jika Anda belum memiliki akun AWS , gunakan prosedur berikut untuk membuatnya.

Mendaftar Amazon Web Services

1. Buka https://portal.aws.amazon.com/billing/signup.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan tindakan menerima panggilan telepon dan memasukkan kode verifikasi di keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

Unduh perpustakaan dan alat

Pustaka dan alat berikut dapat membantu Anda bekerja dengan Amazon MSK:

- The AWS Command Line Interface (AWS CLI) mendukung Amazon MSK. AWS CLI Ini memungkinkan Anda untuk mengontrol beberapa Amazon Web Services dari baris perintah dan mengotomatiskannya melalui skrip. Tingkatkan versi Anda AWS CLI ke versi terbaru untuk

memastikan bahwa ia memiliki dukungan untuk fitur MSK Amazon yang didokumentasikan dalam panduan pengguna ini. Untuk petunjuk terperinci tentang cara meng-upgrade AWS CLI, lihat Menginstal AWS Command Line Interface. Setelah Anda menginstal AWS CLI, Anda harus mengkonfigurasinya. Untuk informasi tentang cara mengonfigurasi AWS CLI, lihat aws configure.

- Referensi API Amazon Managed Streaming for Kafka API mendokumentasikan operasi API yang didukung Amazon MSK.
- Amazon Web Services SDK untuk Go, Java,, .NET JavaScript, Node.js, PHP, Python, dan Ruby menyertakan dukungan Amazon MSK dan sampel.

Memulai menggunakan Amazon MSK

Tutorial ini menunjukkan contoh bagaimana Anda dapat membuat cluster MSK, memproduksi dan mengkonsumsi data, dan memantau kesehatan cluster Anda menggunakan metrik. Contoh ini tidak mewakili semua opsi yang dapat Anda pilih saat Anda membuat klaster MSK. Di berbagai bagian tutorial ini, kami memilih opsi default untuk kesederhanaan. Ini tidak berarti bahwa mereka adalah satu-satunya opsi yang berfungsi untuk menyiapkan cluster MSK atau instance klien.

Topik

Langkah 1: Buat cluster MSK Amazon

Pada langkah Memulai Menggunakan Amazon MSK ini, Anda membuat cluster MSK Amazon.

Untuk membuat cluster MSK Amazon menggunakan AWS Management Console

1. Masuk ke AWS Management Console, dan buka konsol MSK Amazon di https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Pilih Buat klaster.
3. Untuk metode Creation, biarkan opsi Quick create dipilih. Opsi Quick create memungkinkan Anda membuat cluster dengan pengaturan default.
4. Untuk nama Cluster, masukkan nama deskriptif untuk klaster Anda. Misalnya, **`MSKTutorialCluster`**.
5. Untuk properti klaster Umum, pilih Disediakan sebagai tipe Cluster.
6. Dari tabel di bawah Semua pengaturan cluster, salin nilai pengaturan berikut dan simpan karena Anda membutuhkannya nanti dalam tutorial ini:

- VPC
- Subnet
- Grup keamanan yang terkait dengan VPC

7. Pilih Buat klaster.
8. Periksa Status cluster pada halaman ringkasan Cluster. Status berubah dari Creating menjadi Active karena Amazon MSK menyediakan klaster. Ketika statusnya Aktif, Anda dapat terhubung ke cluster. Untuk informasi selengkapnya tentang status klaster, lihatNegara cluster.

Langkah Selanjutnya

Langkah 2: Buat peran IAM

Langkah 2: Buat peran IAM

Pada langkah ini, Anda melakukan dua tugas. Tugas pertama adalah membuat kebijakan IAM yang memberikan akses untuk membuat topik di cluster dan mengirim data ke topik tersebut. Tugas kedua adalah membuat peran IAM dan mengaitkan kebijakan ini dengannya. Pada langkah selanjutnya, Anda membuat mesin klien yang mengasumsikan peran ini dan menggunakannya untuk membuat topik di klaster dan mengirim data ke topik tersebut.

Untuk membuat kebijakan IAM yang memungkinkan untuk membuat topik dan menulis kepada mereka

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, pilih Kebijakan.
3. Pilih Buat Kebijakan.
4. Pilih tab JSON, lalu ganti JSON di jendela editor dengan JSON berikut.

   Ganti *wilayah* dengan kode AWS wilayah tempat Anda membuat cluster. Ganti *Account-ID dengan ID* akun Anda. Ganti *MSK TutorialCluster* dengan nama cluster Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "kafka-cluster:Connect",
                "kafka-cluster:AlterCluster",
                "kafka-cluster:DescribeCluster"
            ],
            "Resource": [
                "arn:aws:kafka:region:Account-ID:cluster/MSKTutorialCluster/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:*Topic*",
                "kafka-cluster:WriteData",
                "kafka-cluster:ReadData"
            ],
            "Resource": [
                "arn:aws:kafka:region:Account-ID:topic/MSKTutorialCluster/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:AlterGroup",
                "kafka-cluster:DescribeGroup"
            ],
            "Resource": [
                "arn:aws:kafka:region:Account-ID:group/MSKTutorialCluster/*"
            ]
        }
    ]
}
```

Untuk petunjuk tentang cara menulis kebijakan aman, lihatthe section called “Kontrol akses IAM”.

5. Pilih Berikutnya: Tanda.
6. Pilih Berikutnya: Tinjau.
7. Untuk nama kebijakan, masukkan nama deskriptif, seperti msk-tutorial-policy.
8. Pilih Buat kebijakan.

Untuk membuat peran IAM dan melampirkan kebijakan padanya

1. Pada panel navigasi, pilih Peran.

2. Pilih Buat peran.
3. Di bawah Kasus penggunaan umum, pilih EC2, lalu pilih Berikutnya: Izin.
4. Di kotak pencarian, masukkan nama kebijakan yang sebelumnya Anda buat untuk tutorial ini. Kemudian pilih kotak di sebelah kiri kebijakan.
5. Pilih Berikutnya: Tanda.
6. Pilih Berikutnya: Tinjau.
7. Untuk nama peran, masukkan nama deskriptif, seperti msk-tutorial-role.
8. Pilih Buat peran.

Langkah Selanjutnya

Langkah 3: Buat mesin klien

Langkah 3: Buat mesin klien

Pada langkah Memulai Menggunakan Amazon MSK ini, Anda membuat mesin klien. Anda menggunakan mesin klien ini untuk membuat topik yang menghasilkan dan mengkonsumsi data. Untuk mempermudah, Anda akan membuat mesin klien ini di VPC yang terkait dengan cluster MSK sehingga klien dapat dengan mudah terhubung ke cluster.

Untuk membuat mesin klien

1. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
2. Pilih Luncurkan Instans.
3. Masukkan Nama untuk mesin klien Anda, seperti**`MSKTutorialClient`**.
4. Tinggalkan Amazon Linux 2 AMI (HVM) - Kernel 5.10, Jenis Volume SSD dipilih untuk jenis Amazon Machine Image (AMI).
5. Biarkan tipe instans t2.micro dipilih.
6. Di bawah Key pair (login), pilih Create a new key pair. Masukkan **`MSKKeyPair`** nama pasangan Kunci, lalu pilih Unduh Pasangan Kunci. Alternatifnya, Anda dapat menggunakan pasangan kunci yang sudah ada.
7. Perluas bagian Detail lanjutan dan pilih peran IAM yang Anda buat di Langkah 2: Buat peran IAM.
8. Pilih Luncurkan instans.

9. Pilih Lihat Instans. Kemudian, di kolom Grup Keamanan, pilih grup keamanan yang terkait dengan instans baru Anda. Salin ID grup keamanan, dan simpan untuk nanti.
10. Buka konsol Amazon VPC di https://console.aws.amazon.com/vpc/.
11. Di panel navigasi, pilih Grup Keamanan. Temukan grup keamanan yang ID Anda simpanthe section called “Langkah 1: Buat cluster”.
12. Di tab Aturan Masuk, pilih Edit aturan masuk.
13. Pilih Tambahkan aturan.
14. Dalam aturan baru, pilih Semua lalu lintas di kolom Jenis. Di bidang kedua di kolom Sumber, pilih grup keamanan mesin klien Anda. Ini adalah grup yang namanya Anda simpan setelah Anda meluncurkan instance mesin klien.
15. Pilih Simpan aturan. Sekarang grup keamanan cluster dapat menerima lalu lintas yang berasal dari grup keamanan mesin klien.

Langkah Selanjutnya

Langkah 4: Buat topik

Langkah 4: Buat topik

Pada langkah Memulai Menggunakan Amazon MSK ini, Anda menginstal pustaka dan alat klien Apache Kafka di mesin klien, dan kemudian Anda membuat topik.

> **Warning**
>
> Nomor versi Apache Kafka yang digunakan dalam tutorial ini adalah contoh saja. Kami menyarankan Anda menggunakan versi klien yang sama dengan versi cluster MSK Anda. Versi klien yang lebih lama mungkin kehilangan fitur tertentu dan perbaikan bug penting.

Untuk menemukan versi kluster MSK Anda

1. Pergi ke https://eu-west-2.console.aws.amazon.com/msk/
2. Pilih cluster MSK.
3. Perhatikan versi Apache Kafka yang digunakan pada cluster.
4. Ganti contoh nomor versi Amazon MSK dalam tutorial ini dengan versi yang diperoleh pada Langkah 3.

Untuk membuat topik di mesin klien

1. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
2. Di panel navigasi, pilih Instans. Kemudian pilih kotak centang di samping nama mesin klien yang Anda buatLangkah 3: Buat mesin klien.
3. Pilih Actions, lalu pilih Connect. Ikuti petunjuk di konsol untuk terhubung ke mesin klien Anda.
4. Instal Java pada mesin klien dengan menjalankan perintah berikut:

```
sudo yum -y install java-11
```

5. Jalankan perintah berikut untuk mengunduh Apache Kafka.

```
wget https://archive.apache.org/dist/kafka/{YOUR MSK VERSION}/kafka_2.13-{YOUR MSK
 VERSION}.tgz
```

> **Note**
>
> Jika Anda ingin menggunakan situs cermin selain yang digunakan dalam perintah ini, Anda dapat memilih yang berbeda di situs web Apache.

6. Jalankan perintah berikut di direktori tempat Anda mengunduh file TAR pada langkah sebelumnya.

```
tar -xzf kafka_2.13-{YOUR MSK VERSION}.tgz
```

7. Buka `kafka_2.13-{YOUR MSK VERSION}/libs` direktori, lalu jalankan perintah berikut untuk mengunduh file Amazon MSK IAM JAR. Amazon MSK IAM JAR memungkinkan mesin klien mengakses cluster.

```
wget https://github.com/aws/aws-msk-iam-auth/releases/download/v1.1.1/aws-msk-iam-
auth-1.1.1-all.jar
```

8. Pergi ke `kafka_2.13-{YOUR MSK VERSION}/bin` direktori. Salin pengaturan properti berikut dan tempel ke file baru. Beri nama file **`client.properties`** dan simpan.

```
security.protocol=SASL_SSL
sasl.mechanism=AWS_MSK_IAM
sasl.jaas.config=software.amazon.msk.auth.iam.IAMLoginModule required;
```

```
sasl.client.callback.handler.class=software.amazon.msk.auth.iam.IAMClientCallbackHandler
```

9. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
10. Tunggu status klaster Anda menjadi Aktif. Ini mungkin memakan waktu beberapa menit. Setelah status menjadi Aktif, pilih nama cluster. Ini membawa Anda ke halaman yang berisi ringkasan cluster.
11. Pilih Lihat informasi klien.
12. Salin string koneksi untuk titik akhir pribadi.

    Anda akan mendapatkan tiga titik akhir untuk masing-masing broker. Anda hanya perlu satu titik akhir broker untuk langkah berikut.
13. Jalankan perintah berikut, ganti *BootstrapServerString* dengan salah satu endpoint broker yang Anda peroleh pada langkah sebelumnya.

    ```
    <path-to-your-kafka-installation>/bin/kafka-topics.sh --create --bootstrap-server
     BootstrapServerString --command-config client.properties --replication-factor 3 --
    partitions 1 --topic MSKTutorialTopic
    ```

    Jika perintah berhasil, Anda melihat pesan berikut: `Created topic MSKTutorialTopic.`

Langkah Selanjutnya

Langkah 5: Menghasilkan dan mengkonsumsi data

Langkah 5: Menghasilkan dan mengkonsumsi data

Dalam langkah Memulai Menggunakan Amazon MSK ini, Anda memproduksi dan mengkonsumsi data.

Untuk menghasilkan dan mengkonsumsi pesan

1. Jalankan perintah berikut untuk memulai produsen konsol. Ganti *BootstrapServerString dengan string* koneksi plaintext yang Anda peroleh di Buat topik. Untuk petunjuk tentang cara mengambil string koneksi ini, lihat Mendapatkan broker bootstrap untuk klaster MSK Amazon.

   ```
   <path-to-your-kafka-installation>/bin/kafka-console-producer.sh --
   broker-list BootstrapServerString --producer.config client.properties --
   topic MSKTutorialTopic
   ```

2. Masukkan pesan apa pun yang Anda inginkan, dan tekan Enter. Ulangi langkah ini dua atau tiga kali. Setiap kali Anda memasukkan baris dan tekan Enter, baris itu dikirim ke cluster Apache Kafka Anda sebagai pesan terpisah.
3. Biarkan koneksi ke mesin klien tetap terbuka, dan kemudian buka koneksi kedua yang terpisah ke mesin itu di jendela baru.
4. Dalam perintah berikut, ganti *BootstrapServerString dengan string* koneksi plaintext yang Anda simpan sebelumnya. Kemudian, untuk membuat konsumen konsol, jalankan perintah berikut dengan koneksi kedua Anda ke mesin klien.

   ```
   <path-to-your-kafka-installation>/bin/kafka-console-consumer.sh --bootstrap-
   server BootstrapServerString --consumer.config client.properties --
   topic MSKTutorialTopic --from-beginning
   ```

   Anda mulai melihat pesan yang Anda masukkan sebelumnya saat Anda menggunakan perintah produser konsol.
5. Masukkan lebih banyak pesan di jendela produser, dan saksikan mereka muncul di jendela konsumen.

Langkah Selanjutnya

Langkah 6: Gunakan Amazon CloudWatch untuk melihat metrik MSK Amazon

Langkah 6: Gunakan Amazon CloudWatch untuk melihat metrik MSK Amazon

Pada langkah Memulai Menggunakan Amazon MSK ini, Anda melihat metrik MSK Amazon di Amazon. CloudWatch

Untuk melihat metrik MSK Amazon di CloudWatch

1. Buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
2. Pada panel navigasi, silakan pilih Metrik.
3. Pilih tab Semua metrik, lalu pilih AWS/Kafka.
4. Untuk melihat metrik tingkat broker, pilih ID Broker, Nama Cluster. Untuk metrik tingkat cluster, pilih Nama Cluster.

5. (Opsional) Di panel grafik, pilih statistik dan periode waktu, lalu buat CloudWatch alarm menggunakan pengaturan ini.

Langkah Selanjutnya

Langkah 7: Hapus AWS sumber daya yang dibuat untuk tutorial ini

Langkah 7: Hapus AWS sumber daya yang dibuat untuk tutorial ini

Pada langkah terakhir Memulai Menggunakan Amazon MSK, Anda menghapus cluster MSK dan mesin klien yang Anda buat untuk tutorial ini.

Untuk menghapus sumber daya menggunakan AWS Management Console

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Pilih nama cluster Anda. Misalnya, MSK TutorialCluster.
3. Pilih Tindakan, lalu pilih Hapus.
4. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
5. Pilih contoh yang Anda buat untuk mesin klien Anda, misalnya,**`MSKTutorialClient`**.
6. Pilih status Instance, lalu pilih Terminate instance.

Untuk menghapus kebijakan IAM

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, pilih Peran.
3. Di kotak pencarian, masukkan nama peran IAM yang Anda buat untuk tutorial ini.
4. Pilih perannya. Kemudian pilih Hapus peran, dan konfirmasikan penghapusan.
5. Pada panel navigasi, pilih Kebijakan.
6. Di kotak pencarian, masukkan nama kebijakan yang Anda buat untuk tutorial ini.
7. Pilih kebijakan untuk membuka halaman ringkasannya. Pada halaman Ringkasan kebijakan, pilih Hapus kebijakan.
8. Pilih Hapus.

Amazon MSK: Cara kerjanya

Cluster MSK Amazon adalah sumber daya MSK Amazon utama yang dapat Anda buat di akun Anda. Topik di bagian ini menjelaskan cara melakukan operasi MSK Amazon yang umum. Untuk daftar semua operasi yang dapat Anda lakukan pada klaster MSK, lihat berikut ini:

- Sebuah AWS Management Console
- Referensi API MSK Amazon
- Referensi Perintah Amazon MSK CLI

Topik

Membuat cluster MSK Amazon

> **Important**
>
> Anda tidak dapat mengubah VPC untuk klaster MSK Amazon setelah Anda membuat cluster.

Sebelum Anda dapat membuat kluster MSK Amazon, Anda harus memiliki Amazon Virtual Private Cloud (VPC) dan mengatur subnet dalam VPC itu.

Anda memerlukan dua subnet di dua Availability Zone yang berbeda di Wilayah AS Barat (California Utara). Di semua Wilayah lain di mana Amazon MSK tersedia, Anda dapat menentukan dua atau tiga subnet. Subnet Anda semua harus berada di Availability Zone yang berbeda. Saat Anda membuat cluster, Amazon MSK mendistribusikan node broker secara merata di atas subnet yang Anda tentukan.

Ukuran broker

Saat Anda membuat cluster MSK Amazon, Anda menentukan ukuran broker yang Anda inginkan. Amazon MSK mendukung ukuran broker berikut:

- kafka.t3.small
- kafka.m5.large, kafka.m5.xlarge, kafka.m5.2xlarge, kafka.m5.4xlarge, kafka.m5.8xlarge, kafka.m5.12xlarge, kafka.m5.16xlarge, kafka.m5.24xlarge
- kafka.m7g.large, kafka.m7g.xlarge, kafka.m7g.2xlarge, kafka.m7g.4xlarge, kafka.m7g.8xlarge, kafka.m7g.12xlarge, kafka.m7g.16xlarge

Broker M7g menggunakan prosesor AWS Graviton (prosesor berbasis ARM khusus yang dibuat oleh Amazon Web Services). Broker M7g menawarkan peningkatan kinerja harga relatif terhadap instance M5 yang sebanding. Pialang m7g mengkonsumsi daya lebih sedikit daripada instans M5 yang sebanding.

Broker Graviton M7g tidak tersedia di wilayah ini: CDG (Paris), CGK (Jakarta), CPT (Cape Town), DXB (Dubai), HKG (Hong Kong), KIX (Osaka), LHR (London), MEL (Melbourne), MXP (Milan), OSU (AS-Timur), PDT (AS-Barat), TLV (Tel Aviv), YYC (Calgary), ZRH (Zürich).

MSK mendukung broker M7g pada cluster yang menjalankan salah satu versi Kafka berikut:

- 2.8.2.berjenjang
- 3.3.2
- 3.4.0
- 3.5.1
- 3.6.0 dengan penyimpanan berjenjang

- 3.7.x
- 3.7.x.kraft

Broker M7g dan M5 memiliki kinerja throughput baseline yang lebih tinggi daripada broker T3 dan direkomendasikan untuk beban kerja produksi. Broker M7g dan M5 juga dapat memiliki lebih banyak partisi per broker daripada broker T3. Gunakan broker M7G atau M5 jika Anda menjalankan beban kerja tingkat produksi yang lebih besar atau memerlukan partisi yang lebih banyak. Untuk mempelajari lebih lanjut tentang ukuran instans M7g dan M5, lihat Instans Tujuan Umum Amazon EC2.

Broker T3 memiliki kemampuan untuk menggunakan kredit CPU untuk meningkatkan kinerja sementara. Gunakan broker T3 untuk pengembangan berbiaya rendah, jika Anda menguji beban kerja streaming kecil hingga menengah, atau jika Anda memiliki beban kerja streaming throughput rendah yang mengalami lonjakan sementara dalam throughput. Kami menyarankan Anda menjalankan proof-of-concept tes untuk menentukan apakah broker T3 cukup untuk produksi atau beban kerja kritis. Untuk mempelajari lebih lanjut tentang ukuran broker T3, lihat Instans Amazon EC2T3.

Untuk informasi lebih lanjut tentang cara memilih ukuran broker, lihatPraktik terbaik.

Membuat cluster menggunakan AWS Management Console

Proses ini menjelaskan tugas umum membuat klaster yang disediakan menggunakan opsi pembuatan kustom. Anda dapat memilih opsi lain di konsol MSK untuk membuat cluster tanpa server.

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Pilih Buat klaster.
3. Untuk metode pembuatan Cluster, pilih Custom create.
4. Tentukan nama Cluster yang unik dan tidak lebih dari 64 karakter.
5. Untuk tipe Cluster, pilih Provisioned, yang memungkinkan Anda menentukan jumlah broker, ukuran broker, dan kapasitas penyimpanan cluster.
6. Pilih versi Apache Kafka yang ingin Anda jalankan di broker. Untuk melihat perbandingan fitur MSK yang didukung oleh setiap versi Apache Kafka, pilih Lihat kompatibilitas versi.
7. Bergantung pada versi Apache Kafka yang Anda pilih, Anda mungkin memiliki opsi untuk memilih mode Metadata cluster: atau kRAFT. ZooKeeper

8. Pilih ukuran broker yang akan digunakan untuk cluster berdasarkan kebutuhan komputasi, memori, dan penyimpanan cluster. Lihat???,
9. Pilih Jumlah zona di mana broker didistribusikan.
10. Tentukan jumlah broker yang ingin dibuat MSK di setiap Availability Zone. Minimal adalah satu broker per Availability Zone dan maksimum adalah 30 broker per cluster untuk cluster ZooKeeper berbasis dan 60 broker per cluster untuk cluster berbasis Kraft.
11. Pilih jumlah awal Storage yang Anda inginkan untuk dimiliki klaster Anda. Anda tidak dapat mengurangi kapasitas penyimpanan setelah membuat cluster.
12. Bergantung pada ukuran broker (ukuran instans) yang Anda pilih, Anda dapat menentukan throughput penyimpanan yang disediakan per broker. Untuk mengaktifkan opsi ini, pilih ukuran broker (ukuran instans) kafka.m5.4xlarge atau lebih besar untuk x86, dan kafka.m7g.2xlarge atau lebih besar untuk instance berbasis Graviton. Lihat ???.
13. Pilih opsi mode penyimpanan Cluster, baik penyimpanan EBS saja atau penyimpanan berjenjang dan penyimpanan EBS.
14. Jika Anda ingin membuat dan menggunakan konfigurasi Cluster kustom (atau jika Anda sudah memiliki konfigurasi cluster yang disimpan), pilih konfigurasi. Jika tidak, Anda dapat membuat cluster menggunakan konfigurasi cluster default Amazon MSK. Untuk informasi tentang konfigurasi MSK Amazon, lihat. Konfigurasi
15. Pilih Selanjutnya.
16. Untuk pengaturan Jaringan, pilih VPC yang ingin Anda gunakan untuk cluster.
17. Berdasarkan Jumlah zona yang Anda pilih sebelumnya, tentukan Availability Zone dan subnet tempat broker akan digunakan. Subnet harus berada di Availability Zone yang berbeda.
18. Anda dapat memilih satu atau beberapa grup keamanan yang ingin Anda berikan akses ke klaster Anda (misalnya, grup keamanan mesin klien). Jika Anda menentukan grup keamanan yang dibagikan dengan Anda, Anda harus memastikan bahwa Anda memiliki izin untuk menggunakannya. Secara khusus, Anda memerlukan izin `ec2:DescribeSecurityGroups`. Menghubungkan ke cluster MSK Amazon.
19. Pilih Selanjutnya.
20. Pilih metode kontrol Akses kluster dan pengaturan Enkripsi untuk mengenkripsi data saat transit antara klien dan broker. Untuk informasi selengkapnya, lihat the section called “Enkripsi bergerak”.
21. Pilih jenis kunci KMS yang ingin Anda gunakan untuk mengenkripsi data saat istirahat. Untuk informasi selengkapnya, lihat the section called “Enkripsi diam”.
22. Pilih Selanjutnya.

23. Pilih Monitoring dan tag yang Anda inginkan. Ini menentukan set metrik yang Anda dapatkan. Untuk informasi selengkapnya, lihat Memantau klaster. Amazon CloudWatch, Prometheus, Pengiriman log Broker, atau tag Cluster, lalu pilih Berikutnya.
24. Tinjau pengaturan untuk cluster Anda. Anda dapat kembali dan mengubah pengaturan dengan memilih Sebelumnya untuk kembali ke layar konsol sebelumnya, atau Edit untuk mengubah pengaturan cluster tertentu. Jika pengaturannya benar, pilih Buat cluster.
25. Periksa Status cluster pada halaman ringkasan Cluster. Status berubah dari Creating menjadi Active karena Amazon MSK menyediakan klaster. Ketika statusnya Aktif, Anda dapat terhubung ke cluster. Untuk informasi selengkapnya tentang status klaster, lihatNegara cluster.

Membuat cluster menggunakan AWS CLI

1. Salin JSON berikut dan simpan ke file. Beri nama file`brokernodegroupinfo.json`. Ganti ID subnet di JSON dengan nilai yang sesuai dengan subnet Anda. Subnet ini harus berada di Availability Zone yang berbeda. Ganti *“Security-Group-ID”* dengan ID dari satu atau lebih grup keamanan VPC klien. Klien yang terkait dengan grup keamanan ini mendapatkan akses ke cluster. Jika Anda menentukan grup keamanan yang dibagikan untuk Anda, Anda harus memastikan bahwa Anda memiliki izin untuknya. Secara khusus, Anda memerlukan izin `ec2:DescribeSecurityGroups`. Sebagai contoh, lihat Amazon EC2: Mengizinkan Mengelola Grup Keamanan Amazon EC2 yang Terkait Dengan VPC Tertentu, Secara Terprogram, dan di Konsol. Terakhir, simpan file JSON yang diperbarui di komputer tempat Anda AWS CLI menginstal.

```
{
  "InstanceType": "kafka.m5.large",
  "ClientSubnets": [
    "Subnet-1-ID",
    "Subnet-2-ID"
  ],
  "SecurityGroups": [
    "Security-Group-ID"
  ]
}
```

> **Important**
>
> Tentukan tepat dua subnet jika Anda menggunakan Wilayah US West (N. California). Untuk Wilayah lain tempat Amazon MSK tersedia, Anda dapat menentukan dua atau tiga subnet. Subnet yang Anda tentukan harus berada di Availability Zone yang berbeda. Ketika Anda membuat klaster, Amazon MSK mendistribusikan simpul broker secara merata di seluruh subnet yang Anda tentukan.

2. Jalankan AWS CLI perintah berikut di direktori tempat Anda menyimpan `brokernodegroupinfo.json` file, ganti *"Your-Cluster-Name" dengan nama pilihan* Anda. Untuk *"Monitoring-Level"*, Anda dapat menentukan salah satu dari tiga nilai berikut:`DEFAULT`,, `PER_BROKER` atau. `PER_TOPIC_PER_BROKER` Untuk informasi tentang tiga tingkat pemantauan yang berbeda ini, lihat???. Parameter `enhanced-monitoring` bersifat opsional. Jika Anda tidak menentukannya dalam `create-cluster` perintah, Anda mendapatkan `DEFAULT` tingkat pemantauan.

```
aws kafka create-cluster --cluster-name "Your-Cluster-Name" --broker-node-group-
info file://brokernodegroupinfo.json --kafka-version "2.8.1" --number-of-broker-
nodes 3 --enhanced-monitoring "Monitoring-Level"
```

Output dari perintah terlihat seperti JSON berikut:

```
{
    "ClusterArn": "...",
    "ClusterName": "AWSKafkaTutorialCluster",
    "State": "CREATING"
}
```

> **Note**
>
> `create-cluster`Perintah mungkin menampilkan kesalahan yang menyatakan bahwa satu atau beberapa subnet milik Availability Zone yang tidak didukung. Ketika ini terjadi, kesalahan menunjukkan Availability Zone mana yang tidak didukung. Buat subnet yang tidak menggunakan Availability Zone yang tidak didukung dan coba `create-cluster` perintahnya lagi.

3. Simpan nilai `ClusterArn` kunci karena Anda membutuhkannya untuk melakukan tindakan lain di cluster Anda.
4. Jalankan perintah berikut untuk memeriksa cluster Anda`STATE`. `STATE`Nilai berubah dari `CREATING` menjadi `ACTIVE` saat Amazon MSK menyediakan cluster. Ketika statusnya`ACTIVE`, Anda dapat terhubung ke cluster. Untuk informasi selengkapnya tentang status klaster, lihatNegara cluster.

```
aws kafka describe-cluster --cluster-arn <your-cluster-ARN>
```

Membuat cluster dengan konfigurasi MSK Amazon khusus menggunakan AWS CLI

Untuk informasi tentang konfigurasi MSK Amazon kustom dan cara membuatnya, lihat. Konfigurasi

1. Simpan JSON berikut ke file, ganti *`configuration-arn dengan`* ARN dari konfigurasi yang ingin Anda gunakan untuk membuat cluster.

```
{
    "Arn": configuration-arn,
    "Revision": 1
}
```

2. Jalankan `create-cluster` perintah dan gunakan `configuration-info` opsi untuk menunjuk ke file JSON yang Anda simpan di langkah sebelumnya. Berikut adalah contohnya.

```
aws kafka create-cluster --cluster-name ExampleClusterName --broker-node-group-
info file://brokernodegroupinfo.json --kafka-version "2.8.1" --number-of-broker-
nodes 3 --enhanced-monitoring PER_TOPIC_PER_BROKER --configuration-info file://
configuration.json
```

Berikut ini adalah contoh respons yang berhasil setelah menjalankan perintah ini.

```
{
    "ClusterArn": "arn:aws:kafka:us-east-1:123456789012:cluster/
CustomConfigExampleCluster/abcd1234-abcd-dcba-4321-a1b2abcd9f9f-2",
    "ClusterName": "CustomConfigExampleCluster",
    "State": "CREATING"
```

```
}
```

Membuat cluster menggunakan API

Untuk membuat cluster menggunakan API, lihat CreateCluster.

Menghapus kluster MSK Amazon

> **Note**
>
> Jika klaster Anda memiliki kebijakan auto-scaling, sebaiknya hapus kebijakan tersebut sebelum menghapus klaster. Untuk informasi selengkapnya, lihat Penskalaan Otomatis.

Menghapus cluster menggunakan AWS Management Console

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Pilih klaster MSK yang ingin Anda hapus dengan memilih kotak centang di sebelahnya.
3. Pilih Hapus, lalu konfirmasikan penghapusan.

Menghapus cluster menggunakan AWS CLI

Jalankan perintah berikut, ganti *ClusterArn*dengan Amazon Resource Name (ARN) yang Anda peroleh saat membuat cluster Anda. Jika Anda tidak memiliki ARN untuk cluster Anda, Anda dapat menemukannya dengan mencantumkan semua cluster. Untuk informasi selengkapnya, lihat the section called “Daftar cluster”.

```
aws kafka delete-cluster --cluster-arn ClusterArn
```

Menghapus klaster menggunakan API

Untuk menghapus klaster menggunakan API, lihat DeleteCluster.

Mendapatkan broker bootstrap untuk cluster MSK Amazon

Mendapatkan broker bootstrap menggunakan AWS Management Console

Istilah broker bootstrap mengacu pada daftar broker yang dapat digunakan klien Apache Kafka sebagai titik awal untuk terhubung ke cluster. Daftar ini tidak selalu mencakup semua broker dalam sebuah cluster.

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Tabel menunjukkan semua cluster untuk wilayah saat ini di bawah akun ini. Pilih nama cluster untuk melihat deskripsinya.
3. Pada halaman ringkasan Cluster, pilih Lihat informasi klien. Ini menunjukkan kepada Anda broker bootstrap, serta string ZooKeeper koneksi Apache.

Mendapatkan broker bootstrap menggunakan AWS CLI

Jalankan perintah berikut, ganti *ClusterArn*dengan Amazon Resource Name (ARN) yang Anda peroleh saat membuat cluster Anda. Jika Anda tidak memiliki ARN untuk cluster Anda, Anda dapat menemukannya dengan mencantumkan semua cluster. Untuk informasi selengkapnya, lihat the section called “Daftar cluster”.

```
aws kafka get-bootstrap-brokers --cluster-arn ClusterArn
```

Untuk cluster MSK yang menggunakanthe section called “Kontrol akses IAM”, output dari perintah ini terlihat seperti contoh JSON berikut.

```
{
    "BootstrapBrokerStringSaslIam": "b-1.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9098,b-2.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9098"
}
```

Contoh berikut menunjukkan broker bootstrap untuk cluster yang memiliki akses publik diaktifkan. Gunakan `BootstrapBrokerStringPublicSaslIam` untuk akses publik, dan `BootstrapBrokerStringSaslIam` string untuk akses dari dalam AWS.

```
{
```

```
    "BootstrapBrokerStringPublicSaslIam": "b-2-public.myTestCluster.v4ni96.c2.kafka-
beta.us-east-1.amazonaws.com:9198,b-1-public.myTestCluster.v4ni96.c2.kafka-
beta.us-east-1.amazonaws.com:9198,b-3-public.myTestCluster.v4ni96.c2.kafka-beta.us-
east-1.amazonaws.com:9198",
    "BootstrapBrokerStringSaslIam": "b-2.myTestCluster.v4ni96.c2.kafka-
beta.us-east-1.amazonaws.com:9098,b-1.myTestCluster.v4ni96.c2.kafka-beta.us-
east-1.amazonaws.com:9098,b-3.myTestCluster.v4ni96.c2.kafka-beta.us-
east-1.amazonaws.com:9098"
}
```

String broker bootstrap harus berisi tiga broker dari seluruh Availability Zone di mana cluster MSK Anda digunakan (kecuali hanya dua broker yang tersedia).

Mendapatkan broker bootstrap menggunakan API

Untuk mendapatkan broker bootstrap menggunakan API, lihat GetBootstrapBroker.

Daftar kluster MSK Amazon

Daftar cluster menggunakan AWS Management Console

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Tabel menunjukkan semua cluster untuk wilayah saat ini di bawah akun ini. Pilih nama cluster untuk melihat detailnya.

Daftar cluster menggunakan AWS CLI

Jalankan perintah berikut.

```
aws kafka list-clusters
```

Membuat daftar cluster menggunakan API

Untuk membuat daftar cluster menggunakan API, lihat ListClusters.

Manajemen metadata

Amazon MSK mendukung mode manajemen metadata Apache ZooKeeper atau kRAFT.

Dari Apache Kafka versi 3.7.x di Amazon MSK, Anda dapat membuat cluster yang menggunakan mode Kraft alih-alih mode. ZooKeeper Cluster berbasis Kraft mengandalkan pengontrol dalam Kafka untuk mengelola metadata.

Topik

ZooKeeper modus

Apache ZooKeeper adalah layanan terpusat untuk memelihara informasi konfigurasi, penamaan, menyediakan sinkronisasi terdistribusi, dan menyediakan layanan grup. Semua jenis layanan ini digunakan dalam beberapa bentuk atau lainnya oleh aplikasi terdistribusi," termasuk Apache Kafka.

Jika cluster Anda menggunakan ZooKeeper mode, Anda dapat menggunakan langkah-langkah di bawah ini untuk mendapatkan string ZooKeeper koneksi Apache. Namun, kami menyarankan Anda menggunakan `BootstrapServerString` untuk terhubung ke operasi admin cluster dan perfom Anda karena `--zookeeper` bendera telah tidak digunakan lagi di Kafka 2.5 dan dihapus dari Kafka 3.0.

Mendapatkan string ZooKeeper koneksi Apache menggunakan AWS Management Console

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Tabel menunjukkan semua cluster untuk wilayah saat ini di bawah akun ini. Pilih nama cluster untuk melihat deskripsinya.
3. Pada halaman ringkasan Cluster, pilih Lihat informasi klien. Ini menunjukkan kepada Anda broker bootstrap, serta string ZooKeeper koneksi Apache.

Mendapatkan string ZooKeeper koneksi Apache menggunakan AWS CLI

1. Jika Anda tidak mengetahui Nama Sumber Daya Amazon (ARN) klaster Anda, Anda dapat menemukannya dengan mencantumkan semua cluster di akun Anda. Untuk informasi selengkapnya, lihat the section called "Daftar cluster".
2. Untuk mendapatkan string ZooKeeper koneksi Apache, bersama dengan informasi lain tentang cluster Anda, jalankan perintah berikut, ganti *`ClusterArn`*dengan ARN cluster Anda.

```
aws kafka describe-cluster --cluster-arn ClusterArn
```

Output dari `describe-cluster` perintah ini terlihat seperti contoh JSON berikut.

```
{
    "ClusterInfo": {
        "BrokerNodeGroupInfo": {
            "BrokerAZDistribution": "DEFAULT",
            "ClientSubnets": [
                "subnet-0123456789abcdef0",
                "subnet-2468013579abcdef1",
                "subnet-1357902468abcdef2"
            ],
            "InstanceType": "kafka.m5.large",
            "StorageInfo": {
                "EbsStorageInfo": {
                    "VolumeSize": 1000
                }
            }
        },
        "ClusterArn": "arn:aws:kafka:us-east-1:111122223333:cluster/
testcluster/12345678-abcd-4567-2345-abcdef123456-2",
        "ClusterName": "testcluster",
        "CreationTime": "2018-12-02T17:38:36.75Z",
        "CurrentBrokerSoftwareInfo": {
            "KafkaVersion": "2.2.1"
        },
        "CurrentVersion": "K13V1IB3VIYZZH",
        "EncryptionInfo": {
            "EncryptionAtRest": {
                "DataVolumeKMSKeyId": "arn:aws:kms:us-
east-1:555555555555:key/12345678-abcd-2345-ef01-abcdef123456"
            }
        },
        "EnhancedMonitoring": "DEFAULT",
        "NumberOfBrokerNodes": 3,
        "State": "ACTIVE",
        "ZookeeperConnectString": "10.0.1.101:2018,10.0.2.101:2018,10.0.3.101:2018"
    }
}
```

Contoh JSON sebelumnya menunjukkan `ZookeeperConnectString` kunci dalam output `describe-cluster` perintah. Salin nilai yang sesuai dengan kunci ini dan simpan untuk saat Anda perlu membuat topik di cluster Anda.

> **Important**
>
> Cluster MSK Amazon Anda harus dalam `ACTIVE` keadaan agar Anda dapat memperoleh string ZooKeeper koneksi Apache. Ketika sebuah cluster masih dalam `CREATING` status, output dari `describe-cluster` perintah tidak termasuk`ZookeeperConnectString`. Jika ini masalahnya, tunggu beberapa menit dan kemudian jalankan `describe-cluster` lagi setelah cluster Anda mencapai `ACTIVE` status.

Mendapatkan string ZooKeeper koneksi Apache menggunakan API

Untuk mendapatkan string ZooKeeper koneksi Apache menggunakan API, lihat DescribeCluster.

modus kRAFT

Amazon MSK memperkenalkan dukungan untuk kRAFT (Apache Kafka Raft) di Kafka versi 3.7.x. Komunitas Apache Kafka mengembangkan kRAFT untuk menggantikan Apache ZooKeeper untuk manajemen metadata di cluster Apache Kafka. Dalam mode kRAFT, metadata cluster disebarkan dalam sekelompok pengendali Kafka, yang merupakan bagian dari cluster Kafka, bukan di seluruh node. ZooKeeper Pengontrol Kraft disertakan tanpa biaya tambahan untuk Anda, dan tidak memerlukan pengaturan atau manajemen tambahan dari Anda. Lihat KIP-500 untuk informasi lebih lanjut tentang kRAFT.

Berikut adalah beberapa poin yang perlu diperhatikan tentang mode kRAFT di MSK:

- Mode kRAFT hanya tersedia untuk cluster baru. Anda tidak dapat mengganti mode metadata setelah cluster dibuat.
- Di konsol MSK, Anda dapat membuat cluster berbasis Kraft dengan memilih Kafka versi 3.7.x dan memilih kotak centang Kraft di jendela pembuatan cluster.
- Untuk membuat cluster dalam mode kRAFT menggunakan MSK API `CreateCluster`atau `CreateClusterV2`operasi, Anda harus menggunakan `3.7.x.kraft` sebagai versi. Gunakan `3.7.x` sebagai versi untuk membuat cluster dalam ZooKeeper mode.

- Jumlah partisi per broker sama pada kRAFT dan cluster ZooKeeper berbasis. Namun, Kraft memungkinkan Anda untuk meng-host lebih banyak partisi per cluster dengan menyediakan lebih banyak broker dalam sebuah cluster.
- Tidak ada perubahan API yang diperlukan untuk menggunakan mode kRAFT di Amazon MSK. Namun, jika klien Anda masih menggunakan string `--zookeeper` koneksi hari ini, Anda harus memperbarui klien Anda untuk menggunakan string `--bootstrap-server` koneksi untuk terhubung ke cluster Anda. `--zookeeper`Bendera tidak digunakan lagi di Apache Kafka versi 2.5 dan dihapus dimulai dengan Kafka versi 3.0. Oleh karena itu kami menyarankan Anda menggunakan versi klien Apache Kafka terbaru dan string `--bootstrap-server` koneksi untuk semua koneksi ke cluster Anda.
- ZooKeeper mode terus tersedia untuk semua versi yang dirilis di mana zookeeper juga didukung oleh Apache Kafka. Lihat Versi Apache Kafka yang didukung detail tentang akhir dukungan untuk versi Apache Kafka dan pembaruan masa depan.
- Anda harus memeriksa apakah alat apa pun yang Anda gunakan mampu menggunakan Kafka Admin API tanpa ZooKeeper koneksi. Lihat langkah-langkah terbaru Menggunakan Cruise LinkedIn Control untuk Apache Kafka dengan Amazon MSK untuk menghubungkan cluster Anda ke Cruise Control. Cruise Control juga memiliki instruksi untuk menjalankan Cruise Control tanpa ZooKeeper.
- Anda tidak perlu mengakses pengontrol kRAFT cluster Anda secara langsung untuk tindakan administratif apa pun. Namun, jika Anda menggunakan pemantauan terbuka untuk mengumpulkan metrik, Anda juga memerlukan titik akhir DNS dari pengontrol Anda untuk mengumpulkan beberapa metrik terkait non-pengontrol tentang klaster Anda. Anda bisa mendapatkan endpoint DNS ini dari MSK Console atau menggunakan operasi API. ListNodes Lihat langkah-langkah terbaru Pemantauan terbuka dengan Prometheus untuk menyiapkan pemantauan terbuka untuk kluster berbasis Kraft.
- Tidak ada CloudWatch metrik tambahan yang perlu Anda pantau untuk kluster mode kRAFT melalui ZooKeeper kluster mode. MSK mengelola kontroler kRAFT yang digunakan dalam cluster Anda.
- Anda dapat terus mengelola ACL menggunakan kluster mode kRAFT menggunakan string koneksi. `--bootstrap-server` Anda tidak boleh menggunakan string `--zookeeper` koneksi untuk mengelola ACL. Lihat Apache Kafka ACL.
- Dalam mode kRAFT, metadata cluster Anda disimpan pada pengontrol kRAFT dalam Kafka dan bukan node eksternal. ZooKeeper Oleh karena itu, Anda tidak perlu mengontrol akses ke node pengontrol secara terpisah seperti yang Anda lakukan dengan ZooKeeper node.

Manajemen penyimpanan

Amazon MSK menyediakan fitur untuk membantu Anda dengan manajemen penyimpanan di kluster MSK Anda.

Topik

Penyimpanan berjenjang

Penyimpanan berjenjang adalah tingkat penyimpanan berbiaya rendah untuk MSK Amazon yang diskalakan ke penyimpanan yang hampir tidak terbatas, sehingga hemat biaya untuk membangun aplikasi data streaming.

Anda dapat membuat kluster MSK Amazon yang dikonfigurasi dengan penyimpanan berjenjang yang menyeimbangkan kinerja dan biaya. Amazon MSK menyimpan data streaming dalam tingkat penyimpanan utama yang dioptimalkan kinerja hingga mencapai batas retensi topik Apache Kafka. Kemudian, Amazon MSK secara otomatis memindahkan data ke tingkat penyimpanan berbiaya rendah yang baru.

Saat aplikasi Anda mulai membaca data dari penyimpanan berjenjang, Anda dapat mengharapkan peningkatan latensi baca untuk beberapa byte pertama. Saat Anda mulai membaca data yang tersisa secara berurutan dari tingkat berbiaya rendah, Anda dapat mengharapkan latensi yang mirip dengan tingkat penyimpanan utama. Anda tidak perlu menyediakan penyimpanan apa pun untuk penyimpanan berjenjang berbiaya rendah atau mengelola infrastruktur. Anda dapat menyimpan sejumlah data dan hanya membayar untuk apa yang Anda gunakan. Fitur ini kompatibel dengan API yang diperkenalkan di KIP-405: Kafka Tiered Storage.

Berikut adalah beberapa fitur penyimpanan berjenjang:

- Anda dapat menskalakan ke penyimpanan yang hampir tidak terbatas. Anda tidak perlu menebak bagaimana menskalakan infrastruktur Apache Kafka Anda.
- Anda dapat menyimpan data lebih lama di topik Apache Kafka Anda, atau meningkatkan penyimpanan topik Anda, tanpa perlu menambah jumlah broker.

- Ini menyediakan buffer keamanan durasi yang lebih lama untuk menangani penundaan pemrosesan yang tidak terduga.
- Anda dapat memproses ulang data lama dalam urutan produksi yang tepat dengan kode pemrosesan aliran yang ada dan API Kafka.
- Partisi menyeimbangkan kembali lebih cepat karena data pada penyimpanan sekunder tidak memerlukan replikasi di seluruh disk broker.
- Data antara broker dan penyimpanan berjenjang bergerak di dalam VPC dan tidak melakukan perjalanan melalui internet.
- Mesin klien dapat menggunakan proses yang sama untuk terhubung ke cluster baru dengan penyimpanan berjenjang diaktifkan seperti halnya untuk terhubung ke cluster tanpa penyimpanan berjenjang diaktifkan. Lihat Membuat mesin klien.

Persyaratan penyimpanan berjenjang

- Anda harus menggunakan klien Apache Kafka versi 3.0.0 atau lebih tinggi untuk membuat topik baru dengan penyimpanan berjenjang diaktifkan. Untuk mentransisikan topik yang ada ke penyimpanan berjenjang, Anda dapat mengonfigurasi ulang mesin klien yang menggunakan versi klien Kafka yang lebih rendah dari 3.0.0 (versi Apache Kafka minimum yang didukung adalah 2.8.2.tiered) untuk mengaktifkan penyimpanan berjenjang. Lihat Langkah 4: Buat topik.
- Cluster MSK Amazon dengan penyimpanan berjenjang yang diaktifkan harus menggunakan versi 3.6.0 atau lebih tinggi, atau 2.8.2.tiered.

Kendala dan batasan penyimpanan berjenjang

Penyimpanan berjenjang memiliki kendala dan batasan berikut:

- Penyimpanan berjenjang hanya berlaku untuk cluster mode yang disediakan.
- Penyimpanan berjenjang tidak mendukung ukuran broker t3.small.
- Periode retensi minimum dalam penyimpanan berbiaya rendah adalah 3 hari. Tidak ada periode retensi minimum untuk penyimpanan primer.
- Penyimpanan berjenjang tidak mendukung direktori Multiple Log pada broker (fitur terkait JBOD).
- Penyimpanan berjenjang tidak mendukung topik yang dipadatkan. Pastikan bahwa semua topik yang telah mengaktifkan penyimpanan berjenjang memiliki cleanup.policy yang dikonfigurasi menjadi 'HAPUS' saja.

- Penyimpanan Berjenjang dapat dinonaktifkan untuk topik individual tetapi tidak untuk seluruh cluster. Setelah dinonaktifkan, penyimpanan berjenjang tidak dapat diaktifkan kembali untuk suatu topik.
- Jika Anda menggunakan Amazon MSK versi 2.8.2.tiered, Anda hanya dapat bermigrasi ke versi Apache Kafka yang didukung penyimpanan berjenjang lainnya. Jika Anda tidak ingin terus menggunakan versi yang didukung penyimpanan berjenjang, buat klaster MSK baru dan migrasi data Anda ke sana.
- kafka-log-dirs Alat ini tidak dapat melaporkan ukuran data penyimpanan berjenjang. Alat ini hanya melaporkan ukuran segmen log di penyimpanan primer.

Bagaimana segmen log disalin ke penyimpanan berjenjang

Saat Anda mengaktifkan penyimpanan berjenjang untuk topik baru atau yang sudah ada, Apache Kafka menyalin segmen log tertutup dari penyimpanan utama ke penyimpanan berjenjang.

- Apache Kafka hanya menyalin segmen log tertutup. Ini menyalin semua pesan dalam segmen log ke penyimpanan berjenjang.
- Segmen aktif tidak memenuhi syarat untuk tiering. Ukuran segmen log (segment.bytes) atau waktu roll segmen (segment.ms) mengontrol tingkat penutupan segmen, dan tingkat Apache Kafka kemudian menyalinnya ke penyimpanan berjenjang.

Pengaturan retensi untuk topik dengan penyimpanan berjenjang diaktifkan berbeda dari pengaturan untuk topik tanpa penyimpanan berjenjang diaktifkan. Aturan berikut mengontrol penyimpanan pesan dalam topik dengan penyimpanan berjenjang diaktifkan:

- Anda menentukan retensi di Apache Kafka dengan dua pengaturan: log.retention.ms (waktu) dan log.retention.bytes (ukuran). Pengaturan ini menentukan total durasi dan ukuran data yang dipertahankan Apache Kafka di cluster. Apakah Anda mengaktifkan mode penyimpanan berjenjang atau tidak, Anda mengatur konfigurasi ini di tingkat cluster. Anda dapat mengganti pengaturan di tingkat topik dengan konfigurasi topik.
- Saat Anda mengaktifkan penyimpanan berjenjang, Anda juga dapat menentukan berapa lama tingkat penyimpanan berkinerja tinggi utama menyimpan data. Misalnya, jika topik memiliki pengaturan retensi keseluruhan (log.retention.ms) 7 hari dan retensi lokal (local.retention.ms) 12 jam, maka penyimpanan utama klaster menyimpan data hanya selama 12 jam pertama. Tingkat penyimpanan berbiaya rendah mempertahankan data selama 7 hari penuh.

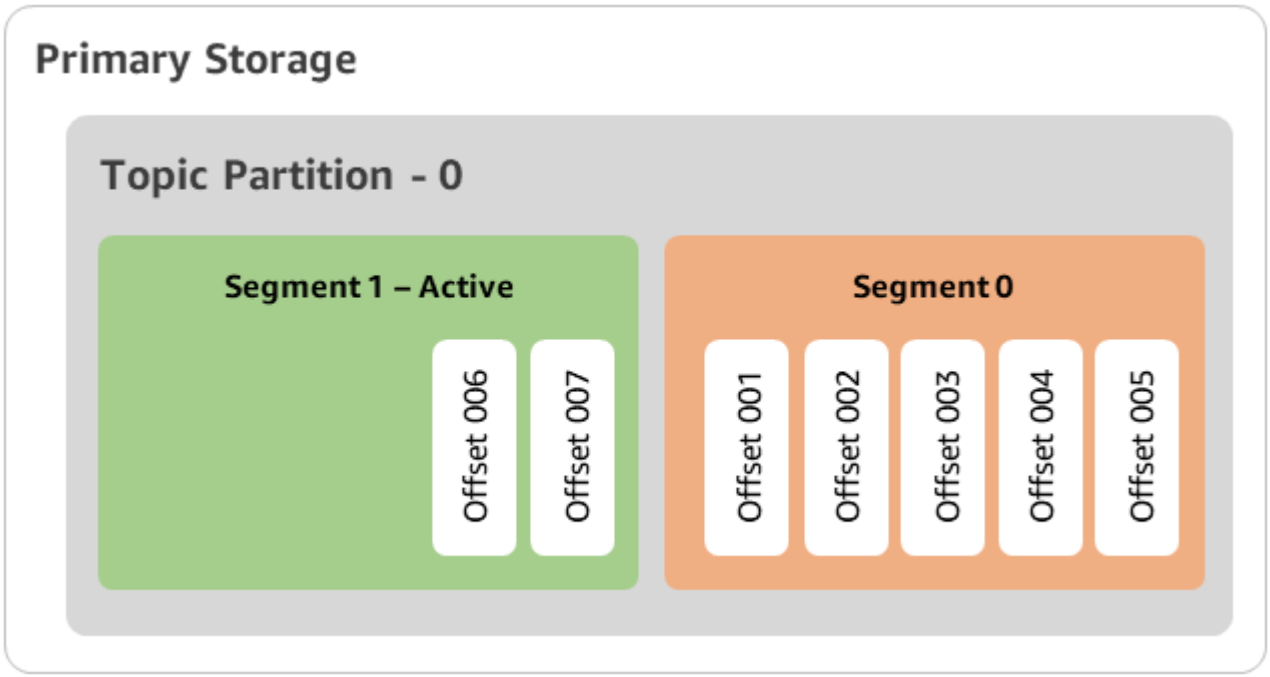
- Pengaturan retensi yang biasa berlaku untuk log lengkap. Ini termasuk bagian berjenjang dan utamanya.
- Setelan local.retention.ms atau local.retention.bytes mengontrol retensi pesan di penyimpanan utama. Ketika data telah mencapai ambang pengaturan penyimpanan penyimpanan primer (local.retention.ms/bytes) pada log penuh, Apache Kafka menyalin data dalam penyimpanan primer ke penyimpanan berjenjang. Data tersebut kemudian memenuhi syarat untuk kedaluwarsa.
- Ketika Apache Kafka menyalin pesan di segmen log ke penyimpanan berjenjang, itu akan menghapus pesan dari cluster berdasarkan pengaturan retention.ms atau retention.bytes.

Contoh skenario penyimpanan berjenjang

Skenario ini menggambarkan bagaimana topik yang ada yang memiliki pesan di penyimpanan utama berperilaku saat penyimpanan berjenjang diaktifkan. Anda mengaktifkan penyimpanan berjenjang pada topik ini saat Anda menyetel remote.storage.enable ke. `true` Dalam contoh ini, retention.ms diatur ke 5 hari dan local.retention.ms diatur ke 2 hari. Berikut ini adalah urutan peristiwa ketika segmen kedaluwarsa.

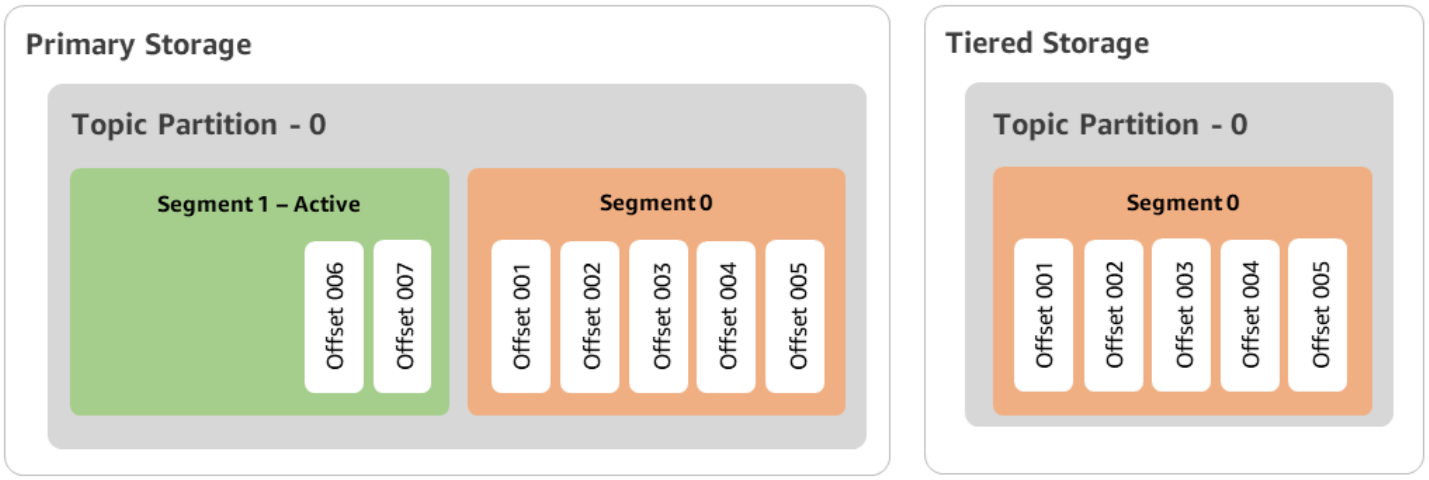
Time T0 - Sebelum Anda mengaktifkan penyimpanan berjenjang.

Sebelum Anda mengaktifkan penyimpanan berjenjang untuk topik ini, ada dua segmen log. Salah satu segmen aktif untuk partisi topik yang ada 0.



Waktu T1 (< 2 hari) - Penyimpanan berjenjang diaktifkan. Segmen 0 disalin ke penyimpanan berjenjang.

Setelah Anda mengaktifkan penyimpanan berjenjang untuk topik ini, Apache Kafka menyalin segmen log 0 ke penyimpanan berjenjang setelah segmen memenuhi pengaturan retensi awal. Apache Kafka juga mempertahankan salinan penyimpanan utama segmen 0. Segmen aktif 1 belum memenuhi syarat untuk menyalin ke penyimpanan berjenjang. Dalam timeline ini, Amazon MSK belum menerapkan pengaturan rerensi ulang apa pun untuk pesan apa pun di segmen 0 dan segmen 1. (local.retention.bytes/ms, retensi.ms/byte)



Waktu T2 - Retensi lokal berlaku.

Setelah 2 hari, pengaturan retensi primer berlaku untuk segmen 0 yang disalin Apache Kafka ke penyimpanan berjenjang. Pengaturan local.retention.ms sebagai 2 hari menentukan ini. Segmen 0 sekarang kedaluwarsa dari penyimpanan utama. Segmen aktif 1 belum memenuhi syarat untuk kedaluwarsa atau memenuhi syarat untuk menyalin ke penyimpanan berjenjang.

Waktu T3 - Retensi keseluruhan berlaku.

Setelah 5 hari, pengaturan retensi mulai berlaku, dan Kafka menghapus segmen log 0 dan pesan terkait dari penyimpanan berjenjang. Segmen 1 belum memenuhi syarat untuk kedaluwarsa atau memenuhi syarat untuk menyalin ke penyimpanan berjenjang karena aktif. Segmen 1 belum ditutup, sehingga tidak memenuhi syarat untuk roll segmen.

Membuat cluster MSK Amazon dengan penyimpanan berjenjang dengan AWS Management Console

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Pilih Buat klaster.
3. Pilih Custom create untuk penyimpanan berjenjang.
4. Tentukan nama untuk cluster.
5. Pada tipe Cluster, pilih Provisioned.
6. Pilih versi Amazon Kafka yang mendukung penyimpanan berjenjang untuk Amazon MSK untuk digunakan untuk membuat cluster.
7. Tentukan ukuran broker selain kafka.t3.small.
8. Pilih jumlah broker yang ingin Anda buat Amazon MSK di setiap Availability Zone. Minimal adalah satu broker per Availability Zone, dan maksimum adalah 30 broker per cluster.
9. Tentukan jumlah zona yang didistribusikan oleh broker.
10. Tentukan jumlah broker Apache Kafka yang digunakan per zona.
11. Pilih opsi Penyimpanan. Ini termasuk penyimpanan berjenjang dan penyimpanan EBS untuk mengaktifkan mode penyimpanan berjenjang.

12. Ikuti langkah-langkah yang tersisa di wizard pembuatan cluster. Saat selesai, Penyimpanan berjenjang dan penyimpanan EBS muncul sebagai mode penyimpanan cluster dalam tampilan Review and create.
13. Pilih Buat klaster.

Membuat cluster MSK Amazon dengan penyimpanan berjenjang dengan AWS CLI

Untuk mengaktifkan penyimpanan berjenjang pada cluster, buat cluster dengan versi Apache Kafka yang benar dan atribut untuk penyimpanan berjenjang. Ikuti contoh kode di bawah ini. Juga, selesaikan langkah-langkah di bagian selanjutnya untukMembuat topik Kafka dengan penyimpanan berjenjang diaktifkan.

Lihat create-cluster untuk daftar lengkap atribut yang didukung untuk pembuatan klaster.

```
aws tiered-storage create-cluster \
 –cluster-name "MessagingCluster" \
 –broker-node-group-info file://brokernodegroupinfo.json \
 –number-of-broker-nodes 3 \
--kafka-version "3.6.0" \
--storage-mode "TIERED"
```

Membuat topik Kafka dengan penyimpanan berjenjang diaktifkan

Untuk menyelesaikan proses yang Anda mulai saat membuat klaster dengan penyimpanan berjenjang diaktifkan, buat juga topik dengan penyimpanan berjenjang yang diaktifkan dengan atribut dalam contoh kode selanjutnya. Atribut khusus untuk penyimpanan berjenjang adalah sebagai berikut:

- `local.retention.ms`(misalnya, 10 menit) untuk pengaturan retensi berbasis waktu atau `local.retention.bytes` untuk batas ukuran segmen log.
- `remote.storage.enable`disetel ke `true` untuk mengaktifkan penyimpanan berjenjang.

Konfigurasi berikut menggunakan local.retention.ms, tetapi Anda dapat mengganti atribut ini dengan local.retention.bytes. Atribut ini mengontrol jumlah waktu yang dapat dilewati atau jumlah byte yang dapat disalin Apache Kafka sebelum Apache Kafka menyalin data dari penyimpanan primer ke penyimpanan berjenjang. Lihat Konfigurasi tingkat topik untuk detail selengkapnya tentang atribut konfigurasi yang didukung.

> **Note**
>
> Anda harus menggunakan klien Apache Kafka versi 3.0.0 dan di atasnya. Versi ini mendukung pengaturan yang disebut `remote.storage.enable` hanya dalam versi klien tersebut`kafka-topics.sh`. Untuk mengaktifkan penyimpanan berjenjang pada topik yang ada yang menggunakan versi Apache Kafka sebelumnya, lihat bagian. Mengaktifkan penyimpanan berjenjang pada topik yang ada

```
bin/kafka-topics.sh --create --bootstrap-server $bs --replication-factor 2
 --partitions 6 --topic MSKTutorialTopic --config remote.storage.enable=true
 --config local.retention.ms=100000 --config retention.ms=604800000 --config
 segment.bytes=134217728
```

Mengaktifkan dan menonaktifkan penyimpanan berjenjang pada topik yang ada

Bagian ini mencakup cara mengaktifkan dan menonaktifkan penyimpanan berjenjang pada topik yang telah Anda buat. Untuk membuat klaster dan topik baru dengan penyimpanan berjenjang diaktifkan, lihat Membuat klaster dengan penyimpanan berjenjang menggunakan. AWS Management Console

Mengaktifkan penyimpanan berjenjang pada topik yang ada

Untuk mengaktifkan penyimpanan berjenjang pada topik yang ada, gunakan sintaks `alter` perintah dalam contoh berikut. Saat Anda mengaktifkan penyimpanan berjenjang pada topik yang sudah ada, Anda tidak dibatasi pada versi klien Apache Kafka tertentu.

```
bin/kafka-configs.sh --bootstrap-server $bsrv --alter --entity-type topics
 --entity-name msk-ts-topic --add-config 'remote.storage.enable=true,
 local.retention.ms=604800000, retention.ms=15550000000'
```

Menonaktifkan penyimpanan berjenjang pada topik yang ada

Untuk menonaktifkan penyimpanan berjenjang pada topik yang ada, gunakan sintaks `alter` perintah dalam urutan yang sama seperti saat Anda mengaktifkan penyimpanan berjenjang.

```
bin/kafka-configs.sh --bootstrap-server $bs --alter --entity-type topics --
entity-name MSKTutorialTopic --add-config 'remote.log.msk.disable.policy=Delete,
 remote.storage.enable=false'
```

> **Note**
>
> Saat Anda menonaktifkan penyimpanan berjenjang, Anda sepenuhnya menghapus data topik di penyimpanan berjenjang. Apache Kafka mempertahankan data penyimpanan primer, tetapi masih menerapkan aturan retensi primer berdasarkan. `local.retention.ms` Setelah menonaktifkan penyimpanan berjenjang pada suatu topik, Anda tidak dapat mengaktifkannya kembali. Jika Anda ingin menonaktifkan penyimpanan berjenjang pada topik yang ada, Anda tidak dibatasi untuk versi klien Apache Kafka tertentu.

Mengaktifkan penyimpanan berjenjang pada cluster yang ada menggunakan CLI AWS

> **Note**
>
> Anda dapat mengaktifkan penyimpanan berjenjang hanya jika log.cleanup.policy klaster Anda diseteldelete, karena topik yang dipadatkan tidak didukung pada penyimpanan berjenjang. Kemudian, Anda dapat mengonfigurasi log.cleanup.policy masing-masing topik menjadi `compact` jika penyimpanan berjenjang tidak diaktifkan pada topik tertentu. Lihat Konfigurasi tingkat topik untuk detail selengkapnya tentang atribut konfigurasi yang didukung.

1. Perbarui versi Kafka - Versi cluster bukan bilangan bulat sederhana. Untuk menemukan versi cluster saat ini, gunakan `DescribeCluster` operasi atau perintah `describe-cluster` AWS CLI. Contoh versi adalah`KTVPDKIKX0DER`.

```
aws kafka update-cluster-kafka-version --cluster-arn ClusterArn --current-version
 Current-Cluster-Version --target-kafka-version 3.6.0
```

2. Edit mode penyimpanan cluster. Contoh kode berikut menunjukkan pengeditan mode penyimpanan cluster untuk `TIERED` menggunakan `update-storage`API.

```
aws kafka update-storage --current-version Current-Cluster-Version --cluster-arn
 Cluster-arn --storage-mode TIERED
```

Memperbarui penyimpanan berjenjang pada cluster yang ada menggunakan konsol

> **Note**
>
> Anda dapat mengaktifkan penyimpanan berjenjang hanya jika log.cleanup.policy klaster Anda diseteldelete, karena topik yang dipadatkan tidak didukung pada penyimpanan berjenjang. Kemudian, Anda dapat mengonfigurasi log.cleanup.policy masing-masing topik menjadi `compact` jika penyimpanan berjenjang tidak diaktifkan pada topik tertentu. Lihat Konfigurasi tingkat topik untuk detail selengkapnya tentang atribut konfigurasi yang didukung.

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Buka halaman ringkasan cluster dan pilih Properties.
3. Buka bagian Penyimpanan dan pilih Edit mode penyimpanan cluster.
4. Pilih Penyimpanan berjenjang dan penyimpanan EBS dan Simpan perubahan.

Meningkatkan penyimpanan broker

Anda dapat meningkatkan jumlah penyimpanan EBS per broker. Anda tidak dapat mengurangi penyimpanan.

Volume penyimpanan tetap tersedia selama operasi penskalaan ini.

> **Important**
>
> Saat penyimpanan diskalakan untuk cluster MSK, penyimpanan tambahan segera tersedia. Namun, cluster memerlukan periode pendinginan setelah setiap peristiwa penskalaan penyimpanan. Amazon MSK menggunakan periode pendinginan ini untuk mengoptimalkan cluster sebelum dapat diskalakan lagi. Periode ini dapat berkisar dari minimal 6 jam hingga lebih dari 24 jam, tergantung pada ukuran penyimpanan dan pemanfaatan cluster dan lalu lintas. Ini berlaku untuk peristiwa penskalaan otomatis dan penskalaan manual menggunakan operasi UpdateBrokerPenyimpanan. Untuk informasi tentang ukuran penyimpanan yang tepat, lihat. Praktik terbaik

Anda dapat menggunakan penyimpanan berjenjang untuk meningkatkan jumlah penyimpanan yang tidak terbatas untuk broker Anda. Lihat, Penyimpanan berjenjang.

Topik

Penskalaan Otomatis

Untuk memperluas penyimpanan klaster secara otomatis sebagai respons terhadap peningkatan penggunaan, Anda dapat mengonfigurasi kebijakan Penskalaan Otomatis Aplikasi untuk Amazon MSK. Dalam kebijakan auto-scaling, Anda menetapkan pemanfaatan disk target dan kapasitas penskalaan maksimum.

Sebelum Anda menggunakan penskalaan otomatis untuk Amazon MSK, Anda harus mempertimbangkan hal berikut:

- > **Important**
  >
  > Tindakan penskalaan penyimpanan hanya dapat terjadi setiap enam jam sekali.

  Kami menyarankan Anda memulai dengan volume penyimpanan berukuran tepat untuk kebutuhan penyimpanan Anda. Untuk panduan tentang ukuran kanan klaster Anda, lihat. Ukuran klaster Anda dengan benar: Jumlah broker per cluster
- Amazon MSK tidak mengurangi penyimpanan cluster sebagai respons terhadap pengurangan penggunaan. Amazon MSK tidak mendukung penurunan ukuran volume penyimpanan. Jika Anda perlu mengurangi ukuran penyimpanan klaster, Anda harus memigrasikan klaster yang ada ke klaster dengan penyimpanan yang lebih kecil. Untuk informasi tentang migrasi klaster, lihatMigrasi:.
- Amazon MSK tidak mendukung penskalaan otomatis di Wilayah Asia Pasifik (Osaka) dan Afrika (Cape Town).
- Saat Anda mengaitkan kebijakan auto-scaling dengan klaster, Auto Scaling Amazon EC2 secara otomatis membuat alarm Amazon untuk pelacakan target. CloudWatch Jika Anda menghapus klaster dengan kebijakan auto-scaling, CloudWatch alarm ini tetap ada. Untuk menghapus CloudWatch alarm, Anda harus menghapus kebijakan auto-scaling dari klaster sebelum menghapus klaster. Untuk mempelajari selengkapnya tentang pelacakan target, lihat Kebijakan penskalaan pelacakan target untuk Penskalaan Otomatis Amazon EC2 di Panduan Pengguna Auto Scaling Amazon EC2.

Detail kebijakan penskalaan otomatis

Kebijakan auto-scaling menentukan parameter berikut untuk klaster Anda:

- Target Pemanfaatan Penyimpanan: Ambang batas pemanfaatan penyimpanan yang digunakan Amazon MSK untuk memicu operasi auto-scaling. Anda dapat menetapkan target pemanfaatan antara 10% dan 80% dari kapasitas penyimpanan saat ini. Kami menyarankan Anda menetapkan Target Pemanfaatan Penyimpanan antara 50% dan 60%.
- Kapasitas Penyimpanan Maksimum: Batas penskalaan maksimum yang dapat ditetapkan MSK Amazon untuk penyimpanan broker Anda. Anda dapat mengatur kapasitas penyimpanan maksimum hingga 16 TiB per broker. Untuk informasi selengkapnya, lihat Kuota MSK Amazon.

Ketika Amazon MSK mendeteksi bahwa `Maximum Disk Utilization` metrik Anda sama dengan atau lebih besar dari `Storage Utilization Target` pengaturan, itu meningkatkan kapasitas penyimpanan Anda dengan jumlah yang sama dengan yang lebih besar dari dua angka: 10 GiB atau 10% dari penyimpanan saat ini. Misalnya, jika Anda memiliki 1000 GiB, jumlah itu adalah 100 GiB. Layanan ini memeriksa penggunaan penyimpanan Anda setiap menit. Operasi penskalaan lebih lanjut terus meningkatkan penyimpanan dengan jumlah yang sama dengan yang lebih besar dari dua angka: 10 GiB atau 10% dari penyimpanan saat ini.

Untuk menentukan apakah operasi auto-scaling telah terjadi, gunakan operasi. ListClusterOperations

Menyiapkan penskalaan otomatis untuk kluster MSK Amazon Anda

Anda dapat menggunakan konsol MSK Amazon, Amazon MSK API, atau AWS CloudFormation untuk menerapkan penskalaan otomatis untuk penyimpanan. CloudFormation Dukungan tersedia melalui Application Auto Scaling

> **Note**
>
> Anda tidak dapat menerapkan penskalaan otomatis saat membuat klaster. Anda harus terlebih dahulu membuat klaster, lalu membuat dan mengaktifkan kebijakan auto-scaling untuknya. Namun, Anda dapat membuat kebijakan saat layanan Amazon MSK membuat klaster Anda.

Menyiapkan penskalaan otomatis menggunakan AWS Management Console

1. Masuk ke AWS Management Console, dan buka konsol MSK Amazon di https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Dalam daftar cluster, pilih cluster Anda. Ini membawa Anda ke halaman yang mencantumkan detail tentang cluster.
3. Di bagian Penskalaan otomatis untuk penyimpanan, pilih Konfigurasi.
4. Buat dan beri nama kebijakan auto-scaling. Tentukan target pemanfaatan penyimpanan, kapasitas penyimpanan maksimum, dan metrik target.
5. Pilih `Save changes`.

Saat Anda menyimpan dan mengaktifkan kebijakan baru, kebijakan menjadi aktif untuk klaster. Amazon MSK kemudian memperluas penyimpanan cluster ketika target pemanfaatan penyimpanan tercapai.

Menyiapkan penskalaan otomatis menggunakan CLI

1. Gunakan RegisterScalableTargetperintah untuk mendaftarkan target pemanfaatan penyimpanan.
2. Gunakan PutScalingPolicyperintah untuk membuat kebijakan ekspansi otomatis.

Menyiapkan penskalaan otomatis menggunakan API

1. Gunakan RegisterScalableTargetAPI untuk mendaftarkan target pemanfaatan penyimpanan.
2. Gunakan PutScalingPolicyAPI untuk membuat kebijakan ekspansi otomatis.

Penskalaan manual

Untuk meningkatkan penyimpanan, tunggu cluster berada dalam `ACTIVE` status. Penskalaan penyimpanan memiliki periode pendinginan setidaknya enam jam di antara acara. Meskipun operasi membuat penyimpanan tambahan segera tersedia, layanan melakukan pengoptimalan pada cluster Anda yang dapat memakan waktu hingga 24 jam atau lebih. Durasi pengoptimalan ini sebanding dengan ukuran penyimpanan Anda.

Meningkatkan penyimpanan broker menggunakan AWS Management Console

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.

2. Pilih cluster MSK yang ingin Anda perbarui penyimpanan broker.
3. Di bagian Penyimpanan, pilih Edit.
4. Tentukan volume penyimpanan yang Anda inginkan. Anda hanya dapat menambah jumlah penyimpanan, Anda tidak dapat menguranginya.
5. Pilih Simpan perubahan.

Meningkatkan penyimpanan broker menggunakan AWS CLI

Jalankan perintah berikut, ganti *ClusterArn*dengan Amazon Resource Name (ARN) yang Anda peroleh saat membuat cluster Anda. Jika Anda tidak memiliki ARN untuk cluster Anda, Anda dapat menemukannya dengan mencantumkan semua cluster. Untuk informasi selengkapnya, lihat the section called “Daftar cluster”.

Ganti *Current-Cluster-Version dengan versi* cluster saat ini.

> **Important**
>
> Versi cluster bukan bilangan bulat sederhana. Untuk menemukan versi cluster saat ini, gunakan DescribeClusteroperasi atau perintah AWS CLI deskripsi-cluster. Contoh versi adalahKTVPDKIKX0DER.

Parameter *Target-Volume-in-GIB* mewakili jumlah penyimpanan yang Anda inginkan untuk dimiliki setiap broker. Hanya mungkin untuk memperbarui penyimpanan untuk semua broker. Anda tidak dapat menentukan broker individu untuk memperbarui penyimpanan. Nilai yang Anda tentukan untuk *Target-Volume-in-GIB* harus berupa bilangan bulat yang lebih besar dari 100 GiB. Penyimpanan per broker setelah operasi pembaruan tidak dapat melebihi 16384 GiB.

```
aws kafka update-broker-storage --cluster-arn ClusterArn --current-version Current-
Cluster-Version --target-broker-ebs-volume-info '{"KafkaBrokerNodeId": "All",
 "VolumeSizeGB": Target-Volume-in-GiB}'
```

Meningkatkan penyimpanan broker menggunakan API

Untuk memperbarui penyimpanan broker menggunakan API, lihat UpdateBrokerPenyimpanan.

Penyediaan throughput penyimpanan

Broker MSK Amazon mempertahankan data tentang volume penyimpanan. Penyimpanan I/O dikonsumsi ketika produsen menulis ke cluster, ketika data direplikasi antara broker, dan ketika konsumen membaca data yang tidak ada dalam memori. Throughput penyimpanan volume adalah tingkat di mana data dapat ditulis dan dibaca dari volume penyimpanan. Throughput penyimpanan yang disediakan adalah kemampuan untuk menentukan tarif tersebut untuk broker di cluster Anda.

Anda dapat menentukan tingkat throughput yang disediakan dalam MiB per detik untuk cluster yang brokernya berukuran `kafka.m5.4xlarge` atau lebih besar dan jika volume penyimpanan 10 GiB atau lebih besar. Dimungkinkan untuk menentukan throughput yang disediakan selama pembuatan cluster. Anda juga dapat mengaktifkan atau menonaktifkan throughput yang disediakan untuk klaster yang berada dalam status. `ACTIVE`

Kemacetan throughput

Ada beberapa penyebab kemacetan dalam throughput broker: throughput volume, throughput jaringan Amazon EC2 ke Amazon EBS, dan throughput keluar Amazon EC2. Anda dapat mengaktifkan throughput penyimpanan yang disediakan untuk menyesuaikan throughput volume. Namun, keterbatasan throughput broker dapat disebabkan oleh throughput jaringan Amazon EC2 ke Amazon EBS dan throughput keluar Amazon EC2.

Output keluar Amazon EC2 dipengaruhi oleh jumlah kelompok konsumen dan konsumen per kelompok konsumen. Selain itu, throughput jaringan Amazon EC2 hingga Amazon EBS dan throughput keluar Amazon EC2 lebih tinggi untuk ukuran broker yang lebih besar.

Untuk ukuran volume 10 GiB atau lebih besar, Anda dapat menyediakan throughput penyimpanan 250 MiB per detik atau lebih besar. 250 MiB per detik adalah default. Untuk menyediakan throughput penyimpanan, Anda harus memilih ukuran broker kafka.m5.4xlarge atau lebih besar (atau kafka.m7g.2xlarge atau lebih besar), dan Anda dapat menentukan throughput maksimum seperti yang ditunjukkan pada tabel berikut.

ukuran broker	Throughput penyimpanan maksimum (MIB/detik)
kafka.m5.4xlarge	593
kafka.m5.8xlarge	850

ukuran broker	Throughput penyimpanan maksimum (MIB/detik)
kafka.m5.12xlarge	1000
kafka.m5.16xlarge	1000
kafka.m5.24xlarge	1000
kafka.m7g.2xlarge	312,5
kafka.m7g.4xlarge	625
kafka.m7g.8xlarge	1000
kafka.m7g.12xlarge	1000
kafka.m7g.16xlarge	1000

Mengukur throughput penyimpanan

Anda dapat menggunakan `VolumeWriteBytes` metrik `VolumeReadBytes` dan untuk mengukur throughput penyimpanan rata-rata sebuah cluster. Jumlah kedua metrik ini memberikan throughput penyimpanan rata-rata dalam byte. Untuk mendapatkan throughput penyimpanan rata-rata untuk sebuah cluster, atur kedua metrik ini ke SUM dan periode menjadi 1 menit, lalu gunakan rumus berikut.

```
Average storage throughput in MiB/s = (Sum(VolumeReadBytes) + Sum(VolumeWriteBytes)) /
 (60 * 1024 * 1024)
```

Untuk informasi tentang `VolumeReadBytes` dan `VolumeWriteBytes` metrik, lihatthe section called “PER_BROKERPemantauan tingkat”.

Pembaruan konfigurasi

Anda dapat memperbarui konfigurasi MSK Amazon sebelum atau setelah Anda mengaktifkan throughput yang disediakan. Namun, Anda tidak akan melihat throughput yang diinginkan hingga Anda melakukan kedua tindakan: perbarui parameter `num.replica.fetchers` konfigurasi dan aktifkan throughput yang disediakan.

Dalam konfigurasi MSK Amazon default, `num.replica.fetchers` memiliki nilai 2. Untuk memperbaruinum.replica.fetchers, Anda dapat menggunakan nilai yang disarankan dari tabel berikut. Nilai-nilai ini untuk tujuan panduan. Kami menyarankan Anda menyesuaikan nilai-nilai ini berdasarkan kasus penggunaan Anda.

ukuran broker	num.replica.fetchers
kafka.m5.4xlarge	4
kafka.m5.8xlarge	8
kafka.m5.12xlarge	14
kafka.m5.16xlarge	16
kafka.m5.24xlarge	16

Konfigurasi Anda yang diperbarui mungkin tidak berlaku hingga 24 jam, dan mungkin memakan waktu lebih lama ketika volume sumber tidak sepenuhnya digunakan. Namun, kinerja volume transisi setidaknya sama dengan kinerja volume penyimpanan sumber selama periode migrasi. Volume 1 TiB yang sepenuhnya digunakan biasanya membutuhkan waktu sekitar enam jam untuk bermigrasi ke konfigurasi yang diperbarui.

Penyediaan throughput penyimpanan menggunakan AWS Management Console

1. Masuk ke AWS Management Console, dan buka konsol MSK Amazon di https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Pilih Buat klaster.
3. Pilih Custom create.
4. Tentukan nama untuk cluster.
5. Di bagian Penyimpanan, pilih Aktifkan.
6. Pilih nilai untuk throughput penyimpanan per broker.
7. Pilih VPC, zona dan subnet, dan grup keamanan.
8. Pilih Selanjutnya.
9. Di bagian bawah langkah Keamanan, pilih Berikutnya.
10. Di bagian bawah langkah Pemantauan dan tag, pilih Berikutnya.

11. Tinjau pengaturan cluster, lalu pilih Buat cluster.

Penyediaan throughput penyimpanan menggunakan AWS CLI

Bagian ini menunjukkan contoh bagaimana Anda dapat menggunakan AWS CLI untuk membuat klaster dengan throughput yang disediakan diaktifkan.

1. Salin JSON berikut dan tempel ke dalam file. Ganti ID subnet dan placeholder ID grup keamanan dengan nilai dari akun Anda. Beri nama file `cluster-creation.json` dan simpan.

```
{
    "Provisioned": {
        "BrokerNodeGroupInfo":{
            "InstanceType":"kafka.m5.4xlarge",
            "ClientSubnets":[
                "Subnet-1-ID",
                "Subnet-2-ID"
            ],
            "SecurityGroups":[
                "Security-Group-ID"
            ],
            "StorageInfo": {
                "EbsStorageInfo": {
                    "VolumeSize": 10,
                    "ProvisionedThroughput": {
                        "Enabled": true,
                        "VolumeThroughput": 250
                    }
                }
            }
        },
        "EncryptionInfo": {
            "EncryptionInTransit": {
                "InCluster": false,
                "ClientBroker": "PLAINTEXT"
            }
        },
        "KafkaVersion":"2.8.1",
        "NumberOfBrokerNodes": 2
    },
    "ClusterName": "provisioned-throughput-example"
}
```

2. Jalankan AWS CLI perintah berikut dari direktori tempat Anda menyimpan file JSON di langkah sebelumnya.

```
aws kafka create-cluster-v2 --cli-input-json file://cluster-creation.json
```

Menyediakan throughput penyimpanan menggunakan API

Untuk mengonfigurasi throughput penyimpanan yang disediakan saat membuat cluster, gunakan V2. CreateCluster

Memperbarui ukuran broker

Anda dapat menskalakan cluster MSK Anda sesuai permintaan dengan mengubah ukuran broker Anda tanpa menetapkan kembali partisi Apache Kafka. Mengubah ukuran broker Anda memberi Anda fleksibilitas untuk menyesuaikan kapasitas komputasi klaster MSK Anda berdasarkan perubahan beban kerja Anda, tanpa mengganggu I/O cluster Anda. Amazon MSK menggunakan ukuran broker yang sama untuk semua broker dalam cluster tertentu.

Bagian ini menjelaskan cara memperbarui ukuran broker untuk kluster MSK Anda. Anda dapat memperbarui ukuran broker cluster Anda dari M5 atau T3 ke m7g, atau dari m7g ke M5. Ketahuilah bahwa bermigrasi ke ukuran broker yang lebih kecil dapat menurunkan kinerja dan mengurangi throughput maksimum yang dapat dicapai per broker. Migrasi ke ukuran broker yang lebih besar dapat meningkatkan kinerja tetapi mungkin lebih mahal.

Pembaruan ukuran broker terjadi secara bergulir saat cluster aktif dan berjalan. Ini berarti bahwa Amazon MSK menurunkan satu broker pada satu waktu untuk melakukan pembaruan ukuran broker. Untuk informasi tentang cara membuat klaster sangat tersedia selama pembaruan ukuran broker, lihat. the section called “Bangun cluster yang sangat tersedia” Untuk lebih mengurangi dampak potensial pada produktivitas, Anda dapat melakukan pembaruan ukuran broker selama periode lalu lintas rendah.

Selama pembaruan ukuran broker, Anda dapat terus memproduksi dan mengkonsumsi data. Namun, Anda harus menunggu hingga pembaruan selesai sebelum Anda dapat me-reboot broker atau memanggil salah satu operasi pembaruan yang terdaftar di bawah operasi MSK Amazon.

Jika Anda ingin memperbarui klaster Anda ke ukuran broker yang lebih kecil, kami sarankan Anda mencoba pembaruan pada klaster uji terlebih dahulu untuk melihat bagaimana pengaruhnya terhadap skenario Anda.

> **Important**
>
> Anda tidak dapat memperbarui cluster ke ukuran broker yang lebih kecil jika jumlah partisi per broker melebihi jumlah maksimum yang ditentukan dalamthe section called " Ukuran kluster Anda dengan benar: Jumlah partisi per broker".

Memperbarui ukuran broker menggunakan AWS Management Console

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Pilih cluster MSK yang ingin Anda perbarui ukuran brokernya.
3. Pada halaman detail untuk cluster, temukan bagian ringkasan Broker, dan pilih Edit ukuran broker.
4. Pilih ukuran broker yang Anda inginkan dari daftar.
5. Simpan perubahan.

Memperbarui ukuran broker menggunakan AWS CLI

1. Jalankan perintah berikut, ganti *ClusterArn*dengan Amazon Resource Name (ARN) yang Anda peroleh saat membuat cluster Anda. Jika Anda tidak memiliki ARN untuk cluster Anda, Anda dapat menemukannya dengan mencantumkan semua cluster. Untuk informasi selengkapnya, lihat the section called "Daftar cluster".

   Ganti *Current-Cluster-Version* dengan versi cluster saat ini dan *TargetType*dengan ukuran baru yang Anda inginkan dari broker. Untuk mempelajari lebih lanjut tentang ukuran broker, lihatthe section called "Ukuran broker".

   ```
   aws kafka update-broker-type --cluster-arn ClusterArn --current-version Current-Cluster-Version --target-instance-type TargetType
   ```

   Berikut ini adalah contoh cara menggunakan perintah ini:

   ```
   aws kafka update-broker-type --cluster-arn "arn:aws:kafka:us-east-1:0123456789012:cluster/exampleName/abcd1234-0123-abcd-5678-1234abcd-1" --current-version "K1X5R6FKA87" --target-instance-type kafka.m5.large
   ```

Output dari perintah ini terlihat seperti contoh JSON berikut.

```
{
    "ClusterArn": "arn:aws:kafka:us-east-1:0123456789012:cluster/exampleName/
abcd1234-0123-abcd-5678-1234abcd-1",
    "ClusterOperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-
operation/exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
abcd-4f7f-1234-9876543210ef"
}
```

2. Untuk mendapatkan hasil `update-broker-type` operasi, jalankan perintah berikut, ganti *`ClusterOperationArn`* dengan ARN yang Anda peroleh dalam output perintah. `update-broker-type`

```
aws kafka describe-cluster-operation --cluster-operation-arn ClusterOperationArn
```

Output dari `describe-cluster-operation` perintah ini terlihat seperti contoh JSON berikut.

```
{
  "ClusterOperationInfo": {
    "ClientRequestId": "982168a3-939f-11e9-8a62-538df00285db",
    "ClusterArn": "arn:aws:kafka:us-east-1:0123456789012:cluster/exampleName/
abcd1234-0123-abcd-5678-1234abcd-1",
    "CreationTime": "2021-01-09T02:24:22.198000+00:00",
    "OperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-operation/
exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
abcd-4f7f-1234-9876543210ef",
    "OperationState": "UPDATE_COMPLETE",
    "OperationType": "UPDATE_BROKER_TYPE",
    "SourceClusterInfo": {
      "InstanceType": "t3.small"
    },
    "TargetClusterInfo": {
      "InstanceType": "m5.large"
    }
  }
}
```

Jika `OperationState` memiliki nilai`UPDATE_IN_PROGRESS`, tunggu sebentar, lalu jalankan `describe-cluster-operation` perintah lagi.

Memperbarui ukuran broker menggunakan API

Untuk memperbarui ukuran broker menggunakan API, lihat UpdateBrokerMengetik.

Anda dapat menggunakan `UpdateBrokerType` untuk memperbarui ukuran broker cluster Anda dari M5 atau T3 ke m7G, atau dari m7g ke M5.

Memperbarui konfigurasi cluster MSK Amazon

Untuk memperbarui konfigurasi cluster, pastikan bahwa cluster berada dalam `ACTIVE` status. Anda juga harus memastikan bahwa jumlah partisi per broker di klaster MSK Anda berada di bawah batas yang dijelaskan dalam. the section called " Ukuran kluster Anda dengan benar: Jumlah partisi per broker" Anda tidak dapat memperbarui konfigurasi klaster yang melebihi batas ini.

Untuk informasi tentang konfigurasi MSK, termasuk cara membuat konfigurasi kustom, properti mana yang dapat Anda perbarui, dan apa yang terjadi ketika Anda memperbarui konfigurasi klaster yang ada, lihatKonfigurasi.

Memperbarui konfigurasi cluster menggunakan AWS CLI

1. Salin JSON berikut dan simpan ke file. Beri nama file`configuration-info.json`. Ganti *`ConfigurationArn`*dengan Amazon Resource Name (ARN) dari konfigurasi yang ingin Anda gunakan untuk memperbarui cluster. String ARN harus dalam tanda kutip di JSON berikut.

   Ganti *`Konfigurasi-Revisi`* dengan revisi konfigurasi yang ingin Anda gunakan. Revisi konfigurasi adalah bilangan bulat (bilangan bulat) yang dimulai dari. 1 Bilangan bulat ini tidak boleh dalam tanda kutip di JSON berikut.

   ```
   {
        "Arn": ConfigurationArn,
        "Revision": Configuration-Revision
   }
   ```

2. Jalankan perintah berikut, ganti *`ClusterArn`*dengan ARN yang Anda peroleh saat membuat cluster Anda. Jika Anda tidak memiliki ARN untuk cluster Anda, Anda dapat menemukannya dengan mencantumkan semua cluster. Untuk informasi selengkapnya, lihat the section called "Daftar cluster".

Ganti *Path-to-config-info-file dengan path ke file* info konfigurasi Anda. Jika Anda menamai file yang Anda buat pada langkah sebelumnya `configuration-info.json` dan menyimpannya di direktori saat ini, maka *Path-to-config-info-file* adalah. `configuration-info.json`

Ganti *Current-Cluster-Version dengan versi* cluster saat ini.

> **Important**
>
> Versi cluster bukan bilangan bulat sederhana. Untuk menemukan versi cluster saat ini, gunakan DescribeClusteroperasi atau perintah AWS CLI deskripsi-cluster. Contoh versi adalah`KTVPDKIKX0DER`.

```
aws kafka update-cluster-configuration --cluster-arn ClusterArn --configuration-
info file://Path-to-Config-Info-File --current-version Current-Cluster-Version
```

Berikut ini adalah contoh cara menggunakan perintah ini:

```
aws kafka update-cluster-configuration --cluster-arn "arn:aws:kafka:us-
east-1:0123456789012:cluster/exampleName/abcd1234-0123-abcd-5678-1234abcd-1" --
configuration-info file://c:\users\tester\msk\configuration-info.json --current-
version "K1X5R6FKA87"
```

Output dari `update-cluster-configuration` perintah ini terlihat seperti contoh JSON berikut.

```
{
    "ClusterArn": "arn:aws:kafka:us-east-1:012345678012:cluster/exampleClusterName/
abcdefab-1234-abcd-5678-cdef0123ab01-2",
    "ClusterOperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-
operation/exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
abcd-4f7f-1234-9876543210ef"
}
```

3. Untuk mendapatkan hasil `update-cluster-configuration` operasi, jalankan perintah berikut, ganti *ClusterOperationArn* dengan ARN yang Anda peroleh dalam output perintah. `update-cluster-configuration`

```
aws kafka describe-cluster-operation --cluster-operation-arn ClusterOperationArn
```

Output dari `describe-cluster-operation` perintah ini terlihat seperti contoh JSON berikut.

```
{
    "ClusterOperationInfo": {
        "ClientRequestId": "982168a3-939f-11e9-8a62-538df00285db",
        "ClusterArn": "arn:aws:kafka:us-east-1:012345678012:cluster/
exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2",
        "CreationTime": "2019-06-20T21:08:57.735Z",
        "OperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-
operation/exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
abcd-4f7f-1234-9876543210ef",
        "OperationState": "UPDATE_COMPLETE",
        "OperationType": "UPDATE_CLUSTER_CONFIGURATION",
        "SourceClusterInfo": {},
        "TargetClusterInfo": {
            "ConfigurationInfo": {
                "Arn": "arn:aws:kafka:us-east-1:123456789012:configuration/
ExampleConfigurationName/abcdabcd-abcd-1234-abcd-abcd123e8e8e-1",
                "Revision": 1
            }
        }
    }
}
```

Dalam output ini, `OperationType` adalah`UPDATE_CLUSTER_CONFIGURATION`. Jika `OperationState` memiliki nilai`UPDATE_IN_PROGRESS`, tunggu sebentar, lalu jalankan `describe-cluster-operation` perintah lagi.

Memperbarui konfigurasi cluster menggunakan API

Untuk menggunakan API untuk memperbarui konfigurasi klaster, lihat UpdateClusterKonfigurasi.

Memperluas kluster MSK Amazon

Gunakan operasi MSK Amazon ini ketika Anda ingin menambah jumlah broker di klaster MSK Anda. Untuk memperluas cluster, pastikan itu dalam `ACTIVE` keadaan.

> **Important**
>
> Jika Anda ingin memperluas kluster MSK, pastikan untuk menggunakan operasi MSK Amazon ini. Jangan mencoba menambahkan broker ke cluster tanpa menggunakan operasi ini.

Untuk informasi tentang cara menyeimbangkan kembali partisi setelah Anda menambahkan broker ke cluster, lihat. the section called “Tetapkan kembali partisi”

Memperluas cluster menggunakan AWS Management Console

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Pilih klaster MSK yang jumlah brokernya ingin Anda tingkatkan.
3. Pada halaman detail cluster, pilih tombol Edit di sebelah judul Cluster-Level Broker Details.
4. Masukkan jumlah broker yang Anda inginkan klaster untuk memiliki per Availability Zone dan kemudian pilih Simpan perubahan.

Memperluas cluster menggunakan AWS CLI

1. Jalankan perintah berikut, ganti *ClusterArn*dengan Amazon Resource Name (ARN) yang Anda peroleh saat membuat cluster Anda. Jika Anda tidak memiliki ARN untuk cluster Anda, Anda dapat menemukannya dengan mencantumkan semua cluster. Untuk informasi selengkapnya, lihat the section called “Daftar cluster”.

   Ganti *Current-Cluster-Version dengan versi* cluster saat ini.

   > **Important**
   >
   > Versi cluster bukan bilangan bulat sederhana. Untuk menemukan versi cluster saat ini, gunakan DescribeClusteroperasi atau perintah AWS CLI deskripsi-cluster. Contoh versi adalah`KTVPDKIKX0DER`.

   Parameter *Target-Number-of-Brokers* mewakili jumlah total node broker yang Anda inginkan untuk dimiliki cluster ketika operasi ini selesai dengan sukses. Nilai yang Anda tentukan

untuk *Target-Number-of-Brokers* harus berupa bilangan bulat yang lebih besar dari jumlah broker saat ini di cluster. Itu juga harus kelipatan dari jumlah Availability Zones.

```
aws kafka update-broker-count --cluster-arn ClusterArn --current-version Current-
Cluster-Version --target-number-of-broker-nodes Target-Number-of-Brokers
```

Output dari `update-broker-count` operasi ini terlihat seperti JSON berikut.

```
{

    "ClusterArn": "arn:aws:kafka:us-east-1:012345678012:cluster/exampleClusterName/
abcdefab-1234-abcd-5678-cdef0123ab01-2",
    "ClusterOperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-
operation/exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
abcd-4f7f-1234-9876543210ef"
}
```

2. Untuk mendapatkan hasil `update-broker-count` operasi, jalankan perintah berikut, ganti *ClusterOperationArn* dengan ARN yang Anda peroleh dalam output perintah. `update-broker-count`

```
aws kafka describe-cluster-operation --cluster-operation-arn ClusterOperationArn
```

Output dari `describe-cluster-operation` perintah ini terlihat seperti contoh JSON berikut.

```
{
    "ClusterOperationInfo": {
        "ClientRequestId": "c0b7af47-8591-45b5-9c0c-909a1a2c99ea",
        "ClusterArn": "arn:aws:kafka:us-east-1:012345678012:cluster/
exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2",
        "CreationTime": "2019-09-25T23:48:04.794Z",
        "OperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-
operation/exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
abcd-4f7f-1234-9876543210ef",
        "OperationState": "UPDATE_COMPLETE",
        "OperationType": "INCREASE_BROKER_COUNT",
        "SourceClusterInfo": {
            "NumberOfBrokerNodes": 9
        },
        "TargetClusterInfo": {
            "NumberOfBrokerNodes": 12
```

```
        }
    }
}
```

Dalam output ini, `OperationType` adalah`INCREASE_BROKER_COUNT`. Jika `OperationState` memiliki nilai`UPDATE_IN_PROGRESS`, tunggu sebentar, lalu jalankan `describe-cluster-operation` perintah lagi.

Memperluas cluster menggunakan API

Untuk meningkatkan jumlah broker dalam klaster menggunakan API, lihat UpdateBrokerMenghitung.

Hapus broker dari cluster MSK Amazon

Gunakan operasi MSK Amazon ini saat Anda ingin menghapus broker dari kluster yang disediakan Amazon Managed Streaming for Apache Kafka (MSK). Anda dapat mengurangi kapasitas penyimpanan dan komputasi klaster Anda dengan menghapus kumpulan broker, tanpa dampak ketersediaan, risiko daya tahan data, atau gangguan pada aplikasi streaming data Anda.

Anda dapat menambahkan lebih banyak broker ke cluster Anda untuk menangani peningkatan lalu lintas, dan menghapus broker ketika lalu lintas mereda. Dengan kemampuan penambahan dan penghapusan broker, Anda dapat memanfaatkan kapasitas cluster Anda dengan sebaik-baiknya dan mengoptimalkan biaya infrastruktur MSK Anda. Penghapusan broker memberi Anda kontrol tingkat broker atas kapasitas klaster yang ada agar sesuai dengan kebutuhan beban kerja Anda dan menghindari migrasi ke klaster lain.

Gunakan AWS Konsol, Antarmuka Baris Perintah (CLI), SDK, atau AWS CloudFormation untuk mengurangi jumlah broker dari klaster yang disediakan. MSK memilih broker yang tidak memiliki partisi pada mereka (kecuali untuk topik kenari) dan mencegah aplikasi menghasilkan data ke broker tersebut, sambil dengan aman menghapus broker tersebut dari cluster.

Anda harus menghapus satu broker per Availability Zone, jika Anda ingin mengurangi penyimpanan dan komputasi cluster. Misalnya, Anda dapat menghapus dua broker dari dua klaster Availability Zone, atau tiga broker dari tiga klaster Availability Zone dalam satu operasi penghapusan broker.

Untuk informasi tentang cara menyeimbangkan kembali partisi setelah Anda menghapus broker dari cluster, lihat. the section called “Tetapkan kembali partisi”

Anda dapat menghapus broker dari semua kluster yang disediakan MSK berbasis M5 dan M7g, terlepas dari ukuran instans.

Penghapusan broker didukung pada Kafka versi 2.8.1 dan di atasnya, termasuk pada kluster mode Kraft.

Topik

Bersiaplah untuk menghapus broker dengan menghapus semua partisi

Sebelum Anda memulai proses penghapusan broker, pertama-tama pindahkan semua partisi, kecuali yang untuk topik `__amazon_msk_canary` dan `__amazon_msk_canary_state` dari broker yang Anda rencanakan untuk dihapus. Ini adalah topik internal yang dibuat Amazon MSK untuk kesehatan klaster dan metrik diagnostik.

Anda dapat menggunakan Kafka admin API atau Cruise Control untuk memindahkan partisi ke broker lain yang ingin Anda pertahankan di cluster. Lihat Menetapkan ulang partisi.

Contoh proses untuk menghapus partisi

Bagian ini adalah contoh cara menghapus partisi dari broker yang ingin Anda hapus. Asumsikan Anda memiliki cluster dengan 6 broker, 2 broker di setiap AZ, dan memiliki empat topik:

- `__amazon_msk_canary`
- `__consumer_offsets`
- `__amazon_msk_connect_offsets_my-mskc-connector_12345678-09e7-c657f7e4ff32-2`
- `msk-brk-rmv`

1. Buat mesin klien seperti yang dijelaskan dalam Buat mesin klien.
2. Setelah mengkonfigurasi mesin klien, jalankan perintah berikut untuk mencantumkan semua topik yang tersedia di cluster Anda.

```
./bin/kafka-topics.sh --bootstrap-server “CLUSTER_BOOTSTRAP_STRING” --list
```

Dalam contoh ini, kita melihat empat nama topik,`__amazon_msk_canary`,`__consumer_offsets`,`__amazon_msk_connect_offsets_my-mskc-connector_12345678-09e7-c657f7e4ff32-2`, dan`msk-brk-rmv`.

3. Buat file json yang dipanggil `topics.json` pada mesin klien dan tambahkan semua nama topik pengguna seperti pada contoh kode berikut. Anda tidak perlu menyertakan nama `__amazon_msk_canary` topik karena ini adalah topik yang dikelola layanan yang akan dipindahkan secara otomatis bila diperlukan.

```
{
"topics": [
{"topic": "msk-brk-rmv"},
{"topic": "__consumer_offsets"},
{"topic": "__amazon_msk_connect_offsets_my-mskc-connector_12345678-09e7-
c657f7e4ff32-2"}
],
"version":1
}
```

4. Jalankan perintah berikut untuk menghasilkan proposal untuk memindahkan partisi ke hanya 3 broker dari 6 broker di cluster.

```
./bin/kafka-reassign-partitions.sh --bootstrap-server “CLUSTER_BOOTSTRAP_STRING” --
topics-to-move-json-file topics.json --broker-list 1,2,3 --generate
```

5. Buat file bernama `reassignment-file.json` dan salin yang `proposed partition reassignment configuration` Anda dapatkan dari perintah di atas.
6. Jalankan perintah berikut untuk memindahkan partisi yang Anda tentukan di. `reassignment-file.json`

```
./bin/kafka-reassign-partitions.sh --bootstrap-server “CLUSTER_BOOTSTRAP_STRING” --
reassignment-json-file reassignment-file.json --execute
```

Output akan terlihat serupa dengan yang berikut ini:

```
Successfully started partition reassignments for morpheus-test-topic-1-0,test-
topic-1-0
```

7. Jalankan perintah berikut untuk memverifikasi semua partisi telah dipindahkan.

```
./bin/kafka-reassign-partitions.sh --bootstrap-server “CLUSTER_BOOTSTRAP_STRING” --
reassignment-json-file reassignment-file.json --verify
```

   Output-nya akan terlihat serupa dengan yang berikut ini. Pantau status hingga semua partisi dalam topik yang Anda minta berhasil ditugaskan kembali:

```
Status of partition reassignment:
Reassignment of partition msk-brk-rmv-0 is completed.
Reassignment of partition msk-brk-rmv-1 is completed.
Reassignment of partition __consumer_offsets-0 is completed.
Reassignment of partition __consumer_offsets-1 is completed.
```

8. Ketika status menunjukkan bahwa penugasan kembali partisi untuk setiap partisi selesai, pantau `UserPartitionExists` metrik selama 5 menit untuk memastikannya ditampilkan `0` untuk broker tempat Anda memindahkan partisi. Setelah mengonfirmasi ini, Anda dapat melanjutkan untuk menghapus broker dari cluster.

Hapus broker dengan AWS Management Console

Untuk menghapus broker dengan Konsol AWS Manajemen

1. Buka konsol MSK Amazon di https://console.aws.amazon.com/msk/.
2. Pilih klaster MSK yang berisi broker yang ingin Anda hapus.
3. Pada halaman detail cluster, pilih tombol Tindakan dan pilih opsi Edit jumlah broker.
4. Masukkan jumlah broker yang Anda inginkan untuk dimiliki cluster per Availability Zone. Konsol merangkum jumlah broker di seluruh zona ketersediaan yang akan dihapus. Pastikan ini apa yang Anda inginkan.
5. Pilih Simpan perubahan.

Untuk mencegah penghapusan broker yang tidak disengaja, konsol meminta Anda untuk mengonfirmasi bahwa Anda ingin menghapus broker.

Hapus broker dengan AWS CLI

Jalankan perintah berikut, ganti `ClusterArn` dengan Amazon Resource Name (ARN) yang Anda peroleh saat membuat cluster Anda. Jika Anda tidak memiliki ARN untuk cluster Anda, Anda dapat menemukannya dengan mencantumkan semua cluster. Untuk informasi lebih lanjut, Daftar kluster MSK Amazon. Ganti `Current-Cluster-Version` dengan versi cluster saat ini.

> **Important**
>
> Versi cluster bukan bilangan bulat sederhana. Untuk menemukan versi cluster saat ini, gunakan DescribeClusteroperasi atau perintah AWS CLI deskripsi-cluster. Contoh versi adalah`KTVPDKIKX0DER`.

Parameter *`Target-Number-of-Brokers`* mewakili jumlah total node broker yang Anda inginkan untuk dimiliki cluster ketika operasi ini selesai dengan sukses. Nilai yang Anda tentukan untuk *`Target-Number-of-Brokers`* harus berupa bilangan bulat yang kurang dari jumlah broker saat ini di cluster. Itu juga harus kelipatan dari jumlah Availability Zones.

```
aws kafka update-broker-count --cluster-arn ClusterArn --current-version Current-
Cluster-Version --target-number-of-broker-nodes Target-Number-of-Brokers
```

Output dari `update-broker-count` operasi ini terlihat seperti JSON berikut.

```
{
"ClusterOperationInfo": {
"ClientRequestId": "c0b7af47-8591-45b5-9c0c-909a1a2c99ea",
        "ClusterArn": "arn:aws:kafka:us-east-1:012345678012:cluster/exampleClusterName/
abcdefab-1234-abcd-5678-cdef0123ab01-2",
        "CreationTime": "2019-09-25T23:48:04.794Z",
        "OperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-
operation/exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
abcd-4f7f-1234-9876543210ef",
        "OperationState": "UPDATE_COMPLETE",
        "OperationType": "DECREASE_BROKER_COUNT",
        "SourceClusterInfo": {
"NumberOfBrokerNodes": 12
        },
        "TargetClusterInfo": {
"NumberOfBrokerNodes": 9
```

```
        }
    }
}
```

Dalam output ini, `OperationType` adalah`DECREASE_BROKER_COUNT`. Jika `OperationState` memiliki nilai`UPDATE_IN_PROGRESS`, tunggu sebentar, lalu jalankan `describe-cluster-operation` perintah lagi.

Hapus broker dengan AWS API

Untuk menghapus broker di klaster menggunakan API, lihat UpdateBrokerMenghitung di Amazon Managed Streaming for Apache Kafka API Referensi.

Memperbarui pengaturan keamanan klaster

Gunakan operasi MSK Amazon ini untuk memperbarui otentikasi dan pengaturan enkripsi pialang klien dari kluster MSK Anda. Anda juga dapat memperbarui Otoritas Keamanan Swasta yang digunakan untuk menandatangani sertifikat untuk otentikasi TLS bersama. Anda tidak dapat mengubah pengaturan enkripsi in-cluster (broker-to-broker).

Cluster harus dalam `ACTIVE` keadaan agar Anda dapat memperbarui pengaturan keamanannya.

Jika Anda mengaktifkan otentikasi menggunakan IAM, SASL, atau TLS, Anda juga harus mengaktifkan enkripsi antara klien dan broker. Tabel berikut menunjukkan kemungkinan kombinasi.

Autentikasi	Opsi enkripsi klien-broker	Enkripsi pialang-pialang
Tidak diautentikasi	TLS, PLAINTEXT, TLS_PLAINTEXT	Bisa on atau off.
MTL	TLS, TLS_PLAINTEXT	Harus di.
SASL/SCRAM	TLS	Harus di.
SELEMPANG/IAM	TLS	Harus di.

Ketika enkripsi klien-broker disetel ke `TLS_PLAINTEXT` dan otentikasi klien diatur ke, `mTLS` Amazon MSK membuat dua jenis pendengar untuk klien untuk terhubung ke: satu pendengar untuk klien

untuk terhubung menggunakan otentikasi mTLS dengan Enkripsi TLS, dan satu lagi untuk klien untuk terhubung tanpa otentikasi atau enkripsi (plaintext).

Untuk informasi selengkapnya tentang setelan keamanan, lihatKeamanan.

Memperbarui pengaturan keamanan klaster menggunakan AWS Management Console

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Pilih kluster MSK yang ingin Anda perbarui.
3. Di bagian Pengaturan keamanan, pilih Edit.
4. Pilih pengaturan otentikasi dan enkripsi yang Anda inginkan untuk cluster, lalu pilih Simpan perubahan.

Memperbarui pengaturan keamanan klaster menggunakan AWS CLI

1. Buat file JSON yang berisi pengaturan enkripsi yang Anda inginkan untuk dimiliki cluster. Berikut adalah contohnya.

   > **Note**
   >
   > Anda hanya dapat memperbarui pengaturan enkripsi klien-broker. Anda tidak dapat memperbarui pengaturan enkripsi in-cluster (broker-to-broker).

   ```
   {"EncryptionInTransit":{"ClientBroker": "TLS"}}
   ```

2. Buat file JSON yang berisi pengaturan otentikasi yang Anda inginkan untuk dimiliki cluster. Berikut adalah contohnya.

   ```
   {"Sasl":{"Scram":{"Enabled":true}}}
   ```

3. Jalankan AWS CLI perintah berikut:

   ```
   aws kafka update-security --cluster-arn ClusterArn --current-version Current-Cluster-Version --client-authentication file://Path-to-Authentication-Settings-JSON-File --encryption-info file://Path-to-Encryption-Settings-JSON-File
   ```

Output dari `update-security` operasi ini terlihat seperti JSON berikut.

```
{

    "ClusterArn": "arn:aws:kafka:us-east-1:012345678012:cluster/exampleClusterName/
abcdefab-1234-abcd-5678-cdef0123ab01-2",
    "ClusterOperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-
operation/exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
abcd-4f7f-1234-9876543210ef"
}
```

4. Untuk melihat status `update-security` operasi, jalankan perintah berikut, ganti *`ClusterOperationArn`* dengan ARN yang Anda peroleh dalam output perintah. `update-security`

```
aws kafka describe-cluster-operation --cluster-operation-arn ClusterOperationArn
```

Output dari `describe-cluster-operation` perintah ini terlihat seperti contoh JSON berikut.

```
{
    "ClusterOperationInfo": {
        "ClientRequestId": "c0b7af47-8591-45b5-9c0c-909a1a2c99ea",
        "ClusterArn": "arn:aws:kafka:us-east-1:012345678012:cluster/
exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2",
        "CreationTime": "2021-09-17T02:35:47.753000+00:00",
        "OperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-
operation/exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
abcd-4f7f-1234-9876543210ef",
        "OperationState": "PENDING",
        "OperationType": "UPDATE_SECURITY",
        "SourceClusterInfo": {},
        "TargetClusterInfo": {}
    }
}
```

Jika `OperationState` memiliki nilai `PENDING` atau`UPDATE_IN_PROGRESS`, tunggu sebentar, lalu jalankan `describe-cluster-operation` perintah lagi.

Memperbarui setelan keamanan klaster menggunakan API

Untuk memperbarui setelan keamanan klaster yang menggunakan API, lihat UpdateSecurity.

> **Note**
>
> Operasi AWS CLI dan API untuk memperbarui pengaturan keamanan klaster adalah idempoten. Ini berarti bahwa jika Anda menjalankan operasi pembaruan keamanan dan menentukan pengaturan otentikasi atau enkripsi yang merupakan pengaturan yang sama dengan yang dimiliki cluster saat ini, pengaturan itu tidak akan berubah.

Mem-boot ulang broker untuk cluster MSK Amazon

Gunakan operasi MSK Amazon ini saat Anda ingin me-reboot broker untuk cluster MSK Anda. Untuk me-reboot broker untuk cluster, pastikan bahwa cluster dalam `ACTIVE` keadaan.

Layanan MSK Amazon dapat me-reboot broker untuk klaster MSK Anda selama pemeliharaan sistem, seperti patching atau upgrade versi. Mem-boot ulang broker secara manual memungkinkan Anda menguji ketahanan klien Kafka Anda untuk menentukan bagaimana mereka merespons pemeliharaan sistem.

Mem-boot ulang broker menggunakan AWS Management Console

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Pilih cluster MSK yang brokernya ingin Anda reboot.
3. Gulir ke bawah ke bagian Detail Broker, dan pilih broker yang ingin Anda reboot.
4. Pilih tombol Reboot broker.

Mem-boot ulang broker menggunakan AWS CLI

1. Jalankan perintah berikut, ganti *`ClusterArn`*dengan Amazon Resource Name (ARN) yang Anda peroleh ketika Anda membuat cluster Anda, dan *`BrokerId`*dengan ID broker yang ingin Anda reboot.

> **Note**
>
> `reboot-broker`Operasi ini hanya mendukung reboot satu broker pada satu waktu.

Jika Anda tidak memiliki ARN untuk cluster Anda, Anda dapat menemukannya dengan mencantumkan semua cluster. Untuk informasi selengkapnya, lihat the section called “Daftar cluster”.

Jika Anda tidak memiliki ID broker untuk cluster Anda, Anda dapat menemukannya dengan mencantumkan node broker. Untuk informasi selengkapnya, lihat daftar-node.

```
aws kafka reboot-broker --cluster-arn ClusterArn --broker-ids BrokerId
```

Output dari `reboot-broker` operasi ini terlihat seperti JSON berikut.

```
{

    "ClusterArn": "arn:aws:kafka:us-east-1:012345678012:cluster/exampleClusterName/
abcdefab-1234-abcd-5678-cdef0123ab01-2",
    "ClusterOperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-
operation/exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
abcd-4f7f-1234-9876543210ef"
}
```

2. Untuk mendapatkan hasil `reboot-broker` operasi, jalankan perintah berikut, ganti *`ClusterOperationArn`*dengan ARN yang Anda peroleh dalam output perintah. `reboot-broker`

```
aws kafka describe-cluster-operation --cluster-operation-arn ClusterOperationArn
```

Output dari `describe-cluster-operation` perintah ini terlihat seperti contoh JSON berikut.

```
{
    "ClusterOperationInfo": {
        "ClientRequestId": "c0b7af47-8591-45b5-9c0c-909a1a2c99ea",
        "ClusterArn": "arn:aws:kafka:us-east-1:012345678012:cluster/
exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2",
        "CreationTime": "2019-09-25T23:48:04.794Z",
```

```
        "OperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-
operation/exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
abcd-4f7f-1234-9876543210ef",
        "OperationState": "REBOOT_IN_PROGRESS",
        "OperationType": "REBOOT_NODE",
        "SourceClusterInfo": {},
        "TargetClusterInfo": {}
    }
}
```

Ketika operasi reboot selesai, `OperationState` adalah`REBOOT_COMPLETE`.

Mem-boot ulang broker menggunakan API

Untuk me-reboot broker di cluster menggunakan API, lihat RebootBroker.

Dampak restart broker selama patching dan pemeliharaan lainnya

Secara berkala, Amazon MSK memperbarui perangkat lunak pada broker Anda. Pembaruan ini tidak berdampak pada penulisan dan pembacaan aplikasi Anda jika Anda mengikuti praktik terbaik.

Amazon MSK menggunakan pembaruan bergulir untuk perangkat lunak untuk menjaga ketersediaan klaster Anda yang tinggi. Selama proses ini, broker di-reboot satu per satu, dan Kafka secara otomatis memindahkan kepemimpinan ke broker online lain. Klien Kafka memiliki mekanisme bawaan untuk secara otomatis mendeteksi perubahan kepemimpinan untuk partisi dan terus menulis dan membaca data ke dalam cluster MSK.

Setelah broker offline, adalah normal untuk melihat kesalahan pemutusan sementara pada klien Anda. Anda juga akan mengamati untuk jendela singkat (hingga 2 menit, biasanya kurang) beberapa lonjakan di p99 membaca dan menulis latensi (biasanya milidetik tinggi, hingga ~ 2 detik). Lonjakan ini diharapkan dan disebabkan oleh klien yang terhubung kembali ke broker pemimpin baru; itu tidak memengaruhi produk atau konsumsi Anda dan akan menyelesaikan setelah terhubung kembali.

Anda juga akan mengamati peningkatan metrik`UnderReplicatedPartitions`, yang diharapkan karena partisi pada broker yang dimatikan tidak lagi mereplikasi data. Ini tidak berdampak pada penulisan dan pembacaan aplikasi sebagai replika untuk partisi ini yang di-host di broker lain sekarang melayani permintaan.

Setelah pembaruan perangkat lunak, ketika broker kembali online, ia perlu “mengejar” pesan yang dihasilkan saat offline. Selama catch up, Anda juga dapat mengamati peningkatan penggunaan

volume throughput dan CPU. Ini seharusnya tidak berdampak pada penulisan dan pembacaan ke dalam cluster jika Anda memiliki cukup sumber daya CPU, memori, jaringan, dan volume pada broker Anda.

Menandai kluster MSK Amazon

Anda dapat menetapkan metadata Anda sendiri dalam bentuk tag ke sumber daya MSK Amazon, seperti kluster MSK. Tag adalah pasangan kunci-nilai yang Anda tentukan untuk sumber daya. Menggunakan tag adalah cara sederhana namun ampuh untuk mengelola AWS sumber daya dan mengatur data, termasuk data penagihan.

Topik

Dasar-dasar tag

Anda dapat menggunakan Amazon MSK API untuk menyelesaikan tugas-tugas berikut:

- Tambahkan tag ke sumber daya MSK Amazon.
- Buat daftar tag untuk sumber daya MSK Amazon.
- Hapus tag dari sumber daya MSK Amazon.

Anda dapat menggunakan tag untuk mengkategorikan sumber daya MSK Amazon Anda. Misalnya, Anda dapat mengkategorikan kluster MSK Amazon berdasarkan tujuan, pemilik, atau lingkungan. Karena Anda menentukan kunci dan nilai untuk setiap tanda, Anda dapat membuat serangkaian kategori khusus untuk memenuhi kebutuhan spesifik Anda. Misalnya, Anda dapat menentukan satu set tag yang membantu Anda melacak cluster berdasarkan pemilik dan aplikasi terkait.

Berikut ini adalah beberapa contoh tanda:

- `Project: Project name`
- `Owner: Name`

- `Purpose: Load testing`
- `Environment: Production`

Melacak biaya menggunakan penandaan

Anda dapat menggunakan tag untuk mengkategorikan dan melacak biaya Anda AWS . Saat Anda menerapkan tag ke AWS sumber daya Anda, termasuk kluster MSK Amazon, laporan alokasi AWS biaya Anda mencakup penggunaan dan biaya yang dikumpulkan berdasarkan tag. Anda dapat mengatur biaya di berbagai layanan dengan menerapkan tanda yang mewakili kategori bisnis (seperti pusat biaya, nama aplikasi, atau pemilik). Untuk informasi selengkapnya, lihat Menggunakan Tanda Alokasi Biaya untuk Laporan Penagihan Khusus dalam Panduan Pengguna AWS Billing .

Batasan tag

Pembatasan berikut berlaku untuk tag di Amazon MSK.

Batasan dasar

- Jumlah maksimum tanda per sumber daya adalah 50.
- Kunci dan nilai tag peka huruf besar dan kecil.
- Anda tidak dapat mengubah atau mengedit tag untuk sumber daya yang dihapus.

Batasan kunci tanda

- Setiap kunci tanda harus unik. Jika Anda menambahkan tanda dengan kunci yang sudah digunakan, tanda baru akan menimpa pasangan nilai-kunci yang sudah ada.
- Anda tidak dapat memulai kunci tanda dengan `aws:` karena prefiks ini disimpan untuk digunakan oleh AWS. AWS membuat tanda yang dimulai dengan prefiks ini atas nama Anda, tetapi Anda tidak dapat mengedit atau menghapusnya.
- Kunci tanda harus memiliki panjang antara 1 dan 128 karakter Unicode.
- Kunci tanda harus terdiri dari karakter berikut: huruf Unicode, digit, spasi, dan karakter khusus berikut: `_ . / = + - @`.

Batasan nilai tanda

- Panjang nilai tanda harus antara 0 dan 255 karakter Unicode.

- Nilai tanda dapat kosong. Jika tidak, nilai tanda harus terdiri dari karakter berikut: huruf Unicode, digit, spasi, dan salah satu karakter khusus berikut: `_ . / = + - @`.

Menandai sumber daya menggunakan Amazon MSK API

Anda dapat menggunakan operasi berikut untuk menandai atau menghapus tag sumber daya MSK Amazon atau untuk mencantumkan kumpulan tag saat ini untuk sumber daya:

- ListTagsForResource
- TagResource
- UntagResource

Konfigurasi MSK Amazon

Amazon Managed Streaming for Apache Kafka menyediakan konfigurasi default untuk broker, topik, dan node Apache. ZooKeeper Anda juga dapat membuat konfigurasi kustom dan menggunakannya untuk membuat cluster MSK baru atau untuk memperbarui cluster yang ada. Konfigurasi MSK terdiri dari satu set properti dan nilai yang sesuai.

Topik

Konfigurasi MSK kustom

Anda dapat menggunakan Amazon MSK untuk membuat konfigurasi MSK kustom di mana Anda mengatur properti berikut. Properti yang tidak Anda tetapkan secara eksplisit mendapatkan nilai yang mereka miliki. the section called “Konfigurasi default” Untuk informasi selengkapnya tentang properti konfigurasi, lihat Konfigurasi Apache Kafka.

Properti konfigurasi Apache Kafka

Nama	Penjelasan
allow.everyone.if.no.acl.found	Jika Anda ingin mengatur properti ini`false`, pertama-tama pastikan Anda mendefinisikan Apache Kafka ACL untuk cluster Anda. Jika Anda menyetel properti ini `false` dan Anda tidak mendefinisikan Apache Kafka ACL terlebih dahulu, Anda kehilangan akses ke cluster. Jika itu terjadi, Anda dapat memperbar ui konfigurasi lagi dan mengatur properti ini `true` untuk mendapatkan kembali akses ke cluster.

Nama	Penjelasan
auto.create.topics.enable	Mengaktifkan pembuatan otomatis topik di server.
kompresi.type	Jenis kompresi akhir untuk topik tertentu. Anda dapat mengatur properti ini ke codec kompresi standar (`gzip`,, `snappy`, `lz4`, dan`zstd`). Ini juga menerima. `uncompressed` Nilai ini setara dengan tidak ada kompresi. Jika Anda menetapkan nilainya`producer`, itu berarti mempertahankan codec kompresi asli yang ditetapkan oleh produsen.
koneksi.max.idle.ms	Batas waktu koneksi idle dalam milidetik. Thread prosesor soket server menutup koneksi yang menganggur lebih dari nilai yang Anda tetapkan untuk properti ini.
default.replication.factor	Faktor replikasi default untuk topik yang dibuat secara otomatis.
delete.topic.enable	Mengaktifkan operasi menghapus topik. Jika Anda mematikan setelan ini, Anda tidak dapat menghapus topik melalui alat admin.
group.initial.rebalance.delay.ms	Jumlah waktu koordinator grup menunggu lebih banyak konsumen data untuk bergabung dengan grup baru sebelum koordinator grup melakukan penyeimbangan ulang pertama. Penundaan yang lebih lama berarti berpotens i lebih sedikit penyeimbangan kembali, tetapi ini meningkatkan waktu sampai pemrosesan dimulai.

Nama	Penjelasan
group.max.session.timeout.ms	Batas waktu sesi maksimum untuk konsumen terdaftar. Batas waktu yang lebih lama memberi konsumen lebih banyak waktu untuk memproses pesan di antara detak jantung dengan biaya waktu yang lebih lama untuk mendeteksi kegagalan.
group.min.session.timeout.ms	Batas waktu sesi minimum untuk konsumen terdaftar. Batas waktu yang lebih pendek menghasilkan deteksi kegagalan yang lebih cepat dengan mengorbankan detak jantung konsumen yang lebih sering. Ini dapat membanjiri sumber daya broker.
leader.imbalance.per.broker.percentage	Rasio ketidakseimbangan pemimpin diperbolehkan per broker. Pengontrol memicu saldo pemimpin jika melebihi nilai ini per broker. Nilai ini ditentukan dalam persentase.
log.cleaner.delete.retention.ms	Jumlah waktu yang Anda inginkan Apache Kafka untuk menyimpan catatan yang dihapus. Nilai minimum adalah 0.

Nama	Penjelasan
log.cleaner.min.cleanable.ratio	Properti konfigurasi ini dapat memiliki nilai antara 0 dan 1. Nilai ini menentukan seberapa sering pemadat log mencoba membersihkan log (jika pemadatan log diaktifkan). Secara default, Apache Kafka menghindari pembersih an log jika lebih dari 50% log telah dipadatka n. Rasio ini membatasi ruang maksimum yang terbuang log dengan duplikat (pada 50%, ini berarti paling banyak 50% dari log dapat berupa duplikat). Rasio yang lebih tinggi berarti pembersihan yang lebih sedikit dan lebih efisien, tetapi lebih banyak ruang yang terbuang di log.
log.cleanup.policy	Kebijakan pembersihan default untuk segmen di luar jendela retensi. Daftar kebijakan valid yang dipisahkan koma. Kebijakan yang valid adalah `delete` dan`compact`. Untuk klaster berkemampuan Penyimpanan Berjenjang, kebijakan yang valid hanya berlaku. `delete`
log.flush.interval.messages	Jumlah pesan yang terakumulasi pada partisi log sebelum pesan dibuang ke disk.
log.flush.interval.ms	Waktu maksimum dalam milidetik bahwa pesan dalam topik apa pun tetap dalam memori sebelum dibuang ke disk. Jika Anda tidak menetapkan nilai ini, nilai di log.flush.schedule r.interval.ms digunakan. Nilai minimum adalah 0.

Nama	Penjelasan
log.message.timestamp.difference.max.ms	Perbedaan waktu maksimum antara stempel waktu ketika broker menerima pesan dan stempel waktu yang ditentukan dalam pesan. Jika log.message.timestamp.type=CreateTim e, pesan ditolak jika perbedaan stempel waktu melebihi ambang batas ini. Konfigurasi ini diabaikan jika log.message.timestamp.type= Waktu. LogAppend
log.message.timestamp.type	Menentukan apakah stempel waktu dalam pesan adalah waktu pembuatan pesan atau log menambahkan waktu. Nilai yang diizinkan adalah `CreateTime` dan`LogAppendTime` .
log.retention.bytes	Ukuran maksimum log sebelum menghapus nya.
log.retention.hours	Jumlah jam untuk menyimpan file log sebelum menghapusnya, tersier ke properti log.reten tion.ms.
log.retention.minutes	Jumlah menit untuk menyimpan file log sebelum menghapusnya, sekunder ke properti log.retention.ms. Jika Anda tidak menyetel nilai ini, nilai di log.retention.hours akan digunakan.
log.retention.ms	Jumlah milidetik untuk menyimpan file log sebelum menghapusnya (dalam milidetik), Jika tidak disetel, nilai dalam log.retention.minutes digunakan.
log.roll.ms	Waktu maksimum sebelum segmen log baru diluncurkan (dalam milidetik). Jika Anda tidak menyetel properti ini, nilai di log.roll.hours digunakan. Nilai minimum yang mungkin untuk properti ini adalah 1.

Nama	Penjelasan
log.segment.bytes	Ukuran maksimum dari satu file log.
max.incremental.fetch.session.cache.slots	Jumlah maksimum sesi pengambilan inkrement al yang dipertahankan.
message.max.bytes	Ukuran batch rekor terbesar yang diizinkan Kafka. Jika Anda meningkatkan nilai ini dan ada konsumen yang lebih tua dari 0.10.2, Anda juga harus meningkatkan ukuran pengambilan konsumen sehingga mereka dapat mengambil batch rekaman sebesar ini. Versi format pesan terbaru selalu mengelomp okkan pesan ke dalam batch untuk efisiensi . Versi format pesan sebelumnya tidak mengelompokkan catatan yang tidak terkompre si ke dalam batch, dan dalam kasus seperti itu, batas ini hanya berlaku untuk satu rekaman. Anda dapat mengatur nilai ini per topik dengan konfigurasi max.message.bytes level topik.

Nama	Penjelasan
min.insync.replika	Ketika produser menetapkan acks ke `"all"` (or`"-1"`), nilai dalam min.insync.replicas menentukan jumlah minimum replika yang harus mengakui penulisan agar penulisan dianggap berhasil. Jika minimum ini tidak dapat dipenuhi, produsen menimbulkan pengecual ian (salah satu NotEnoughReplicas atau NotEnoughReplicasAfterAppend). Anda dapat menggunakan nilai di min.insyn c.replicas dan acks untuk menerapkan jaminan daya tahan yang lebih besar. Misalnya, Anda dapat membuat topik dengan faktor replikasi 3, menyetel min.insync.replicas ke 2, dan menghasilkan dengan acks of. `"all"` Ini memastikan bahwa produser memunculkan pengecualian jika sebagian besar replika tidak menerima penulisan.
num.io.thread	Jumlah thread yang digunakan server untuk memproses permintaan, yang mungkin termasuk disk I/O.
num.network.threads	Jumlah thread yang digunakan server untuk menerima permintaan dari jaringan dan mengirim tanggapan ke sana.
num.partisi	Jumlah default partisi log per topik.
num.recovery.threads.per.data.dir	Jumlah thread per direktori data yang akan digunakan untuk memulihkan log saat startup dan dan untuk flush mereka saat shutdown.

Nama	Penjelasan
num.replica.fetchers	Jumlah utas fetcher yang digunakan untuk mereplikasi pesan dari broker sumber. Jika Anda meningkatkan nilai ini, Anda dapat meningkatkan derajat paralelisme I/O di broker pengikut.
offsets.retention.minutes	Setelah kelompok konsumen kehilangan semua konsumennya (yaitu, menjadi kosong) offsetnya disimpan untuk periode retensi ini sebelum dibuang. Untuk konsumen mandiri (yaitu, mereka yang menggunakan penugasan manual), offset kedaluwarsa setelah waktu komit terakhir ditambah periode retensi ini.
offsets.topic.replication.factor	Faktor replikasi untuk topik offset. Tetapkan nilai ini lebih tinggi untuk memastikan ketersedi aan. Pembuatan topik internal gagal hingga ukuran cluster memenuhi persyaratan faktor replikasi ini.
replica.fetch.max.bytes	Jumlah byte pesan untuk mencoba untuk mengambil untuk setiap partisi. Ini bukan maksimum absolut. Jika kumpulan rekaman pertama di partisi pengambilan yang tidak kosong pertama lebih besar dari nilai ini, kumpulan rekaman dikembalikan untuk memastikan kemajuan. Message.max.bytes (konfigurasi broker) atau max.message.bytes (konfigurasi topik) mendefinisikan ukuran batch rekaman maksimum yang diterima broker.

Nama	Penjelasan
replica.fetch.response.max.bytes	Jumlah maksimum byte yang diharapkan untuk seluruh respons pengambilan. Rekaman diambil dalam batch, dan jika kumpulan rekaman pertama di partisi pengambilan yang tidak kosong pertama lebih besar dari nilai ini, kumpulan rekaman akan tetap dikembali kan untuk memastikan kemajuan. Ini bukan maksimum absolut. Properti message.m ax.bytes (broker config) atau max.messa ge.bytes (topic config) menentukan ukuran batch record maksimum yang diterima broker.
replica.lag.time.max.ms	Jika pengikut belum mengirim permintaan pengambilan apa pun atau belum mengkonsu msi hingga offset akhir log pemimpin setidakny a dalam jumlah milidetik ini, pemimpin akan menghapus pengikut dari ISR. MinValue: 10000 MaxValue = 30000
replica.selector.class	Nama kelas yang sepenuhnya memenuhi syarat yang mengimplementasikan. ReplicaSe lector Broker menggunakan nilai ini untuk menemukan replika baca yang disukai. Jika Anda menggunakan Apache Kafka versi 2.4.1 atau lebih tinggi, dan ingin mengizinkan konsumen mengambil dari replika terdekat, atur properti ini ke. `org.apache.kafka.c ommon.replica.RackAwareRepl icaSelector` Untuk informasi selengkap nya, lihat the section called “Apache Kafka versi 2.4.1 (gunakan 2.4.1.1 sebagai gantinya)”.

Nama	Penjelasan
replica.socket.receive.buffer.bytes	Soket menerima buffer untuk permintaan jaringan.
socket.receive.buffer.bytes	Buffer SO_RCVBUF dari soket server soket. Nilai minimum yang dapat Anda atur untuk properti ini adalah -1. Jika nilainya -1, Amazon MSK menggunakan default OS.
socket.request.max.bytes	Jumlah maksimum byte dalam permintaan soket.
socket.send.buffer.bytes	Buffer SO_SNDBUF dari soket server soket. Nilai minimum yang dapat Anda atur untuk properti ini adalah -1. Jika nilainya -1, Amazon MSK menggunakan default OS.
transaksi.max.timeout.ms	Batas waktu maksimum untuk transaksi. Jika waktu transaksi yang diminta dari klien melebihi nilai ini, broker mengembalikan kesalahan InitProducerIdRequest. Ini mencegah klien dari batas waktu yang terlalu besar, dan ini dapat menghambat konsumen yang membaca dari topik yang termasuk dalam transaksi.
transaction.state.log.min.isr	Mengganti konfigurasi min.insync.replicas untuk topik transaksi.
transaction.state.log.replication.factor	Faktor replikasi untuk topik transaksi. Tetapkan properti ini ke nilai yang lebih tinggi untuk meningkatkan ketersediaan. Pembuatan topik internal gagal hingga ukuran cluster memenuhi persyaratan faktor replikasi ini.

Nama	Penjelasan
transactional.id.expiration.ms	Waktu dalam milidetik koordinator transaksi menunggu untuk menerima pembaruan status transaksi untuk transaksi saat ini sebelum koordinator kedaluwarsa ID transaksionalnya. Pengaturan ini juga memengaruhi kedaluwarsa ID produsen karena menyebabkan ID produsen kedaluwarsa saat waktu ini berlalu setelah penulisan terakhir dengan ID produser yang diberikan. ID produser mungkin kedaluwar sa lebih cepat jika penulisan terakhir dari ID produsen dihapus karena pengaturan retensi untuk topik tersebut. Nilai minimum untuk properti ini adalah 1 milidetik.
unclean.leader.election.enable	Menunjukkan jika replika yang tidak ada dalam set ISR harus berfungsi sebagai pemimpin sebagai upaya terakhir, meskipun ini dapat mengakibatkan hilangnya data.
zookeeper.connection.timeout.ms	ZooKeeper kluster mode. Waktu maksimum yang ditunggu klien untuk membuat koneksi. ZooKeeper Jika Anda tidak menetapkan nilai ini, nilai di zookeeper.session.timeout.ms digunakan. MinValue = 6000 MaxValue (inklusif) = 18000
zookeeper.session.timeout.ms	ZooKeeper kluster mode. Batas waktu ZooKeeper sesi Apache dalam milidetik. MinValue = 6000 MaxValue (inklusif) = 18000

Untuk mempelajari cara membuat konfigurasi MSK kustom, daftar semua konfigurasi, atau jelaskan, lihat. the section called “Operasi konfigurasi” Untuk membuat klaster MSK dengan konfigurasi MSK kustom, atau untuk memperbarui cluster dengan konfigurasi kustom baru, lihat. Cara kerjanya

Saat Anda memperbarui klaster MSK yang ada dengan konfigurasi MSK khusus, Amazon MSK melakukan rolling restart bila diperlukan, dan menggunakan praktik terbaik untuk meminimalkan waktu henti pelanggan. Misalnya, setelah Amazon MSK memulai ulang setiap broker, Amazon MSK mencoba membiarkan broker menangkap data yang mungkin terlewatkan oleh broker selama pembaruan konfigurasi sebelum pindah ke broker berikutnya.

Konfigurasi dinamis

Selain properti konfigurasi yang disediakan Amazon MSK, Anda dapat secara dinamis mengatur properti konfigurasi tingkat klaster dan tingkat broker yang tidak memerlukan restart broker. Anda dapat secara dinamis mengatur beberapa properti konfigurasi. Ini adalah properti yang tidak ditandai sebagai hanya-baca dalam tabel di bawah Konfigurasi Broker dalam dokumentasi Apache Kafka. Untuk informasi tentang konfigurasi dinamis dan perintah contoh, lihat Memperbarui Konfigurasi Broker dalam dokumentasi Apache Kafka.

> **Note**
>
> Anda dapat mengatur `advertised.listeners` properti, tetapi bukan `listeners` properti.

Konfigurasi tingkat topik

Anda dapat menggunakan perintah Apache Kafka untuk mengatur atau memodifikasi properti konfigurasi tingkat topik untuk topik baru dan yang sudah ada. Untuk informasi selengkapnya tentang properti konfigurasi tingkat topik dan contoh tentang cara mengaturnya, lihat Konfigurasi Tingkat Topik dalam dokumentasi Apache Kafka.

Status konfigurasi

Konfigurasi MSK Amazon dapat berada di salah satu negara bagian berikut. Untuk melakukan operasi pada konfigurasi, konfigurasi harus dalam `DELETE_FAILED` keadaan `ACTIVE` atau:

- `ACTIVE`
- `DELETING`
- `DELETE_FAILED`

Konfigurasi MSK Amazon default

Bila Anda membuat kluster MSK dan tidak menentukan konfigurasi MSK kustom, Amazon MSK membuat dan menggunakan konfigurasi default dengan nilai yang ditunjukkan dalam tabel berikut. Untuk properti yang tidak ada dalam tabel ini, Amazon MSK menggunakan default yang terkait dengan versi Apache Kafka Anda. Untuk daftar nilai default ini, lihat Konfigurasi Apache Kafka.

Nilai konfigurasi default

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
allow.everyone.if.no.acl.found	Jika tidak ada pola sumber daya yang cocok dengan sumber daya tertentu, sumber daya tidak memiliki ACL terkait. Dalam hal ini, jika Anda menyetel properti ini`true`, semua pengguna dapat mengakses sumber daya, bukan hanya pengguna super.	`true`	`true`
auto.create.topics.enable	Mengaktifkan pembuatan otomatis topik di server.	`false`	`false`
auto.leader.rebalance.enable	Memungkinkan penyeimbangan pemimpin otomatis. Benang latar belakang memeriksa dan memulai	`true`	`true`

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
	keseimbangan pemimpin secara berkala, jika perlu.		
default.replication.factor	Faktor replikasi default untuk topik yang dibuat secara otomatis.	3 untuk cluster di 3 Availability Zone, dan 2 untuk cluster di 2 Availability Zone.	3 untuk cluster di 3 Availability Zone, dan 2 untuk cluster di 2 Availability Zone.

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
local.retention.bytes	Ukuran maksimum segmen log lokal untuk partisi sebelum menghapus segmen lama. Jika Anda tidak menetapkan nilai ini, nilai dalam log.retention.bytes akan digunakan. Nilai efektif harus selalu kurang dari atau sama dengan nilai log.retention.bytes. Nilai default -2 menunjukkan bahwa tidak ada batasan retensi lokal. Ini sesuai dengan pengaturan retensi.ms/bytes -1. Properti local.retention.ms dan local.retention.bytes mirip dengan log.retention karena digunakan untuk menentukan berapa lama segmen log harus tetap berada di penyimpanan lokal. Konfigurasi log.retention.* yang ada adalah konfigurasi retensi	-2 untuk tak terbatas	-2 untuk tak terbatas

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
	untuk partisi topik. Ini termasuk penyimpanan lokal dan jarak jauh. Nilai yang valid: bilangan bulat di [-2; +Inf]		

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
local.retention.ms	Jumlah milidetik untuk mempertahankan segmen log lokal sebelum penghapusan. Jika Anda tidak menetapkan nilai ini, Amazon MSK menggunakan nilai di log.retention.ms. Nilai efektif harus selalu kurang dari atau sama dengan nilai log.retention.bytes. Nilai default -2 menunjukkan bahwa tidak ada batasan retensi lokal. Ini sesuai dengan pengaturan retensi.ms/bytes -1. Nilai local.retention.ms dan local.retention.bytes mirip dengan log.retention. MSK menggunakan konfigurasi ini untuk menentukan berapa lama segmen log harus tetap berada di penyimpanan lokal. Konfigurasi log.retention.* yang	-2 untuk tak terbatas	-2 untuk tak terbatas

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
	ada adalah konfigurasi retensi untuk partisi topik. Ini termasuk penyimpanan lokal dan jarak jauh. Nilai yang valid adalah bilangan bulat yang lebih besar dari 0.		

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
log.message.timestamp.difference.max.ms	Perbedaan maksimum yang diperbolehkan antara stempel waktu ketika broker menerima pesan dan stempel waktu yang ditentukan dalam pesan. Jika log.message.timestamp.type=CreateTime, pesan akan ditolak jika perbedaan stempel waktu melebihi ambang batas ini. Konfigurasi ini diabaikan jika log.message.timestamp.type=Waktu. LogAppend Perbedaan stempel waktu maksimum yang diizinkan tidak boleh lebih besar dari log.retention.ms untuk menghindari penggulungan log yang tidak perlu sering.	9223372036854775807	86400000 untuk Kafka 2.8.2.tiered
log.segment.bytes	Ukuran maksimum dari satu file log.	1073741824	134217728

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
min.insync.replika	Ketika produsen menetapkan nilai acks (produser pengakuan mendapat dari broker Kafka) ke `"all"` (atau`"-1"`), nilai dalam min.insync.replicas menentukan jumlah minimum replika yang harus mengakui penulisan agar penulisan dianggap berhasil. Jika nilai ini tidak memenuhi minimum ini, produsen memunculkan pengecualian (salah satu NotEnough Replicas atau NotEnoughReplicasAfterAppend). Saat Anda menggunakan nilai di min.insync.replicas dan acks bersama-sama, Anda dapat menerapkan jaminan daya tahan yang lebih besar. Misalnya, Anda dapat membuat	2 untuk cluster di 3 Availability Zone, dan 1 untuk cluster di 2 Availability Zone.	2 untuk cluster di 3 Availability Zone, dan 1 untuk cluster di 2 Availability Zone.

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
	topik dengan faktor replikasi 3, menyetel min.insync.replicas ke 2, dan menghasilkan dengan acks of. `"all"` Ini memastikan bahwa produser memunculkan pengecualian jika sebagian besar replika tidak menerima penulisan.		
num.io.thread	Jumlah thread yang digunakan server untuk menghasilkan permintaan, yang mungkin termasuk disk I/O.	8	maks (8, vCPU) di mana vCPU bergantung pada ukuran instance broker
num.network.threads	Jumlah thread yang digunakan server untuk menerima permintaan dari jaringan danmengirim tanggapan ke jaringan.	5	maks (5, vCPU/2) di mana vCPU bergantung pada ukuran instance broker
num.partisi	Jumlah default partisi log per topik.	1	1

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
num.replica.fetchers	Jumlah utas fetcher yang digunakan untuk mereplikasi pesan dari broker sumber.Jika Anda meningkatkan nilai ini, Anda dapat meningkatkan derajat paralelisme I/O di broker pengikut.	2	maks (2, vCPU/4) di mana vCPU bergantung pada ukuran instance broker
remote.log.msk.disable.policy	Digunakan dengan remote.storage.enable untuk menonaktifkan penyimpanan berjenjang. Setel kebijakan ini ke Hapus, untuk menunjukkan bahwa data dalam penyimpanan berjenjang dihapus saat Anda menyetel remote.storage.enable ke false.	N/A	HAPUS

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
remote.log.reader.threads	Ukuran kumpulan utas pembaca log jarak jauh, yang digunakan dalam tugas penjadwalan untuk mengambil data dari penyimpanan jarak jauh.	N/A	maks (10, vCPU* 0.67) di mana vCPU bergantung pada ukuran instance broker

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
remote.storage.enable	Mengaktifkan penyimpanan berjenjang (jarak jauh) untuk topik jika disetel ke true. Menonaktifkan penyimpanan berjenjang tingkat topik jika disetel ke false dan remote.log.msk.disable.policy disetel ke Hapus. Saat Anda menonaktifkan penyimpanan berjenjang, Anda menghapus data dari penyimpanan jarak jauh. Saat Anda menonaktifkan penyimpanan berjenjang untuk suatu topik, Anda tidak dapat mengaktifkannya lagi.	SALAH	true

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
replica.lag.time.max.ms	Jika pengikut belum mengirim permintaan pengambilan apa pun atau belum mengkonsumsi hingga offset akhir log pemimpin setidaknya dalam jumlah milidetik ini, pemimpin akan menghapus pengikut dari ISR.	30000	30000

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
retensi.ms	Bidang wajib. Waktu minimum adalah 3 hari. Tidak ada default karena pengaturannya wajib. Amazon MSK menggunakan nilai retention.ms dengan local.retention.ms untuk menentukan kapan data berpindah dari penyimpanan lokal ke penyimpanan berjenjang. Nilai local.retention.ms menentukan kapan harus memindahkan data dari penyimpanan lokal ke berjenjang. Nilai retention.ms menentukan kapan harus menghapus data dari penyimpanan berjenjang (yaitu, dihapus dari cluster). Nilai yang valid: bilangan bulat di [-1; +Inf]	Minimum 259.200.000 milidetik (3 hari). -1 untuk retensi tak terbatas.	Minimum 259.200.000 milidetik (3 hari). -1 untuk retensi tak terbatas.

Nama	Penjelasan	Nilai default untuk cluster penyimpanan non-tier	Nilai default untuk cluster berkemampuan penyimpanan berjenjang
socket.receive.buffer.bytes	Buffer SO_RCVBUF dari soket pemutus soket. Jika nilainya -1, default OS digunakan.	102400	102400
socket.request.max.bytes	Jumlah maksimum byte dalam permintaan soket.	104857600	104857600
socket.send.buffer.bytes	Buffer SO_SNDBUF dari soket pemutus soket. Jika nilainya -1, default OS digunakan.	102400	102400
unclean.leader.election.enable	Menunjukkan jika Anda ingin replika yang tidak ada dalam set ISR untuk berfungsi sebagai pemimpin sebagai upaya terakhir, meskipun ini dapat mengakibatkan kehilangan data.	true	SALAH
zookeeper.session.timeout.ms	Batas waktu ZooKeeper sesi Apache dalam milidetik.	18000	18000
zookeeper.set.acl	Klien set untuk menggunakan ACL aman.	false	false

Untuk informasi tentang cara menentukan nilai konfigurasi kustom, lihatthe section called “Konfigurasi kustom”.

Pedoman untuk konfigurasi tingkat topik penyimpanan berjenjang

Berikut ini adalah pengaturan dan batasan default saat Anda mengonfigurasi penyimpanan berjenjang di tingkat topik.

- Amazon MSK tidak mendukung ukuran segmen log yang lebih kecil untuk topik dengan penyimpanan berjenjang diaktifkan. Jika Anda ingin membuat segmen, ada ukuran segmen log minimum 48 MiB, atau waktu roll segmen minimum 10 menit. Nilai-nilai ini dipetakan ke properti segment.bytes dan segment.ms.
- Nilai local.retention.ms/bytes tidak dapat sama atau melebihi retensi.ms/bytes. Ini adalah pengaturan retensi penyimpanan berjenjang.
- Nilai default untuk local.retention.ms/bytes adalah -2. Ini berarti bahwa nilai retention.ms digunakan untuk local.retention.ms/bytes. Dalam hal ini, data tetap berada di penyimpanan lokal dan penyimpanan berjenjang (masing-masing satu salinan), dan semuanya kedaluwarsa bersama. Untuk opsi ini, salinan data lokal disimpan ke penyimpanan jarak jauh. Dalam hal ini, data yang dibaca dari lalu lintas konsumsi berasal dari penyimpanan lokal.
- Nilai default untuk retention.ms adalah 7 hari. Tidak ada batasan ukuran default untuk retention.bytes.
- Nilai minimum untuk retention.ms/bytes adalah -1. Ini berarti retensi tak terbatas.
- Nilai minimum untuk local.retention.ms/bytes adalah -2. Ini berarti retensi tak terbatas untuk penyimpanan lokal. Ini cocok dengan pengaturan retention.ms/bytes sebagai -1.
- Retensi konfigurasi tingkat topik.ms wajib untuk topik dengan penyimpanan berjenjang diaktifkan. Retensi minimum.ms adalah 3 hari.

Operasi konfigurasi MSK Amazon

Topik ini menjelaskan cara membuat konfigurasi MSK khusus dan cara melakukan operasi pada mereka. Untuk informasi tentang cara menggunakan konfigurasi MSK untuk membuat atau memperbarui cluster, lihat. Cara kerjanya

Topik ini berisi bagian-bagian berikut:

- Untuk membuat konfigurasi MSK

- Untuk memperbarui konfigurasi MSK
- Untuk menghapus konfigurasi MSK
- Untuk menggambarkan konfigurasi MSK
- Untuk menggambarkan revisi konfigurasi MSK
- Untuk mencantumkan semua konfigurasi MSK di akun Anda untuk Wilayah saat ini

Untuk membuat konfigurasi MSK

1. Buat file tempat Anda menentukan properti konfigurasi yang ingin Anda atur dan nilai yang ingin Anda tetapkan padanya. Berikut ini adalah isi dari contoh file konfigurasi.

```
auto.create.topics.enable = true

log.roll.ms = 604800000
```

2. Jalankan AWS CLI perintah berikut, dan ganti *config-file-path* dengan path ke file tempat Anda menyimpan konfigurasi di langkah sebelumnya.

> **Note**
>
> Nama yang Anda pilih untuk konfigurasi Anda harus cocok dengan regex berikut: "^ [0-9A-za-Z] [0-9A-za-Z-] {0,} $".

```
aws kafka create-configuration --name "ExampleConfigurationName" --description
 "Example configuration description." --kafka-versions "1.1.1" --server-properties
 fileb://config-file-path
```

Berikut ini adalah contoh respons yang berhasil setelah Anda menjalankan perintah ini.

```
{
    "Arn": "arn:aws:kafka:us-east-1:123456789012:configuration/SomeTest/
abcdabcd-1234-abcd-1234-abcd123e8e8e-1",
    "CreationTime": "2019-05-21T19:37:40.626Z",
    "LatestRevision": {
        "CreationTime": "2019-05-21T19:37:40.626Z",
        "Description": "Example configuration description.",
        "Revision": 1
```

```
    },
    "Name": "ExampleConfigurationName"
}
```

3. Perintah sebelumnya mengembalikan Amazon Resource Name (ARN) untuk konfigurasi baru Anda. Simpan ARN ini karena Anda membutuhkannya untuk merujuk ke konfigurasi ini di perintah lain. Jika Anda kehilangan konfigurasi ARN, Anda dapat membuat daftar semua konfigurasi di akun Anda untuk menemukannya lagi.

Untuk memperbarui konfigurasi MSK

1. Buat file tempat Anda menentukan properti konfigurasi yang ingin Anda perbarui dan nilai yang ingin Anda tetapkan padanya. Berikut ini adalah isi dari contoh file konfigurasi.

```
auto.create.topics.enable = true

min.insync.replicas = 2
```

2. Jalankan AWS CLI perintah berikut, dan ganti *`config-file-path`* dengan path ke file tempat Anda menyimpan konfigurasi di langkah sebelumnya.

   Ganti *`konfigurasi-arn`* dengan ARN yang Anda peroleh saat membuat konfigurasi. Jika Anda tidak menyimpan ARN saat membuat konfigurasi, Anda dapat menggunakan `list-configurations` perintah untuk mencantumkan semua konfigurasi di akun Anda. Konfigurasi yang Anda inginkan dalam daftar muncul di respons. ARN konfigurasi juga muncul dalam daftar itu.

```
aws kafka update-configuration --arn configuration-arn --description "Example
 configuration revision description." --server-properties fileb://config-file-path
```

3. Berikut ini adalah contoh respons yang berhasil setelah Anda menjalankan perintah ini.

```
{
    "Arn": "arn:aws:kafka:us-east-1:123456789012:configuration/SomeTest/
abcdabcd-1234-abcd-1234-abcd123e8e8e-1",
    "LatestRevision": {
        "CreationTime": "2020-08-27T19:37:40.626Z",
        "Description": "Example configuration revision description.",
        "Revision": 2
    }
```

```
}
```

Untuk menghapus konfigurasi MSK

Prosedur berikut menunjukkan cara menghapus konfigurasi yang tidak dilampirkan ke cluster. Anda tidak dapat menghapus konfigurasi yang dilampirkan ke klaster.

1. Untuk menjalankan contoh ini, ganti *configuration-arn* dengan ARN yang Anda peroleh saat membuat konfigurasi. Jika Anda tidak menyimpan ARN saat membuat konfigurasi, Anda dapat menggunakan `list-configurations` perintah untuk mencantumkan semua konfigurasi di akun Anda. Konfigurasi yang Anda inginkan dalam daftar muncul di respons. ARN konfigurasi juga muncul dalam daftar itu.

   ```
   aws kafka delete-configuration --arn configuration-arn
   ```

2. Berikut ini adalah contoh respons yang berhasil setelah Anda menjalankan perintah ini.

   ```
   {
       "arn": " arn:aws:kafka:us-east-1:123456789012:configuration/SomeTest/
   abcdabcd-1234-abcd-1234-abcd123e8e8e-1",
       "state": "DELETING"
   }
   ```

Untuk menggambarkan konfigurasi MSK

1. Perintah berikut mengembalikan metadata tentang konfigurasi. Untuk mendapatkan deskripsi terperinci tentang konfigurasi, jalankan file`describe-configuration-revision`.

   Untuk menjalankan contoh ini, ganti *configuration-arn* dengan ARN yang Anda peroleh saat membuat konfigurasi. Jika Anda tidak menyimpan ARN saat membuat konfigurasi, Anda dapat menggunakan `list-configurations` perintah untuk mencantumkan semua konfigurasi di akun Anda. Konfigurasi yang Anda inginkan dalam daftar muncul di respons. ARN konfigurasi juga muncul dalam daftar itu.

   ```
   aws kafka describe-configuration --arn configuration-arn
   ```

2. Berikut ini adalah contoh respons yang berhasil setelah Anda menjalankan perintah ini.

```
{
    "Arn": "arn:aws:kafka:us-east-1:123456789012:configuration/SomeTest/abcdabcd-
abcd-1234-abcd-abcd123e8e8e-1",
    "CreationTime": "2019-05-21T00:54:23.591Z",
    "Description": "Example configuration description.",
    "KafkaVersions": [
        "1.1.1"
    ],
    "LatestRevision": {
        "CreationTime": "2019-05-21T00:54:23.591Z",
        "Description": "Example configuration description.",
        "Revision": 1
    },
    "Name": "SomeTest"
}
```

Untuk menggambarkan revisi konfigurasi MSK

Jika Anda menggunakan `describe-configuration` perintah untuk menggambarkan konfigurasi MSK, Anda melihat metadata konfigurasi. Untuk mendapatkan deskripsi konfigurasi, gunakan perintah,`describe-configuration-revision`.

- Jalankan perintah berikut dan ganti *`configuration-arn`* dengan ARN yang Anda peroleh saat membuat konfigurasi. Jika Anda tidak menyimpan ARN saat membuat konfigurasi, Anda dapat menggunakan `list-configurations` perintah untuk mencantumkan semua konfigurasi di akun Anda. Konfigurasi yang Anda inginkan dalam daftar yang muncul dalam respons. ARN konfigurasi juga muncul dalam daftar itu.

  ```
  aws kafka describe-configuration-revision --arn configuration-arn --revision 1
  ```

  Berikut ini adalah contoh respons yang berhasil setelah Anda menjalankan perintah ini.

  ```
  {
      "Arn": "arn:aws:kafka:us-east-1:123456789012:configuration/SomeTest/abcdabcd-
  abcd-1234-abcd-abcd123e8e8e-1",
      "CreationTime": "2019-05-21T00:54:23.591Z",
      "Description": "Example configuration description.",
      "Revision": 1,
  ```

```
    "ServerProperties":
 "YXV0by5jcmVhdGUudG9waWNzLmVuYWJsZSA9IHRydWUKCgp6b29rZWVwZXIuY29ubmVjdGlvbi50aW1lb3V0Lm1zI
}
```

Nilai `ServerProperties` dikodekan dengan base64. Jika Anda menggunakan decoder base64 (misalnya, https://www.base64decode.org/) untuk memecahkan kode secara manual, Anda mendapatkan konten dari file konfigurasi asli yang Anda gunakan untuk membuat konfigurasi kustom. Dalam hal ini, Anda mendapatkan yang berikut:

```
auto.create.topics.enable = true

log.roll.ms = 604800000
```

Untuk mencantumkan semua konfigurasi MSK di akun Anda untuk Wilayah saat ini

- Jalankan perintah berikut.

```
aws kafka list-configurations
```

Berikut ini adalah contoh respons yang berhasil setelah Anda menjalankan perintah ini.

```
{
    "Configurations": [
        {
            "Arn": "arn:aws:kafka:us-east-1:123456789012:configuration/SomeTest/
abcdabcd-abcd-1234-abcd-abcd123e8e8e-1",
            "CreationTime": "2019-05-21T00:54:23.591Z",
            "Description": "Example configuration description.",
            "KafkaVersions": [
                "1.1.1"
            ],
            "LatestRevision": {
                "CreationTime": "2019-05-21T00:54:23.591Z",
                "Description": "Example configuration description.",
                "Revision": 1
            },
            "Name": "SomeTest"
        },
```

```
        {
            "Arn": "arn:aws:kafka:us-east-1:123456789012:configuration/SomeTest/
abcdabcd-1234-abcd-1234-abcd123e8e8e-1",
            "CreationTime": "2019-05-03T23:08:29.446Z",
            "Description": "Example configuration description.",
            "KafkaVersions": [
                "1.1.1"
            ],
            "LatestRevision": {
                "CreationTime": "2019-05-03T23:08:29.446Z",
                "Description": "Example configuration description.",
                "Revision": 1
            },
            "Name": "ExampleConfigurationName"
        }
    ]
}
```

MSK Tanpa Server

> **Note**
>
> MSK Tanpa Server tersedia di AS Timur (Ohio), AS Timur (Virginia N.), AS Barat (Oregon), Kanada (Tengah), Asia Pasifik (Mumbai), Asia Pasifik (Singapura), Asia Pasifik (Sydney), Asia Pasifik (Tokyo), Asia Pasifik (Seoul), Eropa (Frankfurt), Eropa (Stockholm), Eropa (Irlandia), Eropa (Paris), dan Eropa (London) Daerah.

MSK Serverless adalah tipe cluster untuk Amazon MSK yang memungkinkan Anda menjalankan Apache Kafka tanpa harus mengelola dan menskalakan kapasitas cluster. Ini secara otomatis menyediakan dan menskalakan kapasitas saat mengelola partisi dalam topik Anda, sehingga Anda dapat mengalirkan data tanpa memikirkan ukuran yang tepat atau penskalaan cluster. MSK Serverless menawarkan model harga berbasis throughput, jadi Anda hanya membayar untuk apa yang Anda gunakan. Pertimbangkan untuk menggunakan klaster tanpa server jika aplikasi Anda memerlukan kapasitas streaming sesuai permintaan yang menskalakan naik dan turun secara otomatis.

MSK Serverless sepenuhnya kompatibel dengan Apache Kafka, sehingga Anda dapat menggunakan aplikasi klien yang kompatibel untuk menghasilkan dan mengkonsumsi data. Ini juga terintegrasi dengan layanan berikut:

- AWS PrivateLink untuk menyediakan konektivitas pribadi
- AWS Identity and Access Management (IAM) untuk otentikasi dan otorisasi menggunakan bahasa Java dan non-Java. Untuk instruksi tentang mengkonfigurasi klien untuk IAM, lihat. Konfigurasikan klien untuk kontrol akses IAM
- AWS Glue Schema Registry untuk manajemen skema
- Amazon Managed Service untuk Apache Flink untuk pemrosesan aliran berbasis Apache Flink
- AWS Lambda untuk pemrosesan acara

> **Note**
>
> MSK Tanpa Server memerlukan kontrol akses IAM untuk semua cluster. Apache Kafka Access Control List (ACL) tidak didukung. Untuk informasi selengkapnya, lihat the section called “Kontrol akses IAM”.

Untuk informasi tentang kuota layanan yang berlaku untuk MSK Serverless, lihat. the section called “Kuota untuk klaster tanpa server”

Untuk membantu Anda memulai dengan klaster tanpa server, dan untuk mempelajari lebih lanjut tentang opsi konfigurasi dan pemantauan untuk klaster tanpa server, lihat berikut ini.

Topik

- Memulai menggunakan kluster MSK Tanpa Server
- Konfigurasi untuk cluster tanpa server
- Memantau cluster tanpa server

Memulai menggunakan kluster MSK Tanpa Server

Tutorial ini menunjukkan contoh bagaimana Anda dapat membuat kluster MSK Tanpa Server, membuat mesin klien yang dapat mengaksesnya, dan menggunakan klien untuk membuat topik di cluster dan menulis data ke topik tersebut. Latihan ini tidak mewakili semua opsi yang dapat Anda pilih saat membuat cluster tanpa server. Di berbagai bagian latihan ini, kami memilih opsi default untuk kesederhanaan. Ini tidak berarti bahwa mereka adalah satu-satunya opsi yang berfungsi untuk menyiapkan cluster tanpa server. Anda juga dapat menggunakan AWS CLI atau Amazon MSK API. Untuk informasi selengkapnya, lihat Amazon MSK API Referensi 2.0.

Topik

- Langkah 1: Buat kluster MSK Tanpa Server
- Langkah 2: Buat peran IAM
- Langkah 3: Buat mesin klien
- Langkah 4: Buat topik Apache Kafka
- Langkah 5: Menghasilkan dan Mengkonsumsi Data
- Langkah 6: Hapus sumber daya

Langkah 1: Buat kluster MSK Tanpa Server

Pada langkah ini, Anda melakukan dua tugas. Pertama, Anda membuat kluster MSK Tanpa Server dengan pengaturan default. Kedua, Anda mengumpulkan informasi tentang cluster. Ini adalah

informasi yang Anda butuhkan di langkah selanjutnya ketika Anda membuat klien yang dapat mengirim data ke cluster.

Untuk membuat klaster tanpa server

1. Masuk ke AWS Management Console, dan buka konsol MSK Amazon di https://console.aws.amazon.com/msk/home.
2. Pilih Buat klaster.
3. Untuk metode Creation, biarkan opsi Quick create dipilih. Opsi Quick create memungkinkan Anda membuat cluster tanpa server dengan pengaturan default.
4. Untuk nama Cluster, masukkan nama deskriptif, seperti**`msk-serverless-tutorial-cluster`**.
5. Untuk properti klaster Umum, pilih Tanpa Server sebagai tipe Cluster. Gunakan nilai default untuk properti cluster Umum yang tersisa.
6. Perhatikan tabel di bawah Semua pengaturan cluster. Tabel ini mencantumkan nilai default untuk pengaturan penting seperti jaringan dan ketersediaan, dan menunjukkan apakah Anda dapat mengubah setiap pengaturan setelah Anda membuat cluster. Untuk mengubah pengaturan sebelum Anda membuat cluster, Anda harus memilih opsi Custom create di bawah metode Creation.

   > **Note**
   >
   > Anda dapat menghubungkan klien dari hingga lima VPC berbeda dengan kluster MSK Tanpa Server. Untuk membantu aplikasi klien beralih ke Availability Zone lain jika terjadi pemadaman, Anda harus menentukan setidaknya dua subnet di setiap VPC.

7. Pilih Buat klaster.

Untuk mengumpulkan informasi tentang cluster

1. Di bagian Ringkasan klaster, pilih Lihat informasi klien. Tombol ini tetap berwarna abu-abu sampai Amazon MSK selesai membuat cluster. Anda mungkin perlu menunggu beberapa menit sampai tombol menjadi aktif sehingga Anda dapat menggunakannya.
2. Salin string di bawah label Endpoint. Ini adalah string server bootstrap Anda.
3. Pilih tab Properti.

4. Di bawah bagian Pengaturan jaringan, salin ID subnet dan grup keamanan dan simpan karena Anda memerlukan informasi ini nanti untuk membuat mesin klien.
5. Pilih salah satu subnet. Ini membuka Konsol VPC Amazon. Temukan ID VPC Amazon yang terkait dengan subnet. Simpan ID VPC Amazon ini untuk digunakan nanti.

Langkah Selanjutnya

Langkah 2: Buat peran IAM

Langkah 2: Buat peran IAM

Pada langkah ini, Anda melakukan dua tugas. Tugas pertama adalah membuat kebijakan IAM yang memberikan akses untuk membuat topik di cluster dan mengirim data ke topik tersebut. Tugas kedua adalah membuat peran IAM dan mengaitkan kebijakan ini dengannya. Pada langkah selanjutnya, kami membuat mesin klien yang mengasumsikan peran ini dan menggunakannya untuk membuat topik di cluster dan mengirim data ke topik itu.

Untuk membuat kebijakan IAM yang memungkinkan untuk membuat topik dan menulis kepada mereka

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, pilih Kebijakan.
3. Pilih Buat Kebijakan.
4. Pilih tab JSON, lalu ganti JSON di jendela editor dengan JSON berikut.

   Ganti *wilayah* dengan kode Wilayah AWS tempat Anda membuat cluster Anda. Ganti *Account-ID dengan ID* akun Anda. Ganti *msk-serverless-tutorial-cluster*dengan nama cluster tanpa server Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:Connect",
                "kafka-cluster:AlterCluster",
                "kafka-cluster:DescribeCluster"
            ],
```

```
            "Resource": [
                "arn:aws:kafka:region:Account-ID:cluster/msk-serverless-tutorial-
cluster/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:*Topic*",
                "kafka-cluster:WriteData",
                "kafka-cluster:ReadData"
            ],
            "Resource": [
                "arn:aws:kafka:region:Account-ID:topic/msk-serverless-tutorial-
cluster/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:AlterGroup",
                "kafka-cluster:DescribeGroup"
            ],
            "Resource": [
                "arn:aws:kafka:region:Account-ID:group/msk-serverless-tutorial-
cluster/*"
            ]
        }
    ]
}
```

Untuk petunjuk tentang cara menulis kebijakan aman, lihatthe section called “Kontrol akses IAM”.

5. Pilih Berikutnya: Tanda.
6. Pilih Berikutnya: Tinjau.
7. Untuk nama kebijakan, masukkan nama deskriptif, seperti**msk-serverless-tutorial-policy**.
8. Pilih Buat kebijakan.

Untuk membuat peran IAM dan melampirkan kebijakan padanya

1. Pada panel navigasi, pilih Peran.

2. Pilih Buat peran.
3. Di bawah Kasus penggunaan umum, pilih EC2, lalu pilih Berikutnya: Izin.
4. Di kotak pencarian, masukkan nama kebijakan yang sebelumnya Anda buat untuk tutorial ini. Kemudian pilih kotak di sebelah kiri kebijakan.
5. Pilih Berikutnya: Tanda.
6. Pilih Berikutnya: Tinjau.
7. Untuk nama peran, masukkan nama deskriptif, seperti**`msk-serverless-tutorial-role`**.
8. Pilih Buat peran.

Langkah Selanjutnya

Langkah 3: Buat mesin klien

Langkah 3: Buat mesin klien

Pada langkahnya, Anda melakukan dua tugas. Tugas pertama adalah membuat instans Amazon EC2 untuk digunakan sebagai mesin klien Apache Kafka. Tugas kedua adalah menginstal alat Java dan Apache Kafka pada mesin.

Untuk membuat mesin klien

1. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
2. Pilih Luncurkan instans.
3. Masukkan Nama deskriptif untuk mesin klien Anda, seperti**`msk-serverless-tutorial-client`**.
4. Biarkan Amazon Linux 2 AMI (HVM) - Kernel 5.10, Jenis Volume SSD dipilih untuk jenis Amazon Machine Image (AMI).
5. Biarkan tipe instans t2.micro dipilih.
6. Di bawah Key pair (login), pilih Create a new key pair. Masukkan **`MSKServerlessKeyPair`** untuk nama pasangan kunci. Kemudian pilih Download Key Pair. Alternatifnya, Anda dapat menggunakan pasangan kunci yang sudah ada.
7. Untuk pengaturan Jaringan, pilih Edit.
8. Di bawah VPC, masukkan ID virtual private cloud (VPC) untuk cluster tanpa server Anda. Ini adalah VPC berdasarkan layanan VPC Amazon yang IDnya Anda simpan setelah Anda membuat cluster.

9. Untuk Subnet, pilih subnet yang IDnya Anda simpan setelah Anda membuat cluster.
10. Untuk Firewall (grup keamanan), pilih grup keamanan yang terkait dengan cluster. Nilai ini berfungsi jika grup keamanan tersebut memiliki aturan masuk yang memungkinkan lalu lintas dari grup keamanan ke dirinya sendiri. Dengan aturan seperti itu, anggota kelompok keamanan yang sama dapat berkomunikasi satu sama lain. Untuk informasi selengkapnya, lihat Aturan grup keamanan di Panduan Pengembang Amazon VPC.
11. Perluas bagian Detail lanjutan dan pilih peran IAM yang Anda buat. Langkah 2: Buat peran IAM
12. Pilih Luncurkan.
13. Di panel navigasi kiri, pilih Instans. Kemudian pilih kotak centang di baris yang mewakili instans Amazon EC2 yang baru dibuat. Dari titik ini ke depan, kami menyebut contoh ini mesin klien.
14. Pilih Connect dan ikuti petunjuk untuk terhubung ke mesin klien.

Untuk mengatur alat klien Apache Kafka pada mesin klien

1. Untuk menginstal Java, jalankan perintah berikut pada mesin klien:

```
sudo yum -y install java-11
```

2. Untuk mendapatkan alat Apache Kafka yang kita butuhkan untuk membuat topik dan mengirim data, jalankan perintah berikut:

```
wget https://archive.apache.org/dist/kafka/2.8.1/kafka_2.12-2.8.1.tgz
```

```
tar -xzf kafka_2.12-2.8.1.tgz
```

3. Buka `kafka_2.12-2.8.1/libs` direktori, lalu jalankan perintah berikut untuk mengunduh file Amazon MSK IAM JAR. Amazon MSK IAM JAR memungkinkan mesin klien mengakses cluster.

```
wget https://github.com/aws/aws-msk-iam-auth/releases/download/v1.1.1/aws-msk-iam-
auth-1.1.1-all.jar
```

4. Pergi ke `kafka_2.12-2.8.1/bin` direktori. Salin pengaturan properti berikut dan tempel ke file baru. Beri nama file `client.properties` dan simpan.

```
security.protocol=SASL_SSL
sasl.mechanism=AWS_MSK_IAM
sasl.jaas.config=software.amazon.msk.auth.iam.IAMLoginModule required;
```

```
sasl.client.callback.handler.class=software.amazon.msk.auth.iam.IAMClientCallbackHandler
```

Langkah Selanjutnya

Langkah 4: Buat topik Apache Kafka

Langkah 4: Buat topik Apache Kafka

Pada langkah ini, Anda menggunakan mesin klien yang dibuat sebelumnya untuk membuat topik di klaster tanpa server.

Untuk membuat topik dan menulis data untuk itu

1. Dalam `export` perintah berikut, ganti *`my-endpoint`* dengan string bootstrap-server yang Anda simpan setelah Anda membuat cluster. Kemudian, pergi ke `kafka_2.12-2.8.1/bin` direktori pada mesin klien dan jalankan `export` perintah.

   ```
   export BS=my-endpoint
   ```

2. Jalankan perintah berikut untuk membuat topik yang disebut`msk-serverless-tutorial`.

   ```
   <path-to-your-kafka-installation>/bin/kafka-topics.sh --bootstrap-server $BS
    --command-config client.properties --create --topic msk-serverless-tutorial --
   partitions 6
   ```

Langkah Selanjutnya

Langkah 5: Menghasilkan dan Mengkonsumsi Data

Langkah 5: Menghasilkan dan Mengkonsumsi Data

Pada langkah ini, Anda menghasilkan dan mengkonsumsi data menggunakan topik yang Anda buat pada langkah sebelumnya.

Untuk menghasilkan dan mengkonsumsi pesan

1. Jalankan perintah berikut untuk membuat produser konsol.

```
<path-to-your-kafka-installation>/bin/kafka-console-producer.sh --broker-list $BS
 --producer.config client.properties --topic msk-serverless-tutorial
```

2. Masukkan pesan apa pun yang Anda inginkan, dan tekan Enter. Ulangi langkah ini dua atau tiga kali. Setiap kali Anda memasukkan baris dan tekan Enter, baris itu dikirim ke cluster Anda sebagai pesan terpisah.
3. Biarkan koneksi ke mesin klien tetap terbuka, dan kemudian buka koneksi kedua yang terpisah ke mesin itu di jendela baru.
4. Gunakan koneksi kedua Anda ke mesin klien untuk membuat konsumen konsol dengan perintah berikut. Ganti *my-endpoint* dengan string server bootstrap yang Anda simpan setelah Anda membuat cluster.

   ```
   <path-to-your-kafka-installation>/bin/kafka-console-consumer.sh --bootstrap-
   server my-endpoint --consumer.config client.properties --topic msk-serverless-
   tutorial --from-beginning
   ```

   Anda mulai melihat pesan yang Anda masukkan sebelumnya saat Anda menggunakan perintah produser konsol.
5. Masukkan lebih banyak pesan di jendela produser, dan saksikan mereka muncul di jendela konsumen.

Langkah Selanjutnya

Langkah 6: Hapus sumber daya

Langkah 6: Hapus sumber daya

Pada langkah ini, Anda menghapus sumber daya yang Anda buat dalam tutorial ini.

Untuk menghapus klaster

1. Buka konsol MSK Amazon di https://console.aws.amazon.com/msk/home.
2. Dalam daftar cluster, pilih cluster yang Anda buat untuk tutorial ini.
3. Untuk Tindakan, pilih Hapus klaster.
4. Masukkan `delete` di bidang, lalu pilih Hapus.

Untuk menghentikan mesin klien

1. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
2. Dalam daftar instans Amazon EC2, pilih mesin klien yang Anda buat untuk tutorial ini.
3. Pilih status Instance, lalu pilih Terminate instance.
4. Pilih Akhiri.

Untuk menghapus kebijakan IAM

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, pilih Peran.
3. Di kotak pencarian, masukkan nama peran IAM yang Anda buat untuk tutorial ini.
4. Pilih perannya. Kemudian pilih Hapus peran, dan konfirmasikan penghapusan.
5. Pada panel navigasi, pilih Kebijakan.
6. Di kotak pencarian, masukkan nama kebijakan yang Anda buat untuk tutorial ini.
7. Pilih kebijakan untuk membuka halaman ringkasannya. Pada halaman Ringkasan kebijakan, pilih Hapus kebijakan.
8. Pilih Hapus.

Konfigurasi untuk cluster tanpa server

Amazon MSK menetapkan properti konfigurasi broker untuk cluster tanpa server. Anda tidak dapat mengubah pengaturan properti konfigurasi broker ini. Namun, Anda dapat mengatur properti konfigurasi topik berikut.

Properti konfigurasi	Default	Dapat diedit	Nilai maksimum yang diizinkan
cleanup.policy	Hapus	Ya, tetapi hanya pada waktu pembuatan topik	
kompresi.type	Produser	Ya	
max.message.bytes	1048588	Ya	8 MiB

Properti konfigurasi	Default	Dapat diedit	Nilai maksimum yang diizinkan
message.timestamp.difference.max.ms	panjang.max	Ya	
message.timestamp.type	CreateTime	Ya	
retensi.bytes	250 GiB	Ya	250 GiB
retensi.ms	7 hari	Ya	Tidak terbatas.

Anda juga dapat menggunakan perintah Apache Kafka untuk mengatur atau memodifikasi properti konfigurasi tingkat topik untuk topik baru atau yang sudah ada. Untuk informasi selengkapnya tentang properti konfigurasi tingkat topik dan contoh cara mengaturnya, lihat Konfigurasi Tingkat Topik di dokumentasi resmi Apache Kafka.

Memantau cluster tanpa server

Amazon MSK terintegrasi dengan Amazon CloudWatch sehingga Anda dapat mengumpulkan, melihat, dan menganalisis metrik untuk kluster MSK Tanpa Server Anda. Metrik yang ditunjukkan pada tabel berikut tersedia untuk semua cluster tanpa server. Karena metrik ini diterbitkan sebagai titik data individual untuk setiap partisi dalam topik, kami sarankan untuk melihatnya sebagai statistik 'SUM' untuk mendapatkan tampilan tingkat topik.

Amazon MSK menerbitkan `PerSec` metrik dengan frekuensi CloudWatch sekali per menit. Ini berarti bahwa statistik 'SUM' untuk periode satu menit secara akurat mewakili data per detik untuk metrik. `PerSec` Untuk mengumpulkan data per detik untuk jangka waktu lebih dari satu menit, gunakan ekspresi CloudWatch matematika berikut:. `m1 * 60/PERIOD(m1)`

Metrik tersedia di tingkat pemantauan DEFAULT

Nama	Saat terlihat	Dimensi	Deskripsi
`BytesInPerSec`	Setelah produser	Nama Cluster, Topik	Jumlah byte per detik yang diterima dari klien. Metrik ini tersedia untuk setiap topik.

Nama	Saat terlihat	Dimensi	Deskripsi
	menulis ke suatu topik		
BytesOutPerSec	Setelah kelompok konsumen mengkonsumsi dari suatu topik	Nama Cluster, Topik	Jumlah byte per detik dikirim ke klien. Metrik ini tersedia untuk setiap topik.
FetchMessageConversionsPerSec	Setelah kelompok konsumen mengkonsumsi dari suatu topik	Nama Cluster, Topik	Jumlah konversi pesan pengambilan per detik untuk topik tersebut.
EstimatedMaxTimeLag	Setelah kelompok konsumen mengkonsumsi dari suatu topik	Nama Cluster, Grup Konsumen, Topik	Estimasi waktu MaxOffsetLag metrik.
MaxOffsetLag	Setelah kelompok konsumen mengkonsumsi dari suatu topik	Nama Cluster, Grup Konsumen, Topik	Keterlambatan offset maksimum di semua partisi dalam suatu topik.
MessagesInPerSec	Setelah produser menulis ke suatu topik	Nama Cluster, Topik	Jumlah pesan masuk per detik untuk topik tersebut.
ProduceMessageConversionsPerSec	Setelah produser menulis ke suatu topik	Nama Cluster, Topik	Jumlah konversi pesan hasil per detik untuk topik tersebut.

Nama	Saat terlihat	Dimensi	Deskripsi
`SumOffsetLag`	Setelah kelompok konsumen mengkonsumsi dari suatu topik	Nama Cluster, Grup Konsumen, Topik	Kelambatan offset agregat untuk semua partisi dalam suatu topik.

Untuk melihat metrik MSK Tanpa Server

1. Masuk ke AWS Management Console dan buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
2. Di panel navigasi, di bawah Metrik, pilih Semua metrik.
3. Dalam metrik, cari istilah **`kafka`** tersebut.
4. Pilih AWS/KAFKA/Nama Cluster, Topik atau AWS/KAFKA/Nama Cluster, Grup Konsumen, Topik untuk melihat metrik yang berbeda.

MSK Connect

Apa itu MSK Connect?

MSK Connect adalah fitur Amazon MSK yang memudahkan pengembang untuk melakukan streaming data ke dan dari cluster Apache Kafka mereka. MSK Connect menggunakan Kafka Connect 2.7.1, kerangka open-source untuk menghubungkan cluster Apache Kafka dengan sistem eksternal seperti database, indeks pencarian, dan sistem file. Dengan MSK Connect, Anda dapat menggunakan konektor terkelola penuh yang dibuat untuk Kafka Connect yang memindahkan data ke atau menarik data dari penyimpanan data populer seperti Amazon S3 dan Amazon Service. OpenSearch Anda dapat menerapkan konektor yang dikembangkan oleh pihak ke-3 seperti Debezium untuk streaming log perubahan dari database ke cluster Apache Kafka, atau menyebarkan konektor yang ada tanpa perubahan kode. Konektor secara otomatis menskalakan untuk menyesuaikan perubahan beban dan Anda hanya membayar untuk sumber daya yang Anda gunakan.

Gunakan konektor sumber untuk mengimpor data dari sistem eksternal ke topik Anda. Dengan konektor wastafel, Anda dapat mengekspor data dari topik Anda ke sistem eksternal.

MSK Connect mendukung konektor untuk cluster Apache Kafka apa pun dengan konektivitas ke VPC Amazon, apakah itu cluster MSK atau cluster Apache Kafka yang dihosting secara independen.

MSK Connect terus memantau kesehatan konektor dan status pengiriman, menambal dan mengelola perangkat keras yang mendasarinya, dan menskalakan konektor secara otomatis agar sesuai dengan perubahan throughput.

Untuk mulai menggunakan MSK Connect, lihatthe section called “Memulai”.

Untuk mempelajari tentang AWS sumber daya yang dapat Anda buat dengan MSK Connect, lihatthe section called “Konektor”,the section called “Plugin”, danthe section called “Pekerja”.

Untuk informasi tentang MSK Connect API, lihat Referensi API Amazon MSK Connect.

Memulai menggunakan MSK Connect

Ini adalah step-by-step tutorial yang menggunakan AWS Management Console untuk membuat cluster MSK dan konektor sink yang mengirimkan data dari cluster ke bucket S3.

Topik

Langkah 1: Siapkan sumber daya yang dibutuhkan

Pada langkah ini Anda membuat sumber daya berikut yang Anda butuhkan untuk skenario memulai ini:

- Bucket S3 berfungsi sebagai tujuan yang menerima data dari konektor.
- Cluster MSK tempat Anda akan mengirim data. Konektor kemudian akan membaca data dari cluster ini dan mengirimkannya ke bucket S3 tujuan.
- Peran IAM yang memungkinkan konektor untuk menulis ke bucket S3 tujuan.
- Titik akhir VPC Amazon untuk memungkinkan pengiriman data dari VPC Amazon yang memiliki cluster dan konektor ke Amazon S3.

Untuk membuat bucket S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pilih Buat bucket.
3. Untuk nama bucket, masukkan nama deskriptif seperti `mkc-tutorial-destination-bucket`.
4. Gulir ke bawah dan pilih Buat ember.
5. Dalam daftar ember, pilih bucket yang baru dibuat.
6. Pilih Buat folder.
7. Masukkan `tutorial` nama folder, lalu gulir ke bawah dan pilih Buat folder.

Untuk membuat cluster

1. Buka konsol Amazon MSK di https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.

2. Di panel kiri, di bawah MSK Clusters, pilih Cluster.
3. Pilih Buat klaster.
4. Pilih Custom create.
5. Untuk nama cluster masukkan`mkc-tutorial-cluster`.
6. Di bawah Properti klaster umum, pilih Disediakan untuk jenis cluster.
7. Di bawah Networking, pilih VPC Amazon. Kemudian pilih Availability Zones dan subnet yang ingin Anda gunakan. Ingat ID VPC Amazon dan subnet yang Anda pilih karena Anda membutuhkannya nanti dalam tutorial ini.
8. Di bawah Metode kontrol akses memastikan bahwa hanya akses Tidak Diautentikasi yang dipilih.
9. Di bawah Enkripsi pastikan bahwa hanya Plaintext yang dipilih.
10. Lanjutkan melalui wizard dan kemudian pilih Buat cluster. Ini membawa Anda ke halaman detail untuk cluster. Pada halaman itu, di bawah Grup keamanan diterapkan, temukan ID grup keamanan. Ingat ID itu karena Anda membutuhkannya nanti dalam tutorial ini.

Untuk membuat peran IAM yang dapat menulis ke bucket tujuan

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel kiri, di bawah Manajemen akses, pilih Peran.
3. Pilih Buat peran.
4. Di bawah Atau pilih layanan untuk melihat kasus penggunaannya, pilih S3.
5. Gulir ke bawah dan di bawah Pilih kasus penggunaan Anda, sekali lagi pilih S3.
6. Pilih Berikutnya: Izin.
7. Pilih Buat kebijakan. Ini membuka tab baru di browser Anda tempat Anda akan membuat kebijakan. Biarkan tab role-creation asli terbuka karena kita akan kembali ke sana nanti.
8. Pilih tab JSON, lalu ganti teks di jendela dengan kebijakan berikut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets"
      ],
```

```
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::<my-tutorial-destination-bucket>"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:AbortMultipartUpload",
        "s3:ListMultipartUploadParts",
        "s3:ListBucketMultipartUploads"
      ],
      "Resource": "*"
    }
  ]
}
```

9. Pilih Berikutnya: Tanda.
10. Pilih Berikutnya: Tinjau.
11. Masukkan `mkc-tutorial-policy` nama kebijakan, lalu gulir ke bawah dan pilih Buat kebijakan.
12. Kembali ke tab browser tempat Anda membuat peran, pilih tombol segarkan.
13. Temukan `mkc-tutorial-policy` dan pilih dengan memilih tombol di sebelah kirinya.
14. Pilih Berikutnya: Tanda.
15. Pilih Berikutnya: Tinjau.
16. Masukkan `mkc-tutorial-role` nama peran, dan hapus teks di kotak deskripsi.
17. Pilih Buat peran.

Untuk memungkinkan MSK Connect untuk mengambil peran

1. Di konsol IAM, di panel kiri, di bawah Manajemen akses, pilih Peran.

2. Temukan `mkc-tutorial-role` dan pilih.
3. Di bawah Ringkasan peran, pilih tab Trust relationship.
4. Pilih Edit trust relationship (Edit Hubungan Kepercayaan).
5. Ganti kebijakan kepercayaan yang ada dengan JSON berikut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "kafkaconnect.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

6. Pilih Perbarui Kebijakan Kepercayaan.

Untuk membuat titik akhir VPC Amazon dari VPC cluster ke Amazon S3

1. Buka konsol Amazon VPC di https://console.aws.amazon.com/vpc/.
2. Di panel kiri, pilih Endpoints.
3. Pilih Buat Titik Akhir.
4. Di bawah Nama Layanan pilih layanan com.amazonaws.us-east-1.s3 dan tipe Gateway.
5. Pilih VPC cluster dan kemudian pilih kotak di sebelah kiri tabel rute yang terkait dengan subnet cluster.
6. Pilih Buat Titik Akhir.

Langkah Selanjutnya

Langkah 2: Buat plugin kustom

Langkah 2: Buat plugin kustom

Plugin berisi kode yang mendefinisikan logika konektor. Pada langkah ini Anda membuat plugin khusus yang memiliki kode untuk Lensa Konektor Sink Amazon S3. Pada langkah selanjutnya, saat

Anda membuat konektor MSK, Anda menentukan bahwa kodenya ada di plugin khusus ini. Anda dapat menggunakan plugin yang sama untuk membuat beberapa konektor MSK dengan konfigurasi yang berbeda.

Untuk membuat plugin kustom

1. Unduh konektor S3.
2. Unggah file ZIP ke bucket S3 yang dapat Anda akses. Untuk informasi tentang cara mengunggah file ke Amazon S3, lihat Mengunggah objek di panduan pengguna Amazon S3.
3. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
4. Di panel kiri perluas MSK Connect, lalu pilih Plugin kustom.
5. Pilih Buat plugin kustom.
6. Pilih Jelajahi S3.
7. Dalam daftar bucket, temukan bucket tempat Anda mengunggah file ZIP, dan pilih bucket itu.
8. Dalam daftar objek di ember, pilih tombol radio di sebelah kiri file ZIP, lalu pilih tombol berlabel Pilih.
9. Masukkan `mkc-tutorial-plugin` untuk nama plugin kustom, lalu pilih Buat plugin kustom.

Mungkin perlu AWS beberapa menit untuk menyelesaikan pembuatan plugin khusus. Ketika proses pembuatan selesai, Anda melihat pesan berikut di spanduk di bagian atas jendela browser.

```
Custom plugin mkc-tutorial-plugin was successfully created
The custom plugin was created. You can now create a connector using this custom plugin.
```

Langkah Selanjutnya

Langkah 3: Buat mesin klien dan topik Apache Kafka

Langkah 3: Buat mesin klien dan topik Apache Kafka

Pada langkah ini Anda membuat instans Amazon EC2 untuk digunakan sebagai instans klien Apache Kafka. Anda kemudian menggunakan instance ini untuk membuat topik di cluster.

Untuk membuat mesin klien

1. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.

2. Pilih Luncurkan Instans.
3. Masukkan Nama untuk mesin klien Anda, seperti**`mkc-tutorial-client`**.
4. Tinggalkan Amazon Linux 2 AMI (HVM) - Kernel 5.10, Jenis Volume SSD dipilih untuk jenis Amazon Machine Image (AMI).
5. Pilih jenis instans t2.xlarge.
6. Di bawah Key pair (login), pilih Create a new key pair. Masukkan **`mkc-tutorial-key-pair`** nama pasangan Kunci, lalu pilih Unduh Pasangan Kunci. Alternatifnya, Anda dapat menggunakan pasangan kunci yang sudah ada.
7. Pilih Luncurkan instans.
8. Pilih Lihat Instans. Kemudian, di kolom Grup Keamanan, pilih grup keamanan yang terkait dengan instans baru Anda. Salin ID grup keamanan, dan simpan untuk nanti.

Untuk memungkinkan klien yang baru dibuat mengirim data ke cluster

1. Buka konsol Amazon VPC di https://console.aws.amazon.com/vpc/.
2. Di panel kiri, di bawah KEAMANAN, pilih Grup Keamanan. Di kolom ID grup Keamanan, cari grup keamanan klaster. Anda menyimpan ID grup keamanan ini saat Anda membuat klaster dithe section called “Langkah 1: Siapkan sumber daya yang dibutuhkan”. Pilih grup keamanan ini dengan memilih kotak di sebelah kiri barisnya. Pastikan tidak ada grup keamanan lain yang dipilih secara bersamaan.
3. Di bagian bawah layar, pilih tab Aturan masuk.
4. Pilih Edit aturan masuk.
5. Di kiri bawah layar, pilih Tambahkan aturan.
6. Dalam aturan baru, pilih Semua lalu lintas di kolom Jenis. Di bidang di sebelah kanan kolom Sumber, masukkan ID grup keamanan mesin klien. Ini adalah ID grup keamanan yang Anda simpan setelah Anda membuat mesin klien.
7. Pilih Simpan aturan. Cluster MSK Anda sekarang akan menerima semua lalu lintas dari klien yang Anda buat dalam prosedur sebelumnya.

Cara membuat topik

1. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
2. Dalam tabel contoh pilih`mkc-tutorial-client`.
3. Di dekat bagian atas layar, pilih Connect, lalu ikuti petunjuk untuk terhubung ke instance.

4. Instal Java pada instance klien dengan menjalankan perintah berikut:

```
sudo yum install java-1.8.0
```

5. Jalankan perintah berikut untuk mengunduh Apache Kafka.

```
wget https://archive.apache.org/dist/kafka/2.2.1/kafka_2.12-2.2.1.tgz
```

> **Note**
>
> Jika Anda ingin menggunakan situs mirror selain yang digunakan dalam perintah ini, Anda dapat memilih yang berbeda di situs web Apache.

6. Jalankan perintah berikut di direktori tempat Anda mengunduh file TAR pada langkah sebelumnya.

```
tar -xzf kafka_2.12-2.2.1.tgz
```

7. Buka direktori kafka_2.12-2.2.1.
8. Buka konsol Amazon MSK di https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
9. Di panel kiri pilih Cluster, lalu pilih namanya. `mkc-tutorial-cluster`
10. Pilih Lihat informasi klien.
11. Salin string koneksi Plaintext.
12. Pilih Selesai.
13. Jalankan perintah berikut pada instance klien (`mkc-tutorial-client`), ganti *`bootstrapServerString`*dengan nilai yang Anda simpan saat melihat informasi klien klaster.

```
<path-to-your-kafka-installation>/bin/kafka-topics.sh --create --bootstrap-
server bootstrapServerString --replication-factor 2 --partitions 1 --topic mkc-
tutorial-topic
```

Jika perintah berhasil, Anda melihat pesan berikut: `Created topic mkc-tutorial-topic.`

Langkah Selanjutnya

Langkah 4: Buat konektor

Langkah 4: Buat konektor

Untuk membuat konektor

1. Masuk ke AWS Management Console, dan buka konsol MSK Amazon di https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Di panel kiri, perluas MSK Connect, lalu pilih Connectors.
3. Pilih Buat konektor.
4. Dalam daftar plugin, pilih`mkc-tutorial-plugin`, lalu pilih Berikutnya.
5. Untuk nama konektor masukkan`mkc-tutorial-connector`.
6. Dalam daftar cluster, pilih`mkc-tutorial-cluster`.
7. Salin konfigurasi berikut dan tempel ke bidang konfigurasi konektor.

```
connector.class=io.confluent.connect.s3.S3SinkConnector
s3.region=us-east-1
format.class=io.confluent.connect.s3.format.json.JsonFormat
flush.size=1
schema.compatibility=NONE
tasks.max=2
topics=mkc-tutorial-topic
partitioner.class=io.confluent.connect.storage.partitioner.DefaultPartitioner
storage.class=io.confluent.connect.s3.storage.S3Storage
s3.bucket.name=<my-tutorial-destination-bucket>
topics.dir=tutorial
```

8. Di bawah Izin akses pilih`mkc-tutorial-role`.
9. Pilih Selanjutnya. Pada halaman Keamanan, pilih Berikutnya lagi.
10. Pada halaman Log pilih Berikutnya.
11. Di bawah Tinjau dan buat pilih Buat konektor.

Langkah Selanjutnya

Langkah 5: Kirim data

Langkah 5: Kirim data

Pada langkah ini Anda mengirim data ke topik Apache Kafka yang Anda buat sebelumnya, dan kemudian mencari data yang sama di bucket S3 tujuan.

Untuk mengirim data ke cluster MSK

1. Di `bin` folder instalasi Apache Kafka pada instance klien, buat file teks bernama `client.properties` dengan konten berikut.

```
security.protocol=PLAINTEXT
```

2. Jalankan perintah berikut untuk membuat produser konsol. Ganti *`BootstrapBrokerString`*dengan nilai yang Anda peroleh saat menjalankan perintah sebelumnya.

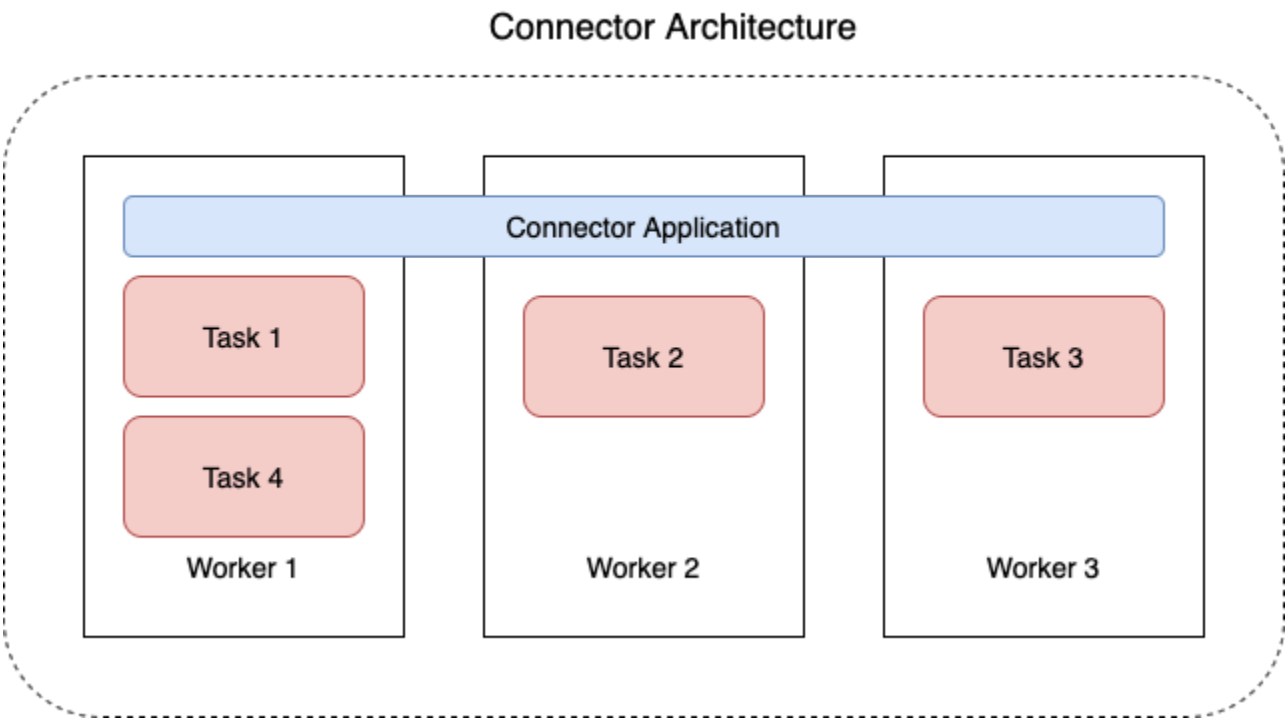
```
<path-to-your-kafka-installation>/bin/kafka-console-producer.sh --broker-
list BootstrapBrokerString --producer.config client.properties --topic mkc-
tutorial-topic
```

3. Masukkan pesan apa pun yang Anda inginkan, dan tekan Enter. Ulangi langkah ini dua atau tiga kali. Setiap kali Anda memasukkan baris dan tekan Enter, baris itu dikirim ke cluster Apache Kafka Anda sebagai pesan terpisah.
4. Lihat di bucket Amazon S3 tujuan untuk menemukan pesan yang Anda kirim pada langkah sebelumnya.

Konektor

Konektor mengintegrasikan sistem eksternal dan layanan Amazon dengan Apache Kafka dengan terus menyalin data streaming dari sumber data ke cluster Apache Kafka Anda, atau terus menyalin data dari cluster Anda ke dalam data sink. Konektor juga dapat melakukan logika ringan seperti transformasi, konversi format, atau memfilter data sebelum mengirimkan data ke tujuan. Konektor sumber menarik data dari sumber data dan mendorong data ini ke dalam cluster, sementara konektor sink menarik data dari cluster dan mendorong data ini ke dalam sink data.

Diagram berikut menunjukkan arsitektur konektor. Seorang pekerja adalah proses mesin virtual Java (JVM) yang menjalankan logika konektor. Setiap pekerja membuat serangkaian tugas yang berjalan di thread paralel dan melakukan pekerjaan menyalin data. Tugas tidak menyimpan status, dan karenanya dapat dimulai, dihentikan, atau dimulai ulang kapan saja untuk menyediakan pipeline data yang tangguh dan dapat diskalakan.



Kapasitas konektor

Kapasitas total konektor tergantung pada jumlah pekerja yang dimiliki konektor, serta pada jumlah MSK Connect Units (MCU) per pekerja. Setiap MCU mewakili 1 vCPU komputasi dan 4 GiB memori. Memori MCU berkaitan dengan memori total instance pekerja dan bukan memori heap yang digunakan.

Pekerja MSK Connect menggunakan alamat IP di subnet yang disediakan pelanggan. Setiap pekerja menggunakan satu alamat IP dari salah satu subnet yang disediakan pelanggan. Anda harus memastikan bahwa Anda memiliki cukup alamat IP yang tersedia di subnet yang disediakan untuk CreateConnector permintaan untuk memperhitungkan kapasitas yang ditentukan, terutama ketika konektor penskalaan otomatis di mana jumlah pekerja dapat berfluktuasi.

Untuk membuat konektor, Anda harus memilih di antara salah satu dari dua mode kapasitas berikut.

- Disediakan - Pilih mode ini jika Anda mengetahui persyaratan kapasitas untuk konektor Anda. Anda menentukan dua nilai:
  - Jumlah pekerja.
  - Jumlah MCU per pekerja.

- Skala otomatis - Pilih mode ini jika persyaratan kapasitas untuk konektor Anda bervariasi atau jika Anda tidak mengetahuinya sebelumnya. Saat Anda menggunakan mode skala otomatis, Amazon MSK Connect akan mengganti `tasks.max` properti konektor Anda dengan nilai yang sebanding dengan jumlah pekerja yang berjalan di konektor dan jumlah MCU per pekerja.

  Anda menentukan tiga set nilai:

  - Jumlah pekerja minimum dan maksimum.
  - Persentase scale-in dan scale-out untuk pemanfaatan CPU, yang ditentukan oleh metrik. `CpuUtilization` Ketika `CpuUtilization` metrik untuk konektor melebihi persentase scale-out, MSK Connect meningkatkan jumlah pekerja yang berjalan di konektor. Ketika `CpuUtilization` metrik berada di bawah persentase scale-in, MSK Connect mengurangi jumlah pekerja. Jumlah pekerja selalu tetap dalam angka minimum dan maksimum yang Anda tentukan saat Anda membuat konektor.
  - Jumlah MCU per pekerja.

Untuk informasi lebih lanjut tentang pekerja, lihatthe section called “Pekerja”. Untuk mempelajari metrik MSK Connect, lihat. the section called “Pemantauan”

Membuat konektor

Membuat konektor menggunakan AWS Management Console

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Di panel kiri, di bawah MSK Connect, pilih Konektor.
3. Pilih Buat konektor.
4. Anda dapat memilih antara menggunakan plugin khusus yang ada untuk membuat konektor, atau membuat plugin khusus baru terlebih dahulu. Untuk informasi tentang plugin khusus dan cara membuatnya, lihatthe section called “Plugin”. Dalam prosedur ini, mari kita asumsikan Anda memiliki plugin khusus yang ingin Anda gunakan. Dalam daftar plugin khusus, temukan salah satu yang ingin Anda gunakan, dan pilih kotak di sebelah kirinya, lalu pilih Berikutnya.
5. Masukkan nama dan, secara opsional, deskripsi.
6. Pilih cluster yang ingin Anda sambungkan.
7. Tentukan konfigurasi konektor. Parameter konfigurasi yang perlu Anda tentukan bergantung pada jenis konektor yang ingin Anda buat. Namun, beberapa parameter umum untuk semua

konektor, misalnya, `connector.class` dan `tasks.max` parameter. Berikut ini adalah contoh konfigurasi untuk Confluent Amazon S3 Sink Connector.

```
connector.class=io.confluent.connect.s3.S3SinkConnector
tasks.max=2
topics=my-example-topic
s3.region=us-east-1
s3.bucket.name=my-destination-bucket
flush.size=1
storage.class=io.confluent.connect.s3.storage.S3Storage
format.class=io.confluent.connect.s3.format.json.JsonFormat
partitioner.class=io.confluent.connect.storage.partitioner.DefaultPartitioner
key.converter=org.apache.kafka.connect.storage.StringConverter
value.converter=org.apache.kafka.connect.storage.StringConverter
schema.compatibility=NONE
```

8. Selanjutnya, Anda mengonfigurasi kapasitas konektor Anda. Anda dapat memilih di antara dua mode kapasitas: provisioned dan auto scaled. Untuk informasi tentang dua opsi ini, lihatthe section called “Kapasitas”.
9. Pilih konfigurasi pekerja default atau konfigurasi pekerja khusus. Untuk informasi tentang membuat konfigurasi pekerja kustom, lihatthe section called “Pekerja”.
10. Selanjutnya, Anda menentukan peran eksekusi layanan. Ini harus menjadi peran IAM yang dapat diasumsikan MSK Connect, dan yang memberikan konektor semua izin yang diperlukan untuk mengakses sumber daya yang diperlukan. AWS Izin tersebut tergantung pada logika konektor. Untuk informasi tentang cara membuat peran ini, lihatthe section called “Peran eksekusi layanan”.
11. Pilih Berikutnya, tinjau informasi keamanan, lalu pilih Berikutnya lagi.
12. Tentukan opsi logging yang Anda inginkan, lalu pilih Berikutnya. Untuk informasi tentang pencatatan, lihatthe section called “Pencatatan log”.
13. Pilih Buat konektor.

Untuk menggunakan MSK Connect API untuk membuat konektor, lihat CreateConnector.

Plugin

Plugin adalah AWS sumber daya yang berisi kode yang mendefinisikan logika konektor Anda. Anda mengunggah file JAR (atau file ZIP yang berisi satu atau beberapa file JAR) ke bucket S3, dan tentukan lokasi bucket saat Anda membuat plugin. Saat Anda membuat konektor, Anda menentukan

plugin yang ingin digunakan MSK Connect untuk itu. Hubungan plugin dengan konektor adalah one-to-many: Anda dapat membuat satu atau lebih konektor dari plugin yang sama.

Untuk informasi tentang cara mengembangkan kode untuk konektor, lihat Panduan Pengembangan Konektor di dokumentasi Apache Kafka.

Membuat plugin khusus menggunakan AWS Management Console

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Di panel kiri, di bawah MSK Connect, pilih Plugin kustom.
3. Pilih Buat plugin kustom.
4. Pilih Jelajahi S3.
5. Dalam daftar bucket S3, pilih bucket yang memiliki file JAR atau ZIP untuk plugin.
6. Dalam daftar objek, pilih kotak di sebelah kiri file JAR atau ZIP untuk plugin, lalu pilih Pilih.
7. Pilih Buat plugin kustom.

Untuk menggunakan MSK Connect API untuk membuat plugin khusus, lihat CreateCustomPlugin.

Pekerja

Seorang pekerja adalah proses mesin virtual Java (JVM) yang menjalankan logika konektor. Setiap pekerja membuat serangkaian tugas yang berjalan di thread paralel dan melakukan pekerjaan menyalin data. Tugas tidak menyimpan status, dan karenanya dapat dimulai, dihentikan, atau dimulai ulang kapan saja untuk menyediakan pipeline data yang tangguh dan dapat diskalakan. Perubahan jumlah pekerja, baik karena peristiwa penskalaan atau karena kegagalan yang tidak terduga, secara otomatis terdeteksi oleh pekerja yang tersisa. Mereka berkoordinasi untuk menyeimbangkan kembali tugas di seluruh set pekerja yang tersisa. Connect worker menggunakan kelompok konsumen Apache Kafka untuk berkoordinasi dan menyeimbangkan kembali.

Jika persyaratan kapasitas konektor Anda bervariasi atau sulit diperkirakan, Anda dapat membiarkan MSK Connect menskalakan jumlah pekerja sesuai kebutuhan antara batas bawah dan batas atas yang Anda tentukan. Atau, Anda dapat menentukan jumlah pasti pekerja yang ingin Anda jalankan logika konektor Anda. Untuk informasi selengkapnya, lihat the section called “Kapasitas”.

Pekerja MSK Connect menggunakan alamat IP

Pekerja MSK Connect menggunakan alamat IP di subnet yang disediakan pelanggan. Setiap pekerja menggunakan satu alamat IP dari salah satu subnet yang disediakan pelanggan. Anda harus

memastikan bahwa Anda memiliki cukup alamat IP yang tersedia di subnet yang disediakan untuk CreateConnector permintaan untuk memperhitungkan kapasitas yang ditentukan, terutama ketika konektor penskalaan otomatis di mana jumlah pekerja dapat berfluktuasi.

Topik

Konfigurasi pekerja default

MSK Connect menyediakan konfigurasi pekerja default berikut:

```
key.converter=org.apache.kafka.connect.storage.StringConverter
value.converter=org.apache.kafka.connect.storage.StringConverter
```

Properti konfigurasi pekerja yang didukung

MSK Connect menyediakan konfigurasi pekerja default. Anda juga memiliki opsi untuk membuat konfigurasi pekerja khusus untuk digunakan dengan konektor Anda. Daftar berikut mencakup informasi tentang properti konfigurasi pekerja yang didukung atau tidak didukung oleh Amazon MSK Connect.

- `value.converter`Properti `key.converter` dan diperlukan.
- MSK Connect mendukung properti `producer.` konfigurasi berikut.

```
producer.acks
producer.batch.size
producer.buffer.memory
producer.compression.type
producer.enable.idempotence
producer.key.serializer
producer.max.request.size
producer.metadata.max.age.ms
producer.metadata.max.idle.ms
producer.partitioner.class
producer.reconnect.backoff.max.ms
```

```
producer.reconnect.backoff.ms
producer.request.timeout.ms
producer.retry.backoff.ms
producer.value.serializer
```

- MSK Connect mendukung properti `consumer.` konfigurasi berikut.

```
consumer.allow.auto.create.topics
consumer.auto.offset.reset
consumer.check.crcs
consumer.fetch.max.bytes
consumer.fetch.max.wait.ms
consumer.fetch.min.bytes
consumer.heartbeat.interval.ms
consumer.key.deserializer
consumer.max.partition.fetch.bytes
consumer.max.poll.records
consumer.metadata.max.age.ms
consumer.partition.assignment.strategy
consumer.reconnect.backoff.max.ms
consumer.reconnect.backoff.ms
consumer.request.timeout.ms
consumer.retry.backoff.ms
consumer.session.timeout.ms
consumer.value.deserializer
```

- Semua properti konfigurasi lain yang tidak dimulai dengan `consumer.` awalan `producer.` atau didukung kecuali untuk properti berikut.

```
access.control.
admin.
admin.listeners.https.
client.
connect.
inter.worker.
internal.
listeners.https.
metrics.
metrics.context.
rest.
sasl.
security.
socket.
```

```
ssl.
topic.tracking.
worker.
bootstrap.servers
config.storage.topic
connections.max.idle.ms
connector.client.config.override.policy
group.id
listeners
metric.reporters
plugin.path
receive.buffer.bytes
response.http.headers.config
scheduled.rebalance.max.delay.ms
send.buffer.bytes
status.storage.topic
```

Untuk informasi selengkapnya tentang properti konfigurasi pekerja dan apa yang diwakilinya, lihat Konfigurasi Kafka Connect di dokumentasi Apache Kafka.

Membuat konfigurasi pekerja khusus

Membuat konfigurasi pekerja khusus menggunakan AWS Management Console

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Di panel kiri, di bawah MSK Connect, pilih konfigurasi Worker.
3. Pilih Buat konfigurasi pekerja.
4. Masukkan nama dan deskripsi opsional, lalu tambahkan properti dan nilai yang ingin Anda atur.
5. Pilih Buat konfigurasi pekerja.

Untuk menggunakan MSK Connect API untuk membuat konfigurasi pekerja, lihat CreateWorkerConfiguration.

Mengelola offset konektor sumber menggunakan `offset.storage.topic`

Bagian ini memberikan informasi untuk membantu Anda mengelola offset konektor sumber menggunakan topik penyimpanan offset. Topik penyimpanan offset adalah topik internal yang digunakan Kafka Connect untuk menyimpan konektor dan offset konfigurasi tugas.

Menggunakan topik penyimpanan offset default

Secara default, Amazon MSK Connect menghasilkan topik penyimpanan offset baru di cluster Kafka Anda untuk setiap konektor yang Anda buat. MSK membangun nama topik default menggunakan bagian dari konektor ARN. Misalnya, `__amazon_msk_connect_offsets_my-mskc-connector_12345678-09e7-4abc-8be8-c657f7e4ff32-2`.

Menentukan topik penyimpanan offset Anda sendiri

Untuk memberikan kontinuitas offset antara konektor sumber, Anda dapat menggunakan topik penyimpanan offset pilihan Anda alih-alih topik default. Menentukan topik penyimpanan offset membantu Anda menyelesaikan tugas seperti membuat konektor sumber yang melanjutkan pembacaan dari offset terakhir konektor sebelumnya.

Untuk menentukan topik penyimpanan offset, Anda memberikan nilai untuk `offset.storage.topic` properti dalam konfigurasi pekerja sebelum membuat konektor. Jika Anda ingin menggunakan kembali topik penyimpanan offset untuk menggunakan offset dari konektor yang dibuat sebelumnya, Anda harus memberi konektor baru nama yang sama dengan konektor lama. Jika Anda membuat topik penyimpanan offset kustom, Anda harus menyetel `cleanup.policy`ke `compact` dalam konfigurasi topik Anda.

> **Note**
>
> Jika Anda menentukan topik penyimpanan offset saat membuat konektor sink, MSK Connect akan membuat topik jika belum ada. Namun, topik tersebut tidak akan digunakan untuk menyimpan offset konektor.
>
> Offset konektor sink malah dikelola menggunakan protokol grup konsumen Kafka. Setiap konektor wastafel membuat grup bernama`connect-{CONNECTOR_NAME}`. Selama grup konsumen ada, konektor wastafel berturut-turut yang Anda buat dengan `CONNECTOR_NAME` nilai yang sama akan berlanjut dari offset komitmen terakhir.

Example : Menentukan topik penyimpanan offset untuk membuat ulang konektor sumber dengan konfigurasi yang diperbarui

Misalkan Anda memiliki konektor change data capture (CDC) dan Anda ingin memodifikasi konfigurasi konektor tanpa kehilangan tempat Anda di aliran CDC. Anda tidak dapat memperbarui konfigurasi konektor yang ada, tetapi Anda dapat menghapus konektor dan membuat yang baru dengan nama yang sama. Untuk memberi tahu konektor baru di mana harus mulai membaca di aliran CDC, Anda dapat menentukan topik penyimpanan offset konektor lama dalam konfigurasi pekerja Anda. Langkah-langkah berikut menunjukkan bagaimana menyelesaikan tugas ini.

1. Pada mesin klien Anda, jalankan perintah berikut untuk menemukan nama topik penyimpanan offset konektor Anda. Ganti *<bootstrapBrokerString>* dengan string broker bootstrap cluster Anda. Untuk petunjuk tentang mendapatkan string broker bootstrap Anda, lihatMendapatkan broker bootstrap untuk cluster MSK Amazon.

   ```
   <path-to-your-kafka-installation>/bin/kafka-topics.sh --list --bootstrap-
   server <bootstrapBrokerString>
   ```

   Output berikut menunjukkan daftar semua topik cluster, termasuk topik konektor internal default. Dalam contoh ini, konektor CDC yang ada menggunakan topik penyimpanan offset default yang dibuat oleh MSK Connect. Inilah sebabnya mengapa topik penyimpanan offset disebut`__amazon_msk_connect_offsets_my-mskc-connector_12345678-09e7-4abc-8be8-c657f7e4ff32-2`.

   ```
   __consumer_offsets
   __amazon_msk_canary
   __amazon_msk_connect_configs_my-mskc-connector_12345678-09e7-4abc-8be8-
   c657f7e4ff32-2
   __amazon_msk_connect_offsets_my-mskc-connector_12345678-09e7-4abc-8be8-
   c657f7e4ff32-2
   __amazon_msk_connect_status_my-mskc-connector_12345678-09e7-4abc-8be8-
   c657f7e4ff32-2
   my-msk-topic-1
   my-msk-topic-2
   ```

2. Buka konsol MSK Amazon di https://console.aws.amazon.com/msk/.
3. Pilih konektor Anda dari daftar Konektor. Salin dan simpan konten bidang konfigurasi Konektor sehingga Anda dapat memodifikasinya dan menggunakannya untuk membuat konektor baru.

4. Pilih Hapus untuk menghapus konektor. Kemudian masukkan nama konektor di bidang input teks untuk mengonfirmasi penghapusan.
5. Buat konfigurasi pekerja khusus dengan nilai yang sesuai dengan skenario Anda. Untuk petunjuk, lihat Membuat konfigurasi pekerja khusus.

   Dalam konfigurasi pekerja Anda, Anda harus menentukan nama topik penyimpanan offset yang sebelumnya Anda ambil sebagai nilai untuk `offset.storage.topic` like dalam konfigurasi berikut.

   ```
   config.providers.secretManager.param.aws.region=us-east-1
   key.converter=<org.apache.kafka.connect.storage.StringConverter>
   value.converter=<org.apache.kafka.connect.storage.StringConverter>
   config.providers.secretManager.class=com.github.jcustenborder.kafka.config.aws.SecretsManag
   config.providers=secretManager
   offset.storage.topic=__amazon_msk_connect_offsets_my-mskc-
   connector_12345678-09e7-4abc-8be8-c657f7e4ff32-2
   ```

6. **Important**

   Anda harus memberikan konektor baru Anda nama yang sama dengan konektor lama.

   Buat konektor baru menggunakan konfigurasi pekerja yang Anda atur di langkah sebelumnya. Untuk petunjuk, lihat Membuat konektor.

Pertimbangan

Pertimbangkan hal berikut ketika Anda mengelola offset konektor sumber.

- Untuk menentukan topik penyimpanan offset, berikan nama topik Kafka tempat offset konektor disimpan sebagai nilai untuk konfigurasi pekerja `offset.storage.topic` Anda.
- Berhati-hatilah saat Anda membuat perubahan pada konfigurasi konektor. Mengubah nilai konfigurasi dapat mengakibatkan perilaku konektor yang tidak diinginkan jika konektor sumber menggunakan nilai dari konfigurasi ke catatan offset kunci. Kami menyarankan Anda merujuk ke dokumentasi plugin Anda untuk panduan.
- Sesuaikan jumlah partisi default — Selain menyesuaikan konfigurasi pekerja dengan menambahkan`offset.storage.topic`, Anda dapat menyesuaikan jumlah partisi untuk topik penyimpanan offset dan status. Partisi default untuk topik internal adalah sebagai berikut.

  - `config.storage.topic`: 1, tidak dapat dikonfigurasi, harus topik partisi tunggal
  - `offset.storage.topic`: 25, dapat dikonfigurasi dengan menyediakan `offset.storage.partitions`
  - `status.storage.topic`: 5, dapat dikonfigurasi dengan menyediakan `status.storage.partitions`
- Menghapus topik secara manual - Amazon MSK Connect membuat topik internal Kafka connect baru (nama topik dimulai dengan`__amazon_msk_connect`) pada setiap penyebaran konektor. Topik lama yang dilampirkan ke konektor yang dihapus tidak dihapus secara otomatis karena topik internal, sepertio`ffset.storage.topic`, dapat digunakan kembali di antara konektor. Namun, Anda dapat secara manual menghapus topik internal yang tidak digunakan yang dibuat oleh MSK Connect. Topik internal diberi nama mengikuti format`__amazon_msk_connect_<offsets|status|configs>_`*`connector_name_connector_id`*.

  Ekspresi reguler `__amazon_msk_connect_<offsets|status|configs>_`*`connector_name_connector_id`* dapat digunakan untuk menghapus topik internal. Anda tidak boleh menghapus topik internal yang saat ini digunakan oleh konektor yang sedang berjalan.
- Menggunakan nama yang sama untuk topik internal yang dibuat oleh MSK Connect — Jika Anda ingin menggunakan kembali topik penyimpanan offset untuk menggunakan offset dari konektor yang dibuat sebelumnya, Anda harus memberikan konektor baru nama yang sama dengan konektor lama. `offset.storage.topic`Properti dapat diatur menggunakan konfigurasi pekerja untuk menetapkan nama yang sama ke `offset.storage.topic` dan digunakan kembali di antara konektor yang berbeda. Konfigurasi ini dijelaskan dalam Mengelola offset konektor. MSK Connect tidak mengizinkan konektor yang berbeda untuk berbagi `config.storage.topic` dan`status.storage.topic`. Topik-topik tersebut dibuat setiap kali Anda membuat konektor baru di MSKC. Mereka secara otomatis dinamai mengikuti format`__amazon_msk_connect_<status|configs>_`*`connector_name_connector_id`*, dan begitu juga berbeda di berbagai konektor yang Anda buat.

Eksternalisasi informasi sensitif menggunakan penyedia konfigurasi

Contoh ini menunjukkan cara mengeksternalisasi informasi sensitif untuk Amazon MSK Connect menggunakan penyedia konfigurasi open source. Penyedia konfigurasi memungkinkan Anda menentukan variabel alih-alih teks biasa dalam konfigurasi konektor atau pekerja, dan pekerja yang berjalan di konektor Anda menyelesaikan variabel ini saat runtime. Ini mencegah kredensyal

dan rahasia lainnya disimpan dalam teks biasa. Penyedia konfigurasi dalam contoh mendukung pengambilan parameter konfigurasi dari AWS Secrets Manager, Amazon S3 dan Systems Manager (SSM). Pada Langkah 2, Anda dapat melihat cara mengatur penyimpanan dan pengambilan informasi sensitif untuk layanan yang ingin Anda konfigurasi.

Topik

Langkah 1: Buat plugin khusus dan unggah ke S3

Untuk membuat plugin khusus, buat file zip yang berisi konektor dan msk-config-provider dengan menjalankan perintah berikut di mesin lokal Anda.

Untuk membuat plugin kustom menggunakan jendela terminal dan Debezium sebagai konektor

Gunakan AWS CLI untuk menjalankan perintah sebagai pengguna super dengan kredensyal yang memungkinkan Anda mengakses bucket S3 Anda. AWS Untuk informasi tentang menginstal dan menyiapkan AWS CLI, lihat Memulai AWS CLI di Panduan Pengguna.AWS Command Line Interface Untuk informasi tentang penggunaan AWS CLI dengan Amazon S3, lihat Menggunakan Amazon S3 dengan AWS CLI di Panduan Pengguna.AWS Command Line Interface

1. Di jendela terminal, buat folder bernama `custom-plugin` di ruang kerja Anda menggunakan perintah berikut.

   ```
   mkdir custom-plugin && cd custom-plugin
   ```

2. Unduh rilis stabil terbaru dari MySQL Connector Plug-in dari situs Debezium menggunakan perintah berikut.

   ```
   wget https://repo1.maven.org/maven2/io/debezium/debezium-connectormysql/
   2.2.0.Final/debezium-connector-mysql-2.2.0.Final-plugin.tar.gz
   ```

   Ekstrak file gzip yang diunduh di `custom-plugin` folder menggunakan perintah berikut.

```
tar xzf debezium-connector-mysql-2.2.0.Final-plugin.tar.gz
```

3. Unduh file zip penyedia konfigurasi MSK menggunakan perintah berikut.

```
wget https://github.com/aws-samples/msk-config-providers/releases/download/r0.1.0/
msk-config-providers-0.1.0-with-dependencies.zip
```

Ekstrak file zip yang diunduh di `custom-plugin` folder menggunakan perintah follwoing.

```
unzip msk-config-providers-0.1.0-with-dependencies.zip
```

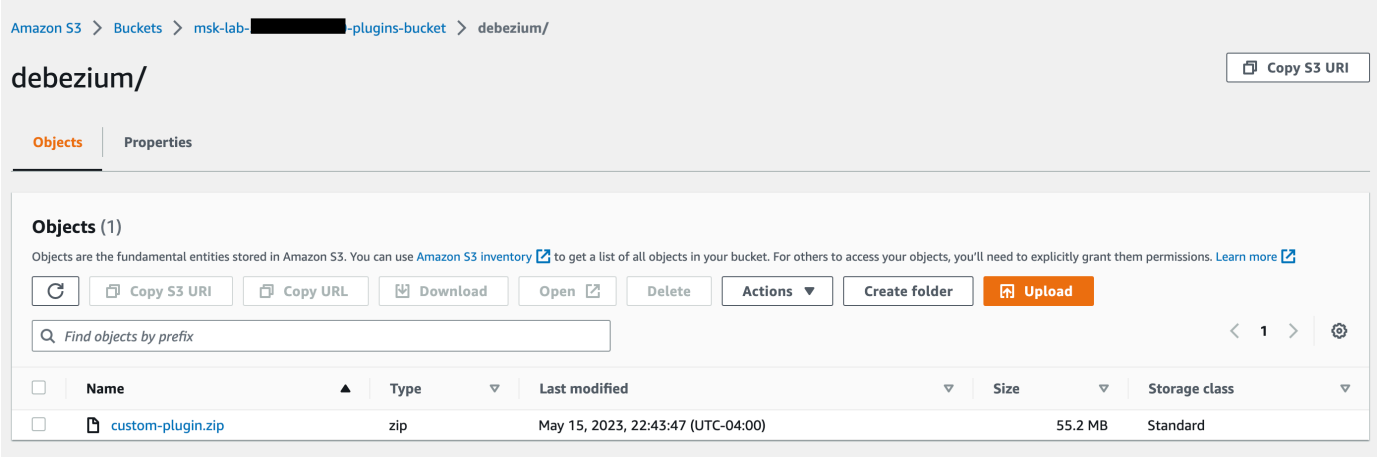
4. Zip konten penyedia konfigurasi MSK dari langkah di atas dan konektor khusus ke dalam satu file bernama. `custom-plugin.zip`

```
zip -r ../custom-plugin.zip *
```

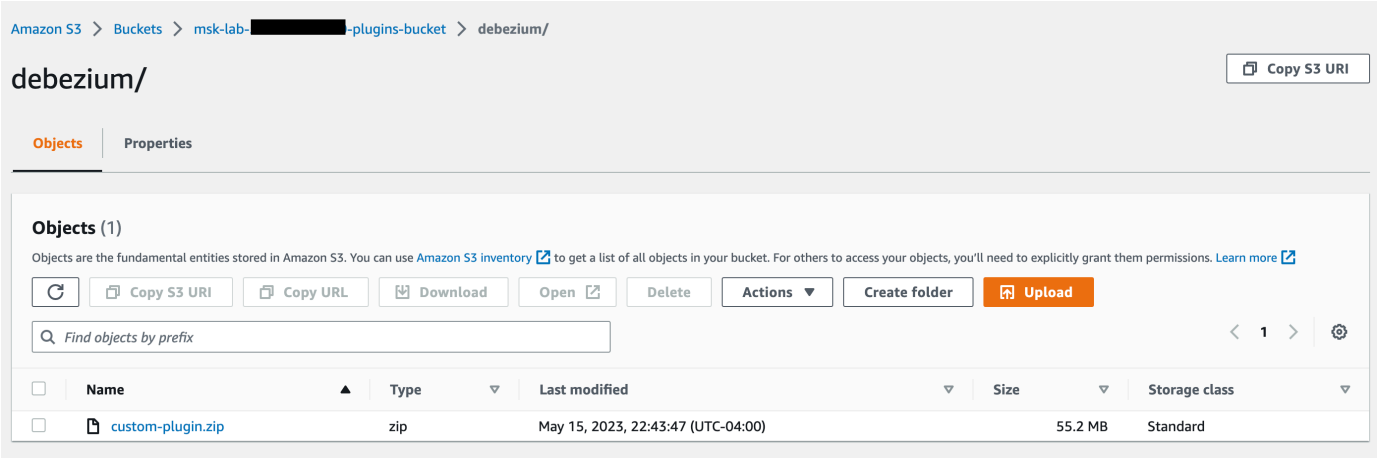
5. Unggah file ke S3 untuk direferensikan nanti.

```
aws s3 cp ../custom-plugin.zip s3:<S3_URI_BUCKET_LOCATION>
```

6. Di konsol MSK Amazon, di bawah bagian MSK Connect, pilih Plugin Kustom, lalu pilih Buat plugin khusus dan telusuri ember s3: **< *S3_URI_BUCKET_LOCATION > S3 untuk memilih file*** ZIP plugin khusus yang baru saja Anda unggah.



7. Masukkan **`debezium-custom-plugin`** untuk nama plugin. Secara opsional, masukkan deskripsi dan pilih Buat Plugin Kustom.



Langkah 2: Konfigurasikan parameter dan izin untuk penyedia yang berbeda

Anda dapat mengonfigurasi nilai parameter dalam tiga layanan ini:

- Secrets Manager
- Penyimpanan Parameter Systems Manager
- S3 - Layanan Penyimpanan Sederhana

Pilih salah satu tab di bawah ini untuk petunjuk tentang pengaturan parameter dan izin yang relevan untuk layanan tersebut.

Configure in Secrets Manager

Untuk mengkonfigurasi nilai parameter di Secrets Manager

1. Buka konsol Secrets Manager.
2. Buat rahasia baru untuk menyimpan kredensyal atau rahasia Anda. Untuk petunjuk, lihat Membuat AWS Secrets Manager rahasia di Panduan AWS Secrets Manager Pengguna.
3. Salin ARN rahasia Anda.
4. Tambahkan izin Secrets Manager dari kebijakan contoh berikut ke peran eksekusi Layanan Anda. Ganti *<arn:aws:secretsmanager:us-east- 1:123456789000:secret : -1234>*dengan ARN rahasia Anda. MySecret
5. Tambahkan konfigurasi pekerja dan instruksi konektor.

```
{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "secretsmanager:GetResourcePolicy",
                    "secretsmanager:GetSecretValue",
                    "secretsmanager:DescribeSecret",
                    "secretsmanager:ListSecretVersionIds"
                ],
                "Resource": [
                "<arn:aws:secretsmanager:us-
east-1:123456789000:secret:MySecret-1234>"
                ]
            }
        ]
    }
```

6. Untuk menggunakan penyedia konfigurasi Secrets Manager, salin baris kode berikut ke kotak teks konfigurasi pekerja di Langkah 3:

```
# define name of config provider:

config.providers = secretsmanager

# provide implementation classes for secrets manager:

config.providers.secretsmanager.class =
 com.amazonaws.kafka.config.providers.SecretsManagerConfigProvider

# configure a config provider (if it needs additional initialization), for
 example you can provide a region where the secrets or parameters are located:

config.providers.secretsmanager.param.region = us-east-1
```

7. Untuk penyedia konfigurasi manajer rahasia, salin baris kode berikut dalam konfigurasi konektor di Langkah 4.

```
#Example implementation for secrets manager variable
database.hostname=${secretsmanager:MSKAuroraDBCredentials:username}
```

```
database.password=${secretsmanager:MSKAuroraDBCredentials:password}
```

Anda juga dapat menggunakan langkah di atas dengan lebih banyak penyedia konfigurasi.

Configure in Systems Manager Parameter Store

Untuk mengkonfigurasi nilai parameter di Systems Manager Parameter Store

1. Buka konsol System Manager.
2. Di panel navigasi, pilih Penyimpanan Parameter.
3. Buat parameter baru untuk disimpan di Systems Manager. Untuk petunjuknya, lihat Membuat parameter Systems Manager (konsol) di Panduan AWS Systems Manager Pengguna.
4. Salin ARN parameter Anda.
5. Tambahkan izin Systems Manager dari kebijakan contoh berikut ke peran eksekusi Layanan Anda. Ganti *<arn:aws:ssm:us-east- MyParameterName 1:123456789000:parameter/* > dengan ARN parameter Anda.

```
{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Sid": "VisualEditor0",
                "Effect": "Allow",
                "Action": [
                    "ssm:GetParameterHistory",
                    "ssm:GetParametersByPath",
                    "ssm:GetParameters",
                    "ssm:GetParameter"
                ],
                "Resource": "arn:aws:ssm:us-east-1:123456789000:parameter/
MyParameterName"
            }
        ]
    }
```

6. Untuk menggunakan penyedia konfigurasi penyimpanan parameter, salin baris kode berikut ke kotak teks konfigurasi pekerja di Langkah 3:

```
# define name of config provider:
```

```
config.providers = ssm

# provide implementation classes for parameter store:

config.providers.ssm.class =
 com.amazonaws.kafka.config.providers.SsmParamStoreConfigProvider

# configure a config provider (if it needs additional initialization), for
 example you can provide a region where the secrets or parameters are located:

config.providers.ssm.param.region = us-east-1
```

7. Untuk penyedia konfigurasi penyimpanan parameter, salin baris kode berikut dalam konfigurasi konektor di Langkah 5.

```
#Example implementation for parameter store variable
schema.history.internal.kafka.bootstrap.servers=
${ssm::MSKBootstrapServerAddress}
```

Anda juga dapat menggabungkan dua langkah di atas dengan lebih banyak penyedia konfigurasi.

Configure in Amazon S3

Untuk mengonfigurasi objek/file di Amazon S3

1. Buka konsol Amazon S3.
2. Unggah objek Anda ke ember di S3. Untuk petunjuk, lihat Mengunggah objek.
3. Salin ARN objek Anda.
4. Tambahkan izin Baca Objek Amazon S3 dari kebijakan contoh berikut ke peran eksekusi Layanan Anda. Ganti <arn:aws:s3: ::MY_S3_BUCKET/path/to/custom-plugin.zip> dengan ARN objek Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
            {
                "Sid": "VisualEditor0",
                "Effect": "Allow",
                "Action": "s3:GetObject",
```

```
                "Resource": "<arn:aws:s3:::MY_S3_BUCKET/path/to/custom-
plugin.zip>"
            }
        ]
    }
```

5. Untuk menggunakan penyedia konfigurasi Amazon S3, salin baris kode berikut ke kotak teks konfigurasi pekerja di Langkah 3:

```
# define name of config provider:

config.providers = s3import
# provide implementation classes for S3:

config.providers.s3import.class =
 com.amazonaws.kafka.config.providers.S3ImportConfigProvider
```

6. Untuk penyedia konfigurasi Amazon S3, salin baris kode berikut ke konfigurasi konektor di Langkah 4.

```
#Example implementation for S3 object

database.ssl.truststore.location = ${s3import:us-west-2:my_cert_bucket/path/to/
trustore_unique_filename.jks}
```

Anda juga dapat menggabungkan dua langkah di atas dengan lebih banyak penyedia konfigurasi.

Langkah 3: Buat konfigurasi pekerja khusus dengan informasi tentang penyedia konfigurasi Anda

1. Pilih Konfigurasi pekerja di bawah bagian Amazon MSK Connect.
2. Pilih Buat konfigurasi pekerja.
3. Masukkan `SourceDebeziumCustomConfig` di kotak teks Nama Konfigurasi Pekerja. Deskripsi adalah opsional.
4. Salin kode konfigurasi yang relevan berdasarkan penyedia yang diinginkan, dan tempelkan di kotak teks konfigurasi Pekerja.

5. Ini adalah contoh konfigurasi pekerja untuk ketiga penyedia:

```
key.converter=org.apache.kafka.connect.storage.StringConverter
key.converter.schemas.enable=false
value.converter=org.apache.kafka.connect.json.JsonConverter
value.converter.schemas.enable=false
offset.storage.topic=offsets_my_debezium_source_connector

# define names of config providers:

config.providers=secretsmanager,ssm,s3import

# provide implementation classes for each provider:

config.providers.secretsmanager.class    =
 com.amazonaws.kafka.config.providers.SecretsManagerConfigProvider
config.providers.ssm.class               =
 com.amazonaws.kafka.config.providers.SsmParamStoreConfigProvider
config.providers.s3import.class          =
 com.amazonaws.kafka.config.providers.S3ImportConfigProvider

# configure a config provider (if it needs additional initialization), for example
 you can provide a region where the secrets or parameters are located:


config.providers.secretsmanager.param.region = us-east-1
config.providers.ssm.param.region = us-east-1
```

6. Klik pada Buat konfigurasi pekerja.

Langkah 4: Buat konektor

1. Buat konektor baru menggunakan instruksi di Buat konektor baru.
2. Pilih `custom-plugin.zip` file yang Anda unggah ke bucket S3 Anda ??? sebagai sumber untuk plugin kustom.
3. Salin kode konfigurasi yang relevan berdasarkan penyedia yang diinginkan, dan tempel di bidang konfigurasi Konektor.
4. Ini adalah contoh untuk konfigurasi konektor untuk ketiga penyedia:

```
#Example implementation for parameter store variable
```

```
schema.history.internal.kafka.bootstrap.servers=${ssm::MSKBootstrapServerAddress}

#Example implementation for secrets manager variable
database.hostname=${secretsmanager:MSKAuroraDBCredentials:username}

database.password=${secretsmanager:MSKAuroraDBCredentials:password}

#Example implementation for Amazon S3 file/object
database.ssl.truststore.location = ${s3import:us-west-2:my_cert_bucket/path/to/
trustore_unique_filename.jks}
```

5. Pilih Gunakan konfigurasi khusus dan pilih SourceDebeziumCustomConfigdari tarik-turun Konfigurasi Pekerja.
6. Ikuti langkah-langkah yang tersisa dari instruksi di Buat konektor.

Pertimbangan

Pertimbangkan hal berikut saat menggunakan penyedia konfigurasi MSK dengan Amazon MSK Connect:

- Tetapkan izin yang sesuai saat menggunakan penyedia konfigurasi ke Peran Eksekusi Layanan IAM.
- Tentukan penyedia konfigurasi dalam konfigurasi pekerja dan implementasinya dalam konfigurasi konektor.
- Nilai konfigurasi sensitif dapat muncul di log konektor jika plugin tidak mendefinisikan nilai-nilai tersebut sebagai rahasia. Kafka Connect memperlakukan nilai konfigurasi yang tidak ditentukan sama dengan nilai plaintext lainnya. Untuk mempelajari selengkapnya, lihat Mencegah rahasia muncul di log konektor.
- Secara default, MSK Connect sering me-restart konektor saat konektor menggunakan penyedia konfigurasi. Untuk menonaktifkan perilaku restart ini, Anda dapat mengatur `config.action.reload` nilai ke `none` dalam konfigurasi konektor Anda.

Peran dan kebijakan IAM untuk MSK Connect

Topik

- Peran eksekusi layanan
- Contoh kebijakan IAM untuk MSK Connect

- Pencegahan confused deputy lintas layanan
- AWS kebijakan terkelola untuk MSK Connect
- Menggunakan peran terkait layanan untuk MSK Connect

Peran eksekusi layanan

> **Note**
>
> Amazon MSK Connect tidak mendukung penggunaan peran Service-linked sebagai peran eksekusi layanan. Anda harus membuat peran eksekusi layanan terpisah. Untuk petunjuk tentang cara membuat peran IAM kustom, lihat Membuat peran untuk mendelegasikan izin ke AWS layanan di Panduan Pengguna IAM.

Saat Anda membuat konektor dengan MSK Connect, Anda diminta untuk menentukan peran AWS Identity and Access Management (IAM) untuk digunakan dengannya. Peran eksekusi layanan Anda harus memiliki kebijakan kepercayaan berikut sehingga MSK Connect dapat menerapkannya. Untuk informasi tentang kunci konteks kondisi dalam kebijakan ini, lihatthe section called “Pencegahan confused deputy lintas layanan”.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "kafkaconnect.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "Account-ID"
        },
        "ArnLike": {
          "aws:SourceArn": "MSK-Connector-ARN"
        }
      }
    }
  ]
```

```
}
```

Jika klaster MSK Amazon yang ingin Anda gunakan dengan konektor adalah klaster yang menggunakan autentikasi IAM, maka Anda harus menambahkan kebijakan izin berikut ke peran eksekusi layanan konektor. Untuk informasi tentang cara menemukan UUID klaster Anda dan cara membuat ARN topik, lihat. the section called “Sumber daya”

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:Connect",
                "kafka-cluster:DescribeCluster"
            ],
            "Resource": [
                "cluster-arn"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:ReadData",
                "kafka-cluster:DescribeTopic"
            ],
            "Resource": [
                "ARN of the topic that you want a sink connector to read from"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:WriteData",
                "kafka-cluster:DescribeTopic"
            ],
            "Resource": [
                "ARN of the topic that you want a source connector to write to"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "kafka-cluster:CreateTopic",
                "kafka-cluster:WriteData",
                "kafka-cluster:ReadData",
                "kafka-cluster:DescribeTopic"
            ],
            "Resource": [
                "arn:aws:kafka:region:account-id:topic/cluster-name/cluster-uuid/
__amazon_msk_connect_*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:AlterGroup",
                "kafka-cluster:DescribeGroup"
            ],
            "Resource": [
                "arn:aws:kafka:region:account-id:group/cluster-name/cluster-uuid/
__amazon_msk_connect_*",
                "arn:aws:kafka:region:account-id:group/cluster-name/cluster-uuid/
connect-*"
            ]
        }
    ]
}
```

Bergantung pada jenis konektornya, Anda mungkin juga perlu melampirkan kebijakan izin ke peran eksekusi layanan yang memungkinkannya mengakses AWS sumber daya. Misalnya, jika konektor Anda perlu mengirim data ke bucket S3, maka peran eksekusi layanan harus memiliki kebijakan izin yang memberikan izin untuk menulis ke bucket tersebut. Untuk tujuan pengujian, Anda dapat menggunakan salah satu kebijakan IAM pra-bangun yang memberikan akses penuh, seperti. `arn:aws:iam::aws:policy/AmazonS3FullAccess` Namun, untuk tujuan keamanan, kami menyarankan Anda menggunakan kebijakan paling ketat yang memungkinkan konektor Anda membaca dari AWS sumber atau menulis ke AWS wastafel.

Contoh kebijakan IAM untuk MSK Connect

Untuk memberi pengguna non-admin akses penuh ke semua fungsionalitas MSK Connect, lampirkan kebijakan seperti berikut ini ke peran IAM pengguna.

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

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kafkaconnect:*",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "ec2:DescribeSecurityGroups",
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries",
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/
kafkaconnect.amazonaws.com/AWSServiceRoleForKafkaConnect*",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "kafkaconnect.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/
kafkaconnect.amazonaws.com/AWSServiceRoleForKafkaConnect*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/
delivery.logs.amazonaws.com/AWSServiceRoleForLogDelivery*",
```

```
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "delivery.logs.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutBucketPolicy",
                "s3:GetBucketPolicy"
            ],
            "Resource": "ARN of the Amazon S3 bucket to which you want MSK Connect to
 deliver logs"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "ARN of the service execution role"
        },
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "ARN of the Amazon S3 object that corresponds to the custom
 plugin that you want to use for creating connectors"
        },
        {
            "Effect": "Allow",
            "Action": "firehose:TagDeliveryStream",
            "Resource": "ARN of the Firehose delivery stream to which you want MSK
 Connect to deliver logs"
        }
    ]
}
```

Pencegahan confused deputy lintas layanan

Masalah confused deputy adalah masalah keamanan saat entitas yang tidak memiliki izin untuk melakukan suatu tindakan dapat memaksa entitas yang lebih berhak untuk melakukan tindakan tersebut. Pada tahun AWS, peniruan lintas layanan dapat mengakibatkan masalah wakil yang membingungkan. Peniruan identitas lintas layanan dapat terjadi ketika satu layanan (layanan yang dipanggil) memanggil layanan lain (layanan yang dipanggil). Layanan pemanggilan dapat

dimanipulasi menggunakan izinnya untuk bertindak pada sumber daya pelanggan lain dengan cara yang seharusnya tidak dilakukannya kecuali bila memiliki izin untuk mengakses. Untuk mencegah hal ini, AWS menyediakan alat yang membantu Anda melindungi data untuk semua layanan dengan pengguna utama layanan yang telah diberi akses ke sumber daya di akun Anda.

Sebaiknya gunakan kunci konteks kondisi `aws:SourceAccount`global `aws:SourceArn`dan global dalam kebijakan sumber daya untuk membatasi izin yang diberikan MSK Connect kepada layanan lain ke sumber daya. Jika `aws:SourceArn` nilai tidak berisi ID akun (misalnya, ARN bucket Amazon S3 tidak berisi ID akun), Anda harus menggunakan kedua kunci konteks kondisi global untuk membatasi izin. Jika Anda menggunakan kunci konteks kondisi global dan nilai `aws:SourceArn` berisi ID akun, nilai `aws:SourceAccount` dan akun dalam nilai `aws:SourceArn` harus menggunakan ID akun yang sama saat digunakan dalam pernyataan kebijakan yang sama. Gunakan `aws:SourceArn` jika Anda hanya ingin satu sumber daya dikaitkan dengan akses lintas layanan. Gunakan `aws:SourceAccount` jika Anda ingin mengizinkan sumber daya apa pun di akun tersebut dikaitkan dengan penggunaan lintas layanan.

Dalam kasus MSK Connect, nilai `aws:SourceArn` harus berupa konektor MSK.

Cara paling efektif untuk melindungi dari masalah confused deputy adalah dengan menggunakan kunci konteks kondisi global `aws:SourceArn` dengan ARN lengkap sumber daya. Jika Anda tidak mengetahui ARN lengkap sumber daya atau jika Anda menentukan beberapa sumber daya, gunakan kunci kondisi konteks `aws:SourceArn` global dengan wildcard (*) untuk bagian ARN yang tidak diketahui. Misalnya, *`arn:aws:kafkaconnect:us-east- 1:123456789012:connector/*`* *mewakili semua konektor yang termasuk dalam akun dengan ID 123456789012* di Wilayah AS Timur (Virginia N.).

Contoh berikut menunjukkan bagaimana Anda dapat menggunakan `aws:SourceArn` dan kunci konteks kondisi `aws:SourceAccount` global di MSK Connect untuk mencegah masalah wakil bingung. Ganti *`Account-ID`* dan *`MSK-Connector-ARN`* dengan informasi Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": " kafkaconnect.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
```

```
        "StringEquals": {
          "aws:SourceAccount": "Account-ID"
        },
        "ArnLike": {
          "aws:SourceArn": "MSK-Connector-ARN"
        }
      }
    }
  ]
}
```

AWS kebijakan terkelola untuk MSK Connect

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, pemutakhiran 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 AWS kebijakan yang dikelola dalam Panduan Pengguna IAM.

AWS kebijakan terkelola: AmazonMSK ConnectReadOnlyAccess

Kebijakan ini memberi pengguna izin yang diperlukan untuk mencantumkan dan menjelaskan sumber daya MSK Connect.

Anda dapat melampirkan kebijakan `AmazonMSKConnectReadOnlyAccess` ke identitas IAM Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "kafkaconnect:ListConnectors",
                "kafkaconnect:ListCustomPlugins",
                "kafkaconnect:ListWorkerConfigurations"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafkaconnect:DescribeConnector"
            ],
            "Resource": [
                "arn:aws:kafkaconnect:*:*:connector/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafkaconnect:DescribeCustomPlugin"
            ],
            "Resource": [
                "arn:aws:kafkaconnect:*:*:custom-plugin/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafkaconnect:DescribeWorkerConfiguration"
            ],
            "Resource": [
                "arn:aws:kafkaconnect:*:*:worker-configuration/*"
            ]
        }
    ]
}
```

AWS kebijakan terkelola: KafkaConnectServiceRolePolicy

Kebijakan ini memberi layanan MSK Connect izin yang diperlukan untuk membuat dan mengelola antarmuka jaringan yang memiliki tag. `AmazonMSKConnectManaged:true` Antarmuka jaringan ini memberikan akses jaringan MSK Connect ke sumber daya di VPC Amazon Anda, seperti cluster Apache Kafka atau sumber atau wastafel.

Anda tidak dapat melampirkan KafkaConnectServiceRolePolicy ke entitas IAM Anda. Kebijakan ini dilampirkan pada peran terkait layanan yang memungkinkan MSK Connect melakukan tindakan atas nama Anda.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Action": [
    "ec2:CreateNetworkInterface"
   ],
   "Resource": "arn:aws:ec2:*:*:network-interface/*",
   "Condition": {
    "StringEquals": {
     "aws:RequestTag/AmazonMSKConnectManaged": "true"
    },
    "ForAllValues:StringEquals": {
     "aws:TagKeys": "AmazonMSKConnectManaged"
    }
   }
  },
  {
   "Effect": "Allow",
   "Action": [
    "ec2:CreateNetworkInterface"
   ],
   "Resource": [
    "arn:aws:ec2:*:*:subnet/*",
    "arn:aws:ec2:*:*:security-group/*"
   ]
  },
  {
   "Effect": "Allow",
   "Action": [
    "ec2:CreateTags"
   ],
   "Resource": "arn:aws:ec2:*:*:network-interface/*",
   "Condition": {
    "StringEquals": {
     "ec2:CreateAction": "CreateNetworkInterface"
    }
   }
```

```
  },
  {
   "Effect": "Allow",
   "Action": [
    "ec2:DescribeNetworkInterfaces",
    "ec2:CreateNetworkInterfacePermission",
    "ec2:AttachNetworkInterface",
    "ec2:DetachNetworkInterface",
    "ec2:DeleteNetworkInterface"
   ],
   "Resource": "arn:aws:ec2:*:*:network-interface/*",
   "Condition": {
    "StringEquals": {
     "ec2:ResourceTag/AmazonMSKConnectManaged": "true"
    }
   }
  }
 ]
}
```

MSK Connect memperbarui kebijakan AWS terkelola

Lihat detail tentang pembaruan kebijakan AWS terkelola untuk MSK Connect sejak layanan ini mulai melacak perubahan ini.

Perubahan	Deskripsi	Tanggal
MSK Connect memperbarui kebijakan hanya-baca	MSK Connect memperbarui ConnectReadOnlyAccess kebijakan AmazonMSK untuk menghapus pembatasan operasi listing.	13 Oktober 2021
MSK Connect mulai melacak perubahan	MSK Connect mulai melacak perubahan untuk kebijakan AWS terkelolanya.	14 September 2021

Menggunakan peran terkait layanan untuk MSK Connect

Amazon MSK Connect menggunakan peran AWS Identity and Access Management terkait layanan (IAM). Peran terkait layanan adalah jenis peran IAM unik yang ditautkan langsung ke MSK Connect. Peran terkait layanan telah ditentukan sebelumnya oleh MSK Connect dan mencakup semua izin yang diperlukan layanan untuk memanggil layanan lain AWS atas nama Anda.

Peran terkait layanan membuat pengaturan MSK Connect menjadi lebih mudah karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. MSK Connect mendefinisikan izin dari peran terkait layanan, dan kecuali ditentukan lain, hanya MSK Connect yang dapat mengambil perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Untuk informasi tentang layanan lain yang mendukung peran terkait layanan, lihat Layanan AWS yang Berfungsi dengan IAM dan cari layanan yang memiliki Ya di kolom Peran Terkait Layanan. Pilih Ya bersama tautan untuk melihat dokumentasi peran tertaut layanan untuk layanan tersebut.

Izin peran terkait layanan untuk MSK Connect

MSK Connect menggunakan peran terkait layanan bernama — AWSServiceRoleForKafkaConnectMemungkinkan Amazon MSK Connect mengakses sumber daya Amazon atas nama Anda.

Peran AWSServiceRoleForKafkaConnect terkait layanan mempercayai `kafkaconnect.amazonaws.com` layanan untuk mengambil peran tersebut.

Untuk informasi tentang kebijakan izin yang digunakan peran, lihatthe section called “KafkaConnectServiceRolePolicy”.

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, silakan lihat Izin Peran Tertaut Layanan di Panduan Pengguna IAM.

Membuat peran terkait layanan untuk MSK Connect

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda membuat konektor di, API AWS Management Console, atau AWS API AWS CLI, MSK Connect membuat peran terkait layanan untuk Anda.

Jika Anda menghapus peran tertaut layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Saat Anda membuat konektor, MSK Connect membuat peran terkait layanan untuk Anda lagi.

Mengedit peran terkait layanan untuk MSK Connect

MSK Connect tidak mengizinkan Anda mengedit peran AWSServiceRoleForKafkaConnect terkait layanan. Setelah membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat menyunting penjelasan peran menggunakan IAM. Untuk informasi selengkapnya, lihat Mengedit Peran Tertaut Layanan dalam Panduan Pengguna IAM.

Menghapus peran terkait layanan untuk MSK Connect

Anda dapat menggunakan konsol IAM, AWS CLI atau AWS API untuk menghapus peran terkait layanan secara manual. Untuk melakukan ini, Anda harus terlebih dahulu menghapus semua konektor MSK Connect Anda secara manual, dan kemudian Anda dapat menghapus peran secara manual. Untuk informasi selengkapnya, silakan lihat Menghapus Peran Terkait Layanan di Panduan Pengguna IAM.

Wilayah yang Didukung untuk peran terkait layanan MSK Connect

MSK Connect mendukung penggunaan peran terkait layanan di semua wilayah tempat layanan tersedia. Untuk informasi lebih lanjut, lihat Wilayah dan Titik Akhir AWS.

Mengaktifkan akses internet untuk Amazon MSK Connect

Jika konektor Anda untuk Amazon MSK Connect memerlukan akses ke internet, kami sarankan Anda menggunakan pengaturan Amazon Virtual Private Cloud (VPC) berikut untuk mengaktifkan akses tersebut.

- Konfigurasikan konektor Anda dengan subnet pribadi.
- Buat gateway NAT publik atau instans NAT untuk VPC Anda di subnet publik. Untuk informasi selengkapnya, lihat halaman Connect subnet ke internet atau VPC lain menggunakan perangkat NAT di Panduan Pengguna. Amazon Virtual Private Cloud
- Izinkan lalu lintas keluar dari subnet pribadi Anda ke gateway atau instance NAT Anda.

Menyiapkan gateway NAT untuk Amazon MSK Connect

Langkah-langkah berikut menunjukkan cara mengatur gateway NAT untuk mengaktifkan akses internet untuk konektor. Anda harus menyelesaikan langkah-langkah ini sebelum membuat konektor di subnet pribadi.

Prasyarat

Pastikan Anda memiliki item berikut.

- ID dari Amazon Virtual Private Cloud (VPC) yang terkait dengan cluster Anda. Misalnya, vpc-123456ab.
- ID subnet pribadi di VPC Anda. Misalnya, subnet-a1b2c3de, subnet-f4g5h6ij, dll. Anda harus mengkonfigurasi konektor Anda dengan subnet pribadi.

Untuk mengaktifkan akses internet untuk konektor Anda

1. Buka Amazon Virtual Private Cloud konsol di https://console.aws.amazon.com/vpc/.
2. Buat subnet publik untuk gateway NAT Anda dengan nama deskriptif, dan catat ID subnet. Untuk petunjuk terperinci, lihat Membuat subnet di VPC Anda.
3. Buat gateway internet sehingga VPC Anda dapat berkomunikasi dengan internet, dan catat ID gateway. Lampirkan gateway internet ke VPC Anda. Untuk petunjuk, lihat Membuat dan melampirkan gateway internet.
4. Menyediakan gateway NAT publik sehingga host di subnet pribadi Anda dapat mencapai subnet publik Anda. Saat Anda membuat gateway NAT, pilih subnet publik yang Anda buat sebelumnya. Untuk petunjuk, silakan lihat Membuat gateway NAT.
5. Konfigurasikan tabel rute Anda. Anda harus memiliki dua tabel rute secara total untuk menyelesaikan pengaturan ini. Anda seharusnya sudah memiliki tabel rute utama yang dibuat secara otomatis bersamaan dengan VPC Anda. Pada langkah ini Anda membuat tabel rute tambahan untuk subnet publik Anda.

    a. Gunakan pengaturan berikut untuk memodifikasi tabel rute utama VPC Anda sehingga subnet pribadi Anda merutekan lalu lintas ke gateway NAT Anda. Untuk petunjuk, lihat Bekerja dengan tabel rute di Panduan Amazon Virtual Private CloudPengguna.

Tabel rute MSKC pribadi

Properti	Nilai
Tag nama	Kami menyarankan Anda memberikan tabel rute ini tag nama deskriptif untuk membantu Anda mengidentifikasinya. Misalnya, MSKC Pribadi.
Subnet terkait	Subnet pribadi Anda
Rute untuk mengaktifkan akses internet untuk MSK Connect	• Tujuan: 0.0.0.0/0 • Target: ID gateway NAT Anda. Misalnya, nat-12a345bc6789efg1h.
Rute lokal untuk lalu lintas internal	• Tujuan: 10.0.0.0/16. Nilai ini mungkin berbeda tergantung pada blok CIDR VPC Anda. • Target: Lokal

b. Ikuti petunjuk di Buat tabel rute kustom untuk membuat tabel rute untuk subnet publik Anda. Saat Anda membuat tabel, masukkan nama deskriptif di kolom Tag nama untuk membantu Anda mengidentifikasi subnet mana yang terkait dengan tabel tersebut. Misalnya, MSKC Publik.

c. Konfigurasikan tabel rute MSKC Publik Anda menggunakan pengaturan berikut.

Properti	Nilai
Tag nama	MSKC publik atau nama deskriptif lain yang Anda pilih
Subnet terkait	Subnet publik Anda dengan gateway NAT
Rute untuk mengaktifkan akses internet untuk MSK Connect	• Tujuan: 0.0.0.0/0 • Target: ID gateway internet Anda. Misalnya, igw-1a234bc5.

Properti	Nilai
Rute lokal untuk lalu lintas internal	• Tujuan: 10.0.0.0/16. Nilai ini mungkin berbeda tergantung pada blok CIDR VPC Anda. • Target: Lokal

Nama host DNS Pribadi

Dengan dukungan nama host DNS Pribadi di MSK Connect, Anda dapat mengonfigurasi konektor untuk mereferensikan nama domain publik atau pribadi. Support tergantung pada server DNS yang ditentukan dalam set opsi VPC DHCP.

Set opsi DHCP adalah sekelompok konfigurasi jaringan yang digunakan instans EC2 dalam VPC untuk berkomunikasi melalui jaringan VPC. Setiap VPC memiliki set opsi DHCP default, tetapi Anda dapat membuat set opsi DHCP khusus jika Anda ingin instance di VPC menggunakan server DNS yang berbeda untuk resolusi nama domain, bukan server DNS yang disediakan Amazon. Lihat set opsi DHCP di Amazon VPC.

Sebelum kemampuan/fitur resolusi DNS Pribadi disertakan dengan MSK Connect, konektor menggunakan resolver DNS VPC layanan untuk kueri DNS dari konektor pelanggan. Konektor tidak menggunakan server DNS yang ditentukan dalam set opsi VPC DHCP pelanggan untuk resolusi DNS.

Konektor hanya dapat mereferensikan nama host dalam konfigurasi konektor pelanggan atau plugin yang dapat diselesaikan secara publik. Mereka tidak dapat menyelesaikan nama host pribadi yang ditentukan di zona yang dihosting secara pribadi atau menggunakan server DNS di jaringan pelanggan lain.

Tanpa DNS Pribadi, pelanggan yang memilih untuk membuat database, gudang data, dan sistem seperti Secrets Manager di VPC mereka sendiri tidak dapat diakses oleh internet, tidak dapat bekerja dengan konektor MSK. Pelanggan sering menggunakan nama host DNS pribadi untuk mematuhi postur keamanan perusahaan.

Topik

- Mengkonfigurasi opsi VPC DHCP yang ditetapkan untuk konektor Anda
- Atribut DNS untuk VPC Anda

- Penanganan kegagalan

Mengkonfigurasi opsi VPC DHCP yang ditetapkan untuk konektor Anda

Konektor secara otomatis menggunakan server DNS yang ditentukan dalam opsi VPC DHCP mereka yang ditetapkan ketika konektor dibuat. Sebelum Anda membuat konektor, pastikan Anda mengonfigurasi opsi VPC DHCP yang ditetapkan untuk persyaratan resolusi nama host DNS konektor Anda.

Konektor yang Anda buat sebelum fitur hostname DNS Pribadi tersedia di MSK Connect terus menggunakan konfigurasi resolusi DNS sebelumnya tanpa perlu modifikasi.

Jika Anda hanya memerlukan resolusi nama host DNS yang dapat diselesaikan secara publik di konektor Anda, untuk pengaturan yang lebih mudah, kami sarankan menggunakan VPC default akun Anda saat Anda membuat konektor. Lihat Server DNS Amazon di Panduan Pengguna Amazon VPC untuk informasi selengkapnya tentang server DNS yang disediakan Amazon atau Resolver Amazon Route 53.

Jika Anda perlu menyelesaikan nama host DNS pribadi, pastikan VPC yang diteruskan selama pembuatan konektor memiliki opsi DHCP yang diatur dengan benar. Untuk informasi selengkapnya, lihat Bekerja dengan set opsi DHCP di Panduan Pengguna Amazon VPC.

Saat Anda mengonfigurasi opsi DHCP yang ditetapkan untuk resolusi nama host DNS pribadi, pastikan konektor dapat mencapai server DNS khusus yang Anda konfigurasikan dalam set opsi DHCP. Jika tidak, pembuatan konektor Anda akan gagal.

Setelah Anda menyesuaikan set opsi VPC DHCP, konektor yang kemudian dibuat di VPC tersebut menggunakan server DNS yang Anda tentukan dalam set opsi. Jika Anda mengubah set opsi setelah Anda membuat konektor, konektor mengadopsi pengaturan dalam opsi baru yang ditetapkan dalam beberapa menit.

Atribut DNS untuk VPC Anda

Pastikan Anda memiliki atribut DNS VPC yang dikonfigurasi dengan benar seperti yang dijelaskan dalam atribut DNS di nama host VPC dan DNS Anda di Panduan Pengguna Amazon VPC.

Lihat Menyelesaikan kueri DNS antara VPC dan jaringan Anda di Panduan Pengembang Amazon Route 53 untuk informasi tentang penggunaan titik akhir resolver masuk dan keluar guna menghubungkan jaringan lain ke VPC agar berfungsi dengan konektor Anda.

Penanganan kegagalan

Bagian ini menjelaskan kemungkinan kegagalan pembuatan konektor yang terkait dengan resolusi DNS dan tindakan yang disarankan untuk menyelesaikan masalah.

Kegagalan	Tindakan yang disarankan
Pembuatan konektor gagal jika kueri resolusi DNS gagal, atau jika server DNS tidak dapat dijangkau dari konektor.	Anda dapat melihat kegagalan pembuatan konektor karena kueri resolusi DNS yang gagal di CloudWatch log Anda, jika Anda telah mengonfigurasi log ini untuk konektor Anda. Periksa konfigurasi server DNS dan pastikan konektivitas jaringan ke server DNS dari konektor.
Jika Anda mengubah konfigurasi server DNS di opsi VPC DHCP yang disetel saat konektor berjalan, kueri resolusi DNS dari konektor dapat gagal. Jika resolusi DNS gagal, beberapa tugas konektor dapat memasuki status gagal.	Anda dapat melihat kegagalan pembuatan konektor karena kueri resolusi DNS yang gagal di CloudWatch log Anda, jika Anda telah mengonfigurasi log ini untuk konektor Anda. Tugas yang gagal harus dimulai ulang secara otomatis untuk mengembalikan konektor. Jika itu tidak terjadi, Anda dapat menghubungi dukungan untuk memulai kembali tugas yang gagal untuk konektor mereka atau Anda dapat membuat ulang konektor.

Logging untuk MSK Connect

MSK Connect dapat menulis peristiwa log yang dapat Anda gunakan untuk men-debug konektor Anda. Saat Anda membuat konektor, Anda dapat menentukan nol atau lebih dari tujuan log berikut:

- Amazon CloudWatch Logs: Anda menentukan grup log yang Anda inginkan MSK Connect untuk mengirim peristiwa log konektor Anda. Untuk informasi tentang cara membuat grup log, lihat Membuat grup CloudWatch log di Panduan Pengguna Log.

- Amazon S3: Anda menentukan bucket S3 yang Anda inginkan MSK Connect untuk mengirim peristiwa log konektor Anda. Untuk informasi tentang cara membuat bucket S3, lihat Membuat bucket di Panduan Pengguna Amazon S3.
- Amazon Data Firehose: Anda menentukan aliran pengiriman yang Anda inginkan MSK Connect untuk mengirim peristiwa log konektor Anda. Untuk informasi tentang cara membuat aliran pengiriman, lihat Membuat aliran pengiriman Amazon Data Firehose di Panduan Pengguna Firehose.

Untuk mempelajari lebih lanjut tentang mengatur logging, lihat Mengaktifkan logging dari AWS layanan tertentu di Panduan Amazon CloudWatch Logs Pengguna.

MSK Connect memancarkan jenis peristiwa log berikut:

Tingkat	Deskripsi
`INFO`	Acara runtime yang menarik saat startup dan shutdown.
`WARN`	Situasi runtime yang bukan kesalahan tetapi tidak diinginkan atau tidak terduga.
`FATAL`	Kesalahan parah yang menyebabkan penghentian prematur.
`ERROR`	Kondisi tak terduga dan kesalahan runtime yang tidak fatal.

Berikut ini adalah contoh peristiwa log yang dikirim ke CloudWatch Log:

```
[Worker-0bb8afa0b01391c41] [2021-09-06 16:02:54,151] WARN [Producer
 clientId=producer-1] Connection to node 1 (b-1.my-test-cluster.twwhtj.c2.kafka.us-
east-1.amazonaws.com/INTERNAL_IP) could not be established. Broker may not be
 available. (org.apache.kafka.clients.NetworkClient:782)
```

Mencegah rahasia muncul di log konektor

> **Note**
>
> Nilai konfigurasi sensitif dapat muncul di log konektor jika plugin tidak mendefinisikan nilai-nilai tersebut sebagai rahasia. Kafka Connect memperlakukan nilai konfigurasi yang tidak ditentukan sama dengan nilai plaintext lainnya.

Jika plugin Anda mendefinisikan properti sebagai rahasia, Kafka Connect menyunting nilai properti dari log konektor. Misalnya, log konektor berikut menunjukkan bahwa jika plugin mendefinisikan `aws.secret.key` sebagai `PASSWORD` tipe, maka nilainya diganti dengan**`[hidden]`**.

```
   2022-01-11T15:18:55.000+00:00    [Worker-05e6586a48b5f331b] [2022-01-11
15:18:55,150] INFO SecretsManagerConfigProviderConfig values:
   2022-01-11T15:18:55.000+00:00    [Worker-05e6586a48b5f331b] aws.access.key =
my_access_key
   2022-01-11T15:18:55.000+00:00    [Worker-05e6586a48b5f331b] aws.region = us-east-1
   2022-01-11T15:18:55.000+00:00    [Worker-05e6586a48b5f331b] aws.secret.key
= [hidden]
   2022-01-11T15:18:55.000+00:00    [Worker-05e6586a48b5f331b] secret.prefix =
   2022-01-11T15:18:55.000+00:00    [Worker-05e6586a48b5f331b] secret.ttl.ms = 300000
   2022-01-11T15:18:55.000+00:00    [Worker-05e6586a48b5f331b]
(com.github.jcustenborder.kafka.config.aws.SecretsManagerConfigProviderConfig:361)
```

Untuk mencegah rahasia muncul di file log konektor, pengembang plugin harus menggunakan konstanta enum Kafka Connect `ConfigDef.Type.PASSWORD`untuk menentukan properti sensitif. Ketika properti adalah tipe`ConfigDef.Type.PASSWORD`, Kafka Connect mengecualikan nilainya dari log konektor bahkan jika nilai dikirim sebagai plaintext.

Pemantauan MSK Connect

Pemantauan merupakan bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja MSK Connect dan AWS solusi Anda yang lain. Amazon CloudWatch memantau AWS sumber daya Anda dan aplikasi yang Anda jalankan AWS secara real time. 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 penggunaan CPU atau metrik lain dari konektor Anda, sehingga Anda

dapat meningkatkan kapasitasnya jika diperlukan. Untuk informasi selengkapnya, lihat Panduan CloudWatch Pengguna Amazon.

Tabel berikut menunjukkan metrik yang MSK Connect kirimkan ke CloudWatch bawah dimensi. `ConnectorName` MSK Connect memberikan metrik ini secara default dan tanpa biaya tambahan. CloudWatch menyimpan metrik ini selama 15 bulan, sehingga Anda dapat mengakses informasi historis dan mendapatkan perspektif yang lebih baik tentang kinerja konektor Anda. Anda juga dapat mengatur alarm yang memperhatikan ambang batas tertentu dan mengirim notifikasi atau mengambil tindakan saat ambang batas tersebut terpenuhi. Untuk informasi selengkapnya, lihat Panduan CloudWatch Pengguna Amazon.

Metrik MSK Connect

Nama metrik	Deskripsi
`BytesInPerSec`	Jumlah total byte yang diterima oleh konektor.
`BytesOutPerSec`	Jumlah total byte yang dikirimkan oleh konektor.
`CpuUtilization`	Persentase konsumsi CPU oleh sistem dan pengguna.
`ErroredTaskCount`	Jumlah tugas yang telah terkeliru.
`MemoryUtilization`	Persentase total memori pada instance pekerja, bukan hanya memori heap Java virtual machine (JVM) yang saat ini digunakan. JVM biasanya tidak melepaskan memori kembali ke sistem operasional. Jadi, ukuran tumpukan JVM (MemoryUtilization) biasanya dimulai dengan ukuran tumpukan minimum yang secara bertahap meningkat hingga maksimum stabil sekitar 80-90%. Penggunaan heap JVM mungkin meningkat atau berkurang karena penggunaan memori aktual konektor berubah.
`RebalanceCompletedTotal`	Jumlah total rebalances yang diselesaikan oleh konektor ini.

Nama metrik	Deskripsi
RebalanceTimeAvg	Rata-rata waktu dalam milidetik yang dihabiskan oleh konektor untuk menyeimbangkan kembali.
RebalanceTimeMax	Waktu maksimum dalam milidetik yang dihabiskan oleh konektor untuk menyeimbangkan kembali.
RebalanceTimeSinceLast	Waktu dalam milidetik sejak konektor ini menyelesaikan penyeimbangan ulang terbaru.
RunningTaskCount	Jumlah tugas yang berjalan di konektor.
SinkRecordReadRate	Rata-rata jumlah catatan per detik dibaca dari kluster Apache Kafka atau Amazon MSK.
SinkRecordSendRate	Rata-rata jumlah catatan per detik yang dihasilkan dari transformasi dan dikirim ke tujuan. Nomor ini tidak termasuk catatan yang difilter.
SourceRecordPollRate	Jumlah rata-rata per detik catatan yang diproduksi atau disurvei.
SourceRecordWriteRate	Rata-rata jumlah rekaman per detik output dari transformasi dan ditulis ke Apache Kafka atau Amazon MSK cluster.
TaskStartupAttemptsTotal	Jumlah total startup tugas yang telah dicoba konektor. Anda dapat menggunakan metrik ini untuk mengidentifikasi anomali dalam upaya memulai tugas.
TaskStartupSuccessPercentage	Persentase rata-rata tugas yang berhasil dimulai untuk konektor. Anda dapat menggunakan metrik ini untuk mengidentifikasi anomali dalam upaya memulai tugas.

Nama metrik	Deskripsi
`WorkerCount`	Jumlah pekerja yang berjalan di konektor.

Contoh

Bagian ini mencakup contoh untuk membantu Anda menyiapkan sumber daya Amazon MSK Connect seperti konektor pihak ketiga umum dan penyedia konfigurasi.

Topik

- Konektor wastafel Amazon S3
- Konektor sumber Debezium dengan penyedia konfigurasi

Konektor wastafel Amazon S3

Contoh ini menunjukkan cara menggunakan konektor wastafel Amazon S3 Confluent dan untuk membuat konektor wastafel Amazon S3 AWS CLI di MSK Connect.

1. Salin JSON berikut dan tempel di file baru. Ganti string placeholder dengan nilai yang sesuai dengan string koneksi server bootstrap Amazon MSK cluster Anda dan subnet dan ID grup keamanan klaster. Untuk informasi tentang cara menyiapkan peran eksekusi layanan, lihatthe section called “Peran dan kebijakan IAM”.

```
{
    "connectorConfiguration": {
        "connector.class": "io.confluent.connect.s3.S3SinkConnector",
        "s3.region": "us-east-1",
        "format.class": "io.confluent.connect.s3.format.json.JsonFormat",
        "flush.size": "1",
        "schema.compatibility": "NONE",
        "topics": "my-test-topic",
        "tasks.max": "2",
        "partitioner.class":
 "io.confluent.connect.storage.partitioner.DefaultPartitioner",
        "storage.class": "io.confluent.connect.s3.storage.S3Storage",
        "s3.bucket.name": "my-test-bucket"
    },
    "connectorName": "example-S3-sink-connector",
```

```
    "kafkaCluster": {
        "apacheKafkaCluster": {
            "bootstrapServers": "<cluster-bootstrap-servers-string>",
            "vpc": {
                "subnets": [
                    "<cluster-subnet-1>",
                    "<cluster-subnet-2>",
                    "<cluster-subnet-3>"
                ],
                "securityGroups": ["<cluster-security-group-id>"]
            }
        }
    },
    "capacity": {
        "provisionedCapacity": {
            "mcuCount": 2,
            "workerCount": 4
        }
    },
    "kafkaConnectVersion": "2.7.1",
    "serviceExecutionRoleArn": "<arn-of-a-role-that-msk-connect-can-assume>",
    "plugins": [
        {
            "customPlugin": {
                "customPluginArn": "<arn-of-custom-plugin-that-contains-connector-
code>",
                "revision": 1
            }
        }
    ],
    "kafkaClusterEncryptionInTransit": {"encryptionType": "PLAINTEXT"},
    "kafkaClusterClientAuthentication": {"authenticationType": "NONE"}
}
```

2. Jalankan AWS CLI perintah berikut di folder tempat Anda menyimpan file JSON di langkah sebelumnya.

```
aws kafkaconnect create-connector --cli-input-json file://connector-info.json
```

Berikut ini adalah contoh output yang Anda dapatkan ketika Anda menjalankan perintah dengan sukses.

```
{
    "ConnectorArn": "arn:aws:kafkaconnect:us-east-1:123450006789:connector/example-
S3-sink-connector/abc12345-abcd-4444-a8b9-123456f513ed-2",
    "ConnectorState": "CREATING",
    "ConnectorName": "example-S3-sink-connector"
}
```

Konektor sumber Debezium dengan penyedia konfigurasi

Contoh ini menunjukkan cara menggunakan plugin konektor Debezium MySQL dengan database Amazon Aurora yang kompatibel dengan MySQL sebagai sumbernya. Dalam contoh ini, kami juga menyiapkan Penyedia Config AWS Secrets Manager open-source untuk mengeksternalisasi kredenal database di. AWS Secrets Manager Untuk mempelajari lebih lanjut tentang penyedia konfigurasi, lihatEksternalisasi informasi sensitif menggunakan penyedia konfigurasi.

> **Important**
>
> Plugin konektor Debezium MySQL hanya mendukung satu tugas dan tidak berfungsi dengan mode kapasitas berskala otomatis untuk Amazon MSK Connect. Sebagai gantinya, Anda harus menggunakan mode kapasitas yang disediakan dan menyetel `workerCount` sama dengan satu di konfigurasi konektor Anda. Untuk mempelajari lebih lanjut tentang mode kapasitas MSK Connect, lihatKapasitas konektor.

Sebelum Anda mulai

Konektor Anda harus dapat mengakses internet sehingga dapat berinteraksi dengan layanan seperti AWS Secrets Manager yang berada di luar Anda Amazon Virtual Private Cloud. Langkah-langkah di bagian ini membantu Anda menyelesaikan tugas-tugas berikut untuk mengaktifkan akses internet.

- Siapkan subnet publik yang menghosting gateway NAT dan merutekan lalu lintas ke gateway internet di VPC Anda.
- Buat rute default yang mengarahkan lalu lintas subnet pribadi Anda ke gateway NAT Anda.

Untuk informasi selengkapnya, lihat Mengaktifkan akses internet untuk Amazon MSK Connect.

Prasyarat

Sebelum Anda dapat mengaktifkan akses internet, Anda memerlukan item berikut:

- ID dari Amazon Virtual Private Cloud (VPC) yang terkait dengan cluster Anda. Misalnya, vpc-123456ab.
- ID subnet pribadi di VPC Anda. Misalnya, subnet-a1b2c3de, subnet-f4g5h6ij, dll. Anda harus mengkonfigurasi konektor Anda dengan subnet pribadi.

Untuk mengaktifkan akses internet untuk konektor Anda

1. Buka Amazon Virtual Private Cloud konsol di https://console.aws.amazon.com/vpc/.
2. Buat subnet publik untuk gateway NAT Anda dengan nama deskriptif, dan catat ID subnet. Untuk petunjuk terperinci, lihat Membuat subnet di VPC Anda.
3. Buat gateway internet sehingga VPC Anda dapat berkomunikasi dengan internet, dan catat ID gateway. Lampirkan gateway internet ke VPC Anda. Untuk petunjuk, lihat Membuat dan melampirkan gateway internet.
4. Menyediakan gateway NAT publik sehingga host di subnet pribadi Anda dapat mencapai subnet publik Anda. Saat Anda membuat gateway NAT, pilih subnet publik yang Anda buat sebelumnya. Untuk petunjuk, silakan lihat Membuat gateway NAT.
5. Konfigurasikan tabel rute Anda. Anda harus memiliki dua tabel rute secara total untuk menyelesaikan pengaturan ini. Anda seharusnya sudah memiliki tabel rute utama yang dibuat secara otomatis bersamaan dengan VPC Anda. Pada langkah ini Anda membuat tabel rute tambahan untuk subnet publik Anda.

   a. Gunakan pengaturan berikut untuk memodifikasi tabel rute utama VPC Anda sehingga subnet pribadi Anda merutekan lalu lintas ke gateway NAT Anda. Untuk petunjuk, lihat Bekerja dengan tabel rute di Panduan Amazon Virtual Private CloudPengguna.

      Tabel rute MSKC pribadi

Properti	Nilai
Tag nama	Kami menyarankan Anda memberikan tabel rute ini tag nama deskriptif untuk membantu Anda mengidentifikasinya. Misalnya, MSKC Pribadi.
Subnet terkait	Subnet pribadi Anda

Properti	Nilai
Rute untuk mengaktifkan akses internet untuk MSK Connect	• Tujuan: 0.0.0.0/0 • Target: ID gateway NAT Anda. Misalnya, nat-12a345bc6789efg1h.
Rute lokal untuk lalu lintas internal	• Tujuan: 10.0.0.0/16. Nilai ini mungkin berbeda tergantung pada blok CIDR VPC Anda. • Target: Lokal

b. Ikuti petunjuk di Buat tabel rute kustom untuk membuat tabel rute untuk subnet publik Anda. Saat Anda membuat tabel, masukkan nama deskriptif di kolom Tag nama untuk membantu Anda mengidentifikasi subnet mana yang terkait dengan tabel tersebut. Misalnya, MSKC Publik.

c. Konfigurasikan tabel rute MSKC Publik Anda menggunakan pengaturan berikut.

Properti	Nilai
Tag nama	MSKC publik atau nama deskriptif lain yang Anda pilih
Subnet terkait	Subnet publik Anda dengan gateway NAT
Rute untuk mengaktifkan akses internet untuk MSK Connect	• Tujuan: 0.0.0.0/0 • Target: ID gateway internet Anda. Misalnya, igw-1a234bc5.
Rute lokal untuk lalu lintas internal	• Tujuan: 10.0.0.0/16. Nilai ini mungkin berbeda tergantung pada blok CIDR VPC Anda. • Target: Lokal

Sekarang setelah Anda mengaktifkan akses internet untuk Amazon MSK Connect, Anda siap membuat konektor.

Membuat konektor sumber Debezium

1. Buat plugin khusus

   a. Unduh plugin konektor MySQL untuk rilis stabil terbaru dari situs Debezium. Catat versi rilis Debezium yang Anda unduh (versi 2.x, atau seri 1.x yang lebih lama). Kemudian dalam prosedur ini, Anda akan membuat konektor berdasarkan versi Debezium Anda.

   b. Unduh dan ekstrak Penyedia Config AWS Secrets Manager.

   c. Tempatkan arsip berikut ke dalam direktori yang sama:

   - `debezium-connector-mysql`Folder
   - `jcusten-border-kafka-config-provider-aws-0.1.1`Folder

   d. Kompres direktori yang Anda buat pada langkah sebelumnya ke dalam file ZIP dan kemudian unggah file ZIP ke bucket S3. Untuk petunjuk, lihat Mengunggah objek di Panduan Pengguna Amazon S3.

   e. Salin JSON berikut dan tempel dalam file. Misalnya, `debezium-source-custom-plugin.json`. Ganti *< example-custom-plugin-name >* dengan nama yang ingin dimiliki plugin, *< arn-of-your-s 3-bucket> dengan* ARN bucket S3 tempat Anda mengunggah file ZIP, dan *<file-key-of-ZIP-object>* dengan kunci file objek ZIP yang Anda unggah ke S3.

```
{
    "name": "<example-custom-plugin-name>",
    "contentType": "ZIP",
    "location": {
        "s3Location": {
            "bucketArn": "<arn-of-your-s3-bucket>",
            "fileKey": "<file-key-of-ZIP-object>"
        }
    }
}
```

   f. Jalankan AWS CLI perintah berikut dari folder tempat Anda menyimpan file JSON untuk membuat plugin.

```
aws kafkaconnect create-custom-plugin --cli-input-json file://<debezium-source-custom-plugin.json>
```

Anda akan melihat output yang mirip dengan contoh berikut.

```
{
    "CustomPluginArn": "arn:aws:kafkaconnect:us-east-1:012345678901:custom-
plugin/example-custom-plugin-name/abcd1234-a0b0-1234-c1-12345678abcd-1",
    "CustomPluginState": "CREATING",
    "Name": "example-custom-plugin-name",
    "Revision": 1
}
```

g. Jalankan perintah berikut untuk memeriksa status plugin. Negara harus berubah dari `CREATING` ke`ACTIVE`. Ganti placeholder ARN dengan ARN yang Anda dapatkan di output dari perintah sebelumnya.

```
aws kafkaconnect describe-custom-plugin --custom-plugin-arn "<arn-of-your-
custom-plugin>"
```

2. Konfigurasikan AWS Secrets Manager dan buat rahasia untuk kredensi database Anda

a. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.

b. Buat rahasia baru untuk menyimpan kredensi login database Anda. Untuk petunjuk, lihat Membuat rahasia di Panduan AWS Secrets ManagerPengguna.

c. Salin ARN rahasia Anda.

d. Tambahkan izin Secrets Manager dari kebijakan contoh berikut ke kebijakan AndaPeran eksekusi layanan. Ganti *<arn:aws:secretsmanager:us-east-1:123456789000:secret : -1234>*dengan ARN rahasia Anda. MySecret

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetResourcePolicy",
        "secretsmanager:GetSecretValue",
        "secretsmanager:DescribeSecret",
        "secretsmanager:ListSecretVersionIds"
      ],
      "Resource": [
        "<arn:aws:secretsmanager:us-east-1:123456789000:secret:MySecret-1234>"
```

```
      ]
    }
  ]
}
```

Untuk petunjuk tentang cara menambahkan izin IAM, lihat Menambahkan dan menghapus izin identitas IAM di Panduan Pengguna IAM.

3. Buat konfigurasi pekerja khusus dengan informasi tentang penyedia konfigurasi Anda

   a. Salin properti konfigurasi pekerja berikut ke dalam file, ganti string placeholder dengan nilai yang sesuai dengan skenario Anda. Untuk mempelajari lebih lanjut tentang properti konfigurasi untuk Penyedia Config AWS Secrets Manager, lihat SecretsManagerConfigProviderdi dokumentasi plugin.

```
key.converter=<org.apache.kafka.connect.storage.StringConverter>
value.converter=<org.apache.kafka.connect.storage.StringConverter>
config.providers.secretManager.class=com.github.jcustenborder.kafka.config.aws.SecretsM
config.providers=secretManager
config.providers.secretManager.param.aws.region=<us-east-1>
```

   b. Jalankan AWS CLI perintah berikut untuk membuat konfigurasi pekerja kustom Anda.

      Ganti nilai-nilai berikut:

      - *< my-worker-config-name >* - nama deskriptif untuk konfigurasi pekerja kustom Anda
      - *< encoded-properties-file-content -string>* - versi berkode base64 dari properti plaintext yang Anda salin pada langkah sebelumnya

```
aws kafkaconnect create-worker-configuration --name <my-worker-config-name> --
properties-file-content <encoded-properties-file-content-string>
```

4. Buat konektor

   a. Salin JSON berikut yang sesuai dengan versi Debezium Anda (2.x atau 1.x) dan tempel di file baru. Ganti *<placeholder>* string dengan nilai yang sesuai dengan skenario Anda. Untuk informasi tentang cara menyiapkan peran eksekusi layanan, lihatthe section called “Peran dan kebijakan IAM”.

Perhatikan bahwa konfigurasi menggunakan variabel seperti `${secretManager:MySecret-1234:dbusername}` bukan plaintext untuk menentukan kredensyal database. Ganti *MySecret-1234* dengan nama rahasia Anda dan kemudian sertakan nama kunci yang ingin Anda ambil. Anda juga harus mengganti *<arn-of-config-provider-worker-configuration>* dengan ARN konfigurasi pekerja kustom Anda.

Debezium 2.x

Untuk versi Debezium 2.x, salin JSON berikut dan tempel di file baru. Ganti <placeholder>string dengan nilai yang sesuai dengan skenario Anda.

```
{
 "connectorConfiguration": {
  "connector.class": "io.debezium.connector.mysql.MySqlConnector",
  "tasks.max": "1",
  "database.hostname": "<aurora-database-writer-instance-endpoint>",
  "database.port": "3306",
  "database.user": "<${secretManager:MySecret-1234:dbusername}>",
  "database.password": "<${secretManager:MySecret-1234:dbpassword}>",
  "database.server.id": "123456",
  "database.include.list": "<list-of-databases-hosted-by-specified-server>",
  "topic.prefix": "<logical-name-of-database-server>",
  "schema.history.internal.kafka.topic": "<kafka-topic-used-by-debezium-to-track-schema-changes>",
  "schema.history.internal.kafka.bootstrap.servers": "<cluster-bootstrap-servers-string>",
  "schema.history.internal.consumer.security.protocol": "SASL_SSL",
  "schema.history.internal.consumer.sasl.mechanism": "AWS_MSK_IAM",
  "schema.history.internal.consumer.sasl.jaas.config":
 "software.amazon.msk.auth.iam.IAMLoginModule required;",
  "schema.history.internal.consumer.sasl.client.callback.handler.class":
 "software.amazon.msk.auth.iam.IAMClientCallbackHandler",
  "schema.history.internal.producer.security.protocol": "SASL_SSL",
  "schema.history.internal.producer.sasl.mechanism": "AWS_MSK_IAM",
  "schema.history.internal.producer.sasl.jaas.config":
 "software.amazon.msk.auth.iam.IAMLoginModule required;",
  "schema.history.internal.producer.sasl.client.callback.handler.class":
 "software.amazon.msk.auth.iam.IAMClientCallbackHandler",
  "include.schema.changes": "true"
 },
```

```
 "connectorName": "example-Debezium-source-connector",
 "kafkaCluster": {
  "apacheKafkaCluster": {
   "bootstrapServers": "<cluster-bootstrap-servers-string>",
   "vpc": {
    "subnets": [
     "<cluster-subnet-1>",
     "<cluster-subnet-2>",
     "<cluster-subnet-3>"
    ],
    "securityGroups": ["<id-of-cluster-security-group>"]
   }
  }
 },
 "capacity": {
  "provisionedCapacity": {
   "mcuCount": 2,
   "workerCount": 1
  }
 },
 "kafkaConnectVersion": "2.7.1",
 "serviceExecutionRoleArn": "<arn-of-service-execution-role-that-msk-
connect-can-assume>",
 "plugins": [{
  "customPlugin": {
   "customPluginArn": "<arn-of-msk-connect-plugin-that-contains-connector-
code>",
   "revision": 1
  }
 }],
 "kafkaClusterEncryptionInTransit": {
  "encryptionType": "TLS"
 },
 "kafkaClusterClientAuthentication": {
  "authenticationType": "IAM"
 },
 "workerConfiguration": {
  "workerConfigurationArn": "<arn-of-config-provider-worker-configuration>",
  "revision": 1
 }
}
```

Debezium 1.x

Untuk versi Debezium 1.x, salin JSON berikut dan tempel di file baru. Ganti <placeholder>string dengan nilai yang sesuai dengan skenario Anda.

```
{
 "connectorConfiguration": {
  "connector.class": "io.debezium.connector.mysql.MySqlConnector",
  "tasks.max": "1",
  "database.hostname": "<aurora-database-writer-instance-endpoint>",
  "database.port": "3306",
  "database.user": "<${secretManager:MySecret-1234:dbusername}>",
  "database.password": "<${secretManager:MySecret-1234:dbpassword}>",
  "database.server.id": "123456",
  "database.server.name": "<logical-name-of-database-server>",
  "database.include.list": "<list-of-databases-hosted-by-specified-server>",
  "database.history.kafka.topic": "<kafka-topic-used-by-debezium-to-track-
schema-changes>",
  "database.history.kafka.bootstrap.servers": "<cluster-bootstrap-servers-
string>",
  "database.history.consumer.security.protocol": "SASL_SSL",
  "database.history.consumer.sasl.mechanism": "AWS_MSK_IAM",
  "database.history.consumer.sasl.jaas.config":
 "software.amazon.msk.auth.iam.IAMLoginModule required;",
  "database.history.consumer.sasl.client.callback.handler.class":
 "software.amazon.msk.auth.iam.IAMClientCallbackHandler",
  "database.history.producer.security.protocol": "SASL_SSL",
  "database.history.producer.sasl.mechanism": "AWS_MSK_IAM",
  "database.history.producer.sasl.jaas.config":
 "software.amazon.msk.auth.iam.IAMLoginModule required;",
  "database.history.producer.sasl.client.callback.handler.class":
 "software.amazon.msk.auth.iam.IAMClientCallbackHandler",
  "include.schema.changes": "true"
 },
 "connectorName": "example-Debezium-source-connector",
 "kafkaCluster": {
  "apacheKafkaCluster": {
   "bootstrapServers": "<cluster-bootstrap-servers-string>",
   "vpc": {
    "subnets": [
     "<cluster-subnet-1>",
     "<cluster-subnet-2>",
     "<cluster-subnet-3>"
```

```
    ],
    "securityGroups": ["<id-of-cluster-security-group>"]
   }
  }
 },
 "capacity": {
  "provisionedCapacity": {
   "mcuCount": 2,
   "workerCount": 1
  }
 },
 "kafkaConnectVersion": "2.7.1",
 "serviceExecutionRoleArn": "<arn-of-service-execution-role-that-msk-
connect-can-assume>",
 "plugins": [{
  "customPlugin": {
   "customPluginArn": "<arn-of-msk-connect-plugin-that-contains-connector-
code>",
   "revision": 1
  }
 }],
 "kafkaClusterEncryptionInTransit": {
  "encryptionType": "TLS"
 },
 "kafkaClusterClientAuthentication": {
  "authenticationType": "IAM"
 },
 "workerConfiguration": {
  "workerConfigurationArn": "<arn-of-config-provider-worker-configuration>",
  "revision": 1
 }
}
```

b. Jalankan AWS CLI perintah berikut di folder tempat Anda menyimpan file JSON di langkah sebelumnya.

```
aws kafkaconnect create-connector --cli-input-json file://connector-info.json
```

Berikut ini adalah contoh output yang Anda dapatkan ketika Anda menjalankan perintah dengan sukses.

```
{
```

```
    "ConnectorArn": "arn:aws:kafkaconnect:us-east-1:123450006789:connector/
example-Debezium-source-connector/abc12345-abcd-4444-a8b9-123456f513ed-2",
    "ConnectorState": "CREATING",
    "ConnectorName": "example-Debezium-source-connector"
}
```

Untuk contoh konektor Debezium dengan langkah-langkah terperinci, lihat Memperkenalkan Amazon MSK Connect - Streaming Data ke dan dari Cluster Apache Kafka Anda Menggunakan Konektor Terkelola.

Praktik terbaik

Gunakan ini sebagai referensi untuk menemukan rekomendasi dengan cepat untuk memaksimalkan kinerja dengan Amazon MSK Connect.

Topik

- Menghubungkan dari konektor

Menghubungkan dari konektor

Praktik terbaik berikut dapat meningkatkan kinerja konektivitas Anda ke Amazon MSK Connect.

Jangan tumpang tindih IP untuk peering Amazon VPC atau Transit Gateway

Jika Anda menggunakan peering VPC Amazon atau Transit Gateway dengan Amazon MSK Connect, jangan konfigurasikan konektor Anda untuk menjangkau sumber daya VPC peered dengan IP dalam rentang CIDR:

- “10.99.0.0/16"
- “192.168.0.0/16"
- “172.21.0.0/16"

Panduan migrasi Amazon MSK Connect

Bagian ini menjelaskan cara memigrasikan aplikasi konektor Apache Kafka Anda ke Amazon Managed Streaming for Apache Kafka Connect (Amazon MSK Connect).

Topik

Manfaat menggunakan Amazon MSK Connect

Apache Kafka adalah salah satu platform streaming open source yang paling banyak diadopsi untuk menelan dan memproses aliran data real-time. Dengan Apache Kafka, Anda dapat memisahkan dan secara mandiri menskalakan aplikasi penghasil data dan penggunaan data Anda.

Kafka Connect adalah komponen penting dalam membangun dan menjalankan aplikasi streaming dengan Apache Kafka. Kafka Connect menyediakan cara standar untuk memindahkan data antara Kafka dan sistem eksternal. Kafka Connect sangat skalabel dan dapat menangani volume besar data Kafka Connect menyediakan serangkaian operasi dan alat API yang kuat untuk mengonfigurasi, menyebarkan, dan memantau konektor yang memindahkan data antara topik Kafka dan sistem eksternal. Anda dapat menggunakan alat ini untuk menyesuaikan dan memperluas fungsionalitas Kafka Connect untuk memenuhi kebutuhan spesifik aplikasi streaming Anda.

Anda mungkin menghadapi tantangan saat mengoperasikan kluster Apache Kafka Connect sendiri, atau ketika Anda mencoba memigrasi aplikasi Apache Kafka Connect open source ke. AWS Tantangan ini termasuk waktu yang diperlukan untuk menyiapkan infrastruktur dan penerapan aplikasi, hambatan rekayasa saat menyiapkan cluster Apache Kafka Connect yang dikelola sendiri, dan overhead operasional administratif.

Untuk mengatasi tantangan ini, sebaiknya gunakan Amazon Managed Streaming for Apache Kafka Connect (Amazon MSK Connect) untuk memigrasikan aplikasi Apache Kafka Connect open source Anda ke. AWS Amazon MSK Connect menyederhanakan penggunaan Kafka Connect untuk mengalirkan data ke dan dari antara cluster Apache Kafka dan sistem eksternal, seperti database, indeks pencarian, dan sistem file.

Berikut adalah beberapa manfaat untuk bermigrasi ke Amazon MSK Connect:

- Penghapusan overhead operasional - Amazon MSK Connect menghilangkan beban operasional yang terkait dengan penambalan, penyediaan, dan penskalaan cluster Apache Kafka Connect. Amazon MSK Connect terus memantau kesehatan cluster Connect Anda dan mengotomatiskan patching dan upgrade versi tanpa menyebabkan gangguan pada beban kerja Anda.
- Memulai ulang tugas Connect secara otomatis — Amazon MSK Connect dapat memulihkan tugas yang gagal secara otomatis untuk mengurangi gangguan produksi. Kegagalan tugas dapat

disebabkan oleh kesalahan sementara, seperti melanggar batas koneksi TCP untuk Kafka, dan penyeimbangan kembali tugas ketika pekerja baru bergabung dengan grup konsumen untuk konektor wastafel.

- Penskalaan horizontal dan vertikal otomatis — Amazon MSK Connect memungkinkan aplikasi konektor untuk menskalakan secara otomatis untuk mendukung throughput yang lebih tinggi. Amazon MSK Connect mengelola penskalaan untuk Anda. Anda hanya perlu menentukan jumlah pekerja dalam grup penskalaan otomatis dan ambang batas pemanfaatan. Anda dapat menggunakan operasi Amazon MSK Connect `UpdateConnector` API untuk meningkatkan atau menurunkan skala vCPU secara vertikal antara 1 dan 8 vCPU untuk mendukung throughput variabel.
- Konektivitas jaringan pribadi — Amazon MSK Connect terhubung secara pribadi ke sistem sumber dan sink dengan menggunakan AWS PrivateLink dan nama DNS pribadi.

Migrasi ke Amazon MSK Connect

Bagian ini menjelaskan secara singkat topik manajemen negara yang digunakan oleh Kafka Connect dan Amazon MSK Connect. Bagian ini juga mencakup prosedur untuk memigrasikan konektor sumber dan wastafel.

Topik

Topik internal yang digunakan oleh Kafka Connect

Aplikasi Apache Kafka Connect yang berjalan dalam mode terdistribusi menyimpan statusnya dengan menggunakan topik internal di cluster Kafka dan keanggotaan grup. Berikut ini adalah nilai konfigurasi yang sesuai dengan topik internal yang digunakan untuk aplikasi Kafka Connect:

- Topik konfigurasi, ditentukan melalui `config.storage.topic`

  Dalam topik konfigurasi, Kafka Connect menyimpan konfigurasi semua konektor dan tugas yang telah dimulai oleh pengguna. Setiap kali pengguna memperbarui konfigurasi konektor atau ketika konektor meminta konfigurasi ulang (misalnya, konektor mendeteksi bahwa ia dapat memulai lebih

banyak tugas), catatan dipancarkan ke topik ini. Topik ini diaktifkan pemadatan, sehingga selalu menyimpan status terakhir untuk setiap entitas.

- Topik offset, ditentukan melalui `offset.storage.topic`

  Dalam topik offset, Kafka Connect menyimpan offset konektor sumber. Seperti topik konfigurasi, topik offset diaktifkan pemadatan. Topik ini digunakan untuk menulis posisi sumber hanya untuk konektor sumber yang menghasilkan data ke Kafka dari sistem eksternal. Konektor sink, yang membaca data dari Kafka dan mengirim ke sistem eksternal, menyimpan offset konsumen mereka dengan menggunakan kelompok konsumen Kafka biasa.

- Topik status, ditentukan melalui `status.storage.topic`

  Dalam topik status, Kafka Connect menyimpan kondisi konektor dan tugas saat ini. Topik ini digunakan sebagai tempat sentral untuk data yang ditanyakan oleh pengguna REST API. Topik ini memungkinkan pengguna untuk menanyakan pekerja mana pun dan masih mendapatkan status semua plugin yang sedang berjalan. Seperti topik konfigurasi dan offset, topik status juga diaktifkan pemadatan.

Selain topik-topik ini, Kafka Connect memanfaatkan API keanggotaan grup Kafka secara ekstensif. Grup diberi nama setelah nama konektor. Misalnya, untuk konektor bernama file-sink, grup diberi nama. connect-file-sink Setiap konsumen dalam grup memberikan catatan untuk satu tugas. Kelompok-kelompok ini dan offsetnya dapat diambil dengan menggunakan alat kelompok konsumen biasa, seperti. `Kafka-consumer-group.sh` Untuk setiap konektor sink, runtime Connect menjalankan grup konsumen reguler yang mengekstrak catatan dari Kafka.

Manajemen negara bagian aplikasi Amazon MSK Connect

Secara default, Amazon MSK Connect membuat tiga topik terpisah di cluster Kafka untuk setiap Konektor MSK Amazon untuk menyimpan konfigurasi, offset, dan status konektor. Nama topik default disusun sebagai berikut:

- *__msk_connect_configs_ konektor-nama _ konektor-id*
- *__msk_connect_status_ konektor-nama _ konektor-id*
- *__msk_connect_offsets_ konektor-nama _ konektor-id*

> **Note**
>
> Untuk memberikan kontinuitas offset antara konektor sumber, Anda dapat menggunakan topik penyimpanan offset pilihan Anda, bukan topik default. Menentukan topik penyimpanan offset membantu Anda menyelesaikan tugas seperti membuat konektor sumber yang melanjutkan pembacaan dari offset terakhir konektor sebelumnya. Untuk menentukan topik penyimpanan offset, berikan nilai untuk offset.storage.topicproperti dalam konfigurasi pekerja Amazon MSK Connect sebelum membuat konektor.

Migrasi konektor sumber ke Amazon MSK Connect

Konektor sumber adalah aplikasi Apache Kafka Connect yang mengimpor catatan dari sistem eksternal ke Kafka. Bagian ini menjelaskan proses migrasi aplikasi konektor sumber Apache Kafka Connect yang menjalankan kluster Kafka Connect lokal atau dikelola sendiri yang berjalan ke Amazon MSK Connect. AWS

Aplikasi konektor sumber Kafka Connect menyimpan offset dalam topik yang diberi nama dengan nilai yang ditetapkan untuk properti config. `offset.storage.topic` Berikut ini adalah contoh pesan offset untuk konektor JDBC yang menjalankan dua tugas yang mengimpor data dari dua tabel berbeda bernama dan. `movies shows` Baris terbaru yang diimpor dari film tabel memiliki ID utama18343. Baris terbaru yang diimpor dari tabel show memiliki ID utama732.

```
["jdbcsource",{"protocol":"1","table":"sample.movies"}] {"incrementing":18343}
["jdbcsource",{"protocol":"1","table":"sample.shows"}] {"incrementing":732}
```

Untuk memigrasikan konektor sumber ke Amazon MSK Connect, lakukan hal berikut:

1. Buat plugin khusus Amazon MSK Connect dengan menarik pustaka konektor dari kluster Kafka Connect lokal atau yang dikelola sendiri.
2. Buat properti pekerja Amazon MSK Connect dan atur properti `key.convertervalue.converter`, dan `offset.storage.topic` ke nilai yang sama yang ditetapkan untuk konektor Kafka yang berjalan di cluster Kafka Connect yang ada.
3. Jeda aplikasi konektor pada cluster yang ada dengan membuat `PUT /connectors/`*`connector-name`*`/pause` permintaan pada cluster Kafka Connect yang ada.
4. Pastikan bahwa semua tugas aplikasi konektor benar-benar dihentikan. Anda dapat menghentikan tugas baik dengan membuat `GET /connectors/`*`connector-name`*`/status` permintaan

pada klaster Kafka Connect yang ada atau dengan menggunakan pesan dari nama topik yang ditetapkan untuk propertistatus.storage.topic.

5. Dapatkan konfigurasi konektor dari cluster yang ada. Anda bisa mendapatkan konfigurasi konektor baik dengan membuat `GET /connectors/connector-name/config/` permintaan pada klaster yang ada atau dengan menggunakan pesan dari nama topik yang ditetapkan untuk properticonfig.storage.topic.
6. Buat Konektor MSK Amazon baru dengan nama yang sama dengan cluster yang ada. Buat konektor ini dengan menggunakan plugin kustom konektor yang Anda buat di langkah 1, properti pekerja yang Anda buat di langkah 2, dan konfigurasi konektor yang Anda ekstrak pada langkah 5.
7. Saat status Konektor MSK Amazonactive, lihat log untuk memverifikasi bahwa konektor telah mulai mengimpor data dari sistem sumber.
8. Hapus konektor di cluster yang ada dengan membuat `DELETE /connectors/connector-name` permintaan.

Migrasi konektor wastafel ke Amazon MSK Connect

Konektor sink adalah aplikasi Apache Kafka Connect yang mengekspor data dari Kafka ke sistem eksternal. Bagian ini menjelaskan proses migrasi aplikasi konektor sink Apache Kafka Connect yang menjalankan kluster Kafka Connect lokal atau dikelola sendiri yang berjalan ke Amazon MSK Connect. AWS

Konektor sink Kafka Connect menggunakan API keanggotaan grup Kafka dan menyimpan offset dalam `__consumer_offset` topik yang sama dengan aplikasi konsumen biasa. Perilaku ini menyederhanakan migrasi konektor sink dari cluster yang dikelola sendiri ke Amazon MSK Connect.

Untuk memigrasikan konektor sink ke Amazon MSK Connect, lakukan hal berikut:

1. Buat plugin khusus Amazon MSK Connect dengan menarik pustaka konektor dari kluster Kafka Connect lokal atau yang dikelola sendiri.
2. Buat properti pekerja Amazon MSK Connect dan atur properti `key.converter` dan `value.converter` ke nilai yang sama yang ditetapkan untuk konektor Kafka yang berjalan di cluster Kafka Connect yang ada.
3. Jeda aplikasi konektor pada cluster Anda yang ada dengan membuat `PUT /connectors/connector-name/pause` permintaan pada cluster Kafka Connect yang ada.
4. Pastikan bahwa semua tugas aplikasi konektor benar-benar dihentikan. Anda dapat menghentikan tugas baik dengan membuat `GET /connectors/connector-name/status` permintaan

pada klaster Kafka Connect yang ada, atau dengan menggunakan pesan dari nama topik yang ditetapkan untuk propertistatus.storage.topic.

5. Dapatkan konfigurasi konektor dari cluster yang ada. Anda bisa mendapatkan konfigurasi konektor baik dengan membuat `GET /connectors/connector-name/config` permintaan pada klaster yang ada, atau dengan menggunakan pesan dari nama topik yang ditetapkan untuk properticonfig.storage.topic.
6. Buat Konektor MSK Amazon baru dengan nama yang sama dengan cluster yang ada. Buat konektor ini dengan menggunakan plugin kustom konektor yang Anda buat di langkah 1, properti pekerja yang Anda buat di langkah 2, dan konfigurasi konektor yang Anda ekstrak pada langkah 5.
7. Saat status Konektor MSK Amazonactive, lihat log untuk memverifikasi bahwa konektor telah mulai mengimpor data dari sistem sumber.
8. Hapus konektor di cluster yang ada dengan membuat `DELETE /connectors/connector-name` permintaan.

Memecahkan Masalah Amazon MSK Connect

Informasi berikut dapat membantu Anda memecahkan masalah yang mungkin Anda alami saat menggunakan MSK Connect. Anda juga dapat memposting masalah Anda ke AWS re:Post.

Konektor tidak dapat mengakses sumber daya yang dihosting di internet publik

Lihat Mengaktifkan akses internet untuk Amazon MSK Connect.

Jumlah tugas yang berjalan konektor tidak sama dengan jumlah tugas yang ditentukan dalam tasks.max

Berikut adalah beberapa alasan konektor mungkin menggunakan lebih sedikit tugas daripada konfigurasi tasks.max yang ditentukan:

- Beberapa implementasi konektor membatasi jumlah tugas yang dapat digunakan. Misalnya, konektor Debezium untuk MySQL terbatas untuk menggunakan satu tugas.
- Saat menggunakan mode kapasitas skala otomatis, Amazon MSK Connect mengganti properti tasks.max konektor dengan nilai yang sebanding dengan jumlah pekerja yang berjalan di konektor dan jumlah MCU per pekerja.
- Untuk konektor wastafel, tingkat paralelisme (jumlah tugas) tidak boleh lebih dari jumlah partisi topik. Meskipun Anda dapat mengatur tasks.max lebih besar dari itu, satu partisi tidak pernah diproses oleh lebih dari satu tugas pada satu waktu.

- Di Kafka Connect 2.7.x, penetapan partisi konsumen default adalah. `RangeAssignor` Perilaku pemberi tugas ini adalah memberikan partisi pertama dari setiap topik kepada satu konsumen, partisi kedua dari setiap topik kepada satu konsumen, dll. Ini berarti bahwa jumlah maksimum tugas aktif untuk konektor wastafel menggunakan `RangeAssignor` sama dengan jumlah maksimum partisi dalam setiap topik yang dikonsumsi. Jika ini tidak berfungsi untuk kasus penggunaan Anda, Anda harus membuat Konfigurasi Pekerja di mana `consumer.partition.assignment.strategy` properti disetel ke penugasan partisi konsumen yang lebih sesuai. Lihat Antarmuka Kafka 2.7 ConsumerPartitionAssignor: Semua Kelas Penerapan yang Dikenal.

Replikator MSK

Apa itu Amazon MSK Replicator?

Amazon MSK Replicator adalah fitur MSK Amazon yang memungkinkan Anda mereplikasi data dengan andal di seluruh kluster MSK Amazon di wilayah yang berbeda atau sama. AWS Dengan MSK Replicator, Anda dapat dengan mudah membangun aplikasi streaming yang tangguh secara regional untuk meningkatkan ketersediaan dan kelangsungan bisnis. MSK Replicator menyediakan replikasi asinkron otomatis di seluruh kluster MSK, menghilangkan kebutuhan untuk menulis kode khusus, mengelola infrastruktur, atau mengatur jaringan lintas wilayah.

MSK Replicator secara otomatis menskalakan sumber daya yang mendasarinya sehingga Anda dapat mereplikasi data sesuai permintaan tanpa harus memantau atau menskalakan kapasitas. MSK Replicator juga mereplikasi metadata Kafka yang diperlukan termasuk konfigurasi topik, Daftar Kontrol Akses (ACL), dan offset grup konsumen. Jika peristiwa tak terduga terjadi di suatu wilayah, Anda dapat melakukan failover ke AWS wilayah lain dan melanjutkan pemrosesan dengan mulus.

MSK Replicator mendukung replikasi lintas wilayah (CRR) dan replikasi wilayah yang sama (SRR). Dalam replikasi lintas wilayah, kluster MSK sumber dan target berada di Wilayah yang berbeda. AWS Dalam replikasi wilayah yang sama, baik kluster MSK sumber maupun target berada di Wilayah yang sama. AWS Anda perlu membuat sumber dan target kluster MSK sebelum menggunakannya dengan MSK Replicator.

> **Note**
>
> MSK Replicator mendukung AWS Wilayah berikut: AS Timur (us-east-1, Virginia N.); AS Timur (us-east-2, Ohio); AS Barat (us-west-2, Oregon); Eropa (eu-west-1, Irlandia); Eropa (eu-west-1, Irlandia); Eropa (eu-central-1, Frankfurt); Asia Pasifik (ap-southeast-1, Singapura); Asia Pasifik (ap-southeast-2, Sydney), Eropa (eu-north-1, Stockholm), Asia Pasifik (ap-south-1, Mumbai), Eropa (eu-west-3, Paris), Amerika Selatan (sa-timur-1, Mumbai), Eropa (eu-west-3, Paris), Amerika Selatan (sa-timur-1 São Paulo), Asia Pasifik (ap-northeast-2, Seoul), Eropa (eu-west-2, London), Asia Pasifik (ap-northeast-1, Tokyo), AS Barat (us-west-1, California Utara), Kanada (ca-central-1, Tengah).

Berikut adalah beberapa kegunaan umum untuk replikator MSK Amazon.

- Bangun aplikasi streaming multi-wilayah: Bangun aplikasi streaming yang sangat tersedia dan toleran terhadap kesalahan untuk meningkatkan ketahanan tanpa menyiapkan solusi khusus.
- Akses data latensi yang lebih rendah: Menyediakan akses data latensi yang lebih rendah ke konsumen di berbagai wilayah geografis.
- Mendistribusikan data ke mitra Anda: Salin data dari satu cluster Apache Kafka ke banyak cluster Apache Kafka, sehingga tim/mitra yang berbeda memiliki salinan data mereka sendiri.
- Data agregat untuk analitik: Salin data dari beberapa cluster Apache Kafka ke dalam satu cluster untuk menghasilkan wawasan tentang data real-time agregat dengan mudah.
- Tulis secara lokal, akses data Anda secara global: Siapkan replikasi multi-aktif untuk secara otomatis menyebarkan penulisan yang dilakukan di satu AWS Wilayah ke Wilayah lain untuk menyediakan data dengan latensi dan biaya lebih rendah.

Bagaimana Amazon MSK Replicator bekerja

Untuk memulai dengan MSK Replicator, Anda perlu membuat Replicator baru di Region cluster target Anda. AWS MSK Replicator secara otomatis menyalin semua data dari cluster di AWS Wilayah utama yang disebut sumber ke cluster di wilayah tujuan yang disebut target. Cluster sumber dan target dapat berada di AWS Wilayah yang sama atau berbeda. Anda perlu membuat cluster target jika belum ada.

Saat Anda membuat Replicator, MSK Replicator menyebarkan semua sumber daya yang diperlukan di AWS Region cluster target untuk mengoptimalkan latensi replikasi data. Latensi replikasi bervariasi berdasarkan banyak faktor, termasuk jarak jaringan antara AWS Wilayah kluster MSK Anda, kapasitas throughput kluster sumber dan target Anda, dan jumlah partisi pada kluster sumber dan target Anda. MSK Replicator secara otomatis menskalakan sumber daya yang mendasarinya sehingga Anda dapat mereplikasi data sesuai permintaan tanpa harus memantau atau menskalakan kapasitas.

Replikasi Data

Secara default, MSK Replicator menyalin semua data secara asinkron dari offset terbaru di partisi topik cluster sumber ke cluster target. Jika pengaturan “Deteksi dan salin topik baru” diaktifkan, MSK Replicator secara otomatis mendeteksi dan menyalin topik atau partisi topik baru ke cluster target. Namun, mungkin diperlukan waktu hingga 30 detik bagi Replicator untuk mendeteksi dan membuat topik baru atau partisi topik pada cluster target. Setiap pesan yang dihasilkan ke topik sumber sebelum topik dibuat di kluster target tidak akan direplikasi. Atau, Anda dapat mengonfigurasi

Replicator selama pembuatan untuk memulai replikasi dari offset paling awal di partisi topik cluster sumber jika Anda ingin mereplikasi pesan yang ada pada topik Anda ke kluster target.

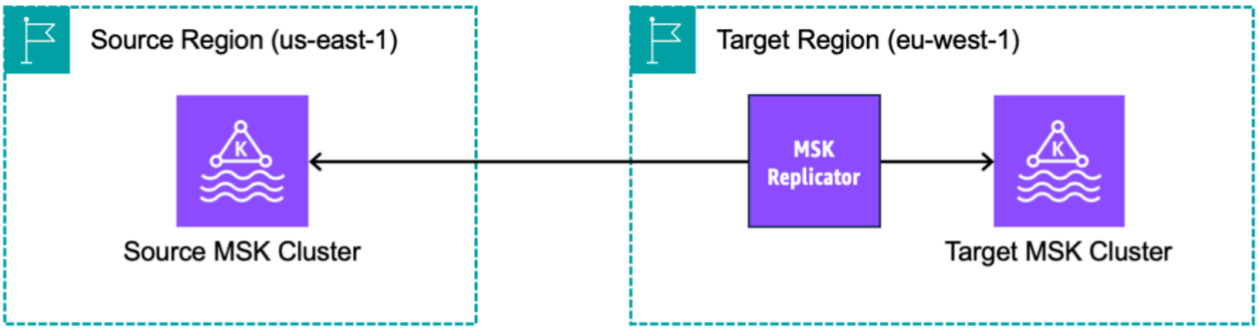
Replikator MSK tidak menyimpan data Anda. Data dikonsumsi dari cluster sumber Anda, di-buffer dalam memori dan ditulis ke cluster target. Buffer dihapus secara otomatis ketika data berhasil ditulis atau gagal setelah mencoba ulang. Semua komunikasi dan data antara MSK Replicator dan cluster Anda selalu dienkripsi dalam perjalanan. Semua panggilan MSK Replicator API seperti`DescribeClusterV2`,`CreateTopic`, `DescribeTopicDynamicConfiguration` ditangkap. AWS CloudTrail Log broker MSK Anda juga akan mencerminkan hal yang sama.

MSK Replicator membuat topik di cluster target dengan Faktor Replikator 3. Jika perlu, Anda dapat memodifikasi faktor replikasi langsung pada cluster target.

Replikasi Metadata

MSK Replicator juga mendukung penyalinan metadata dari cluster sumber ke cluster target. Metadata mencakup konfigurasi topik, baca Daftar Kontrol Akses (ACL), dan offset grup konsumen. Seperti replikasi data, replikasi metadata juga terjadi secara asinkron. Untuk kinerja yang lebih baik, MSK Replicator memprioritaskan replikasi data daripada replikasi metadata.

Sebagai bagian dari sinkronisasi offset grup konsumen, MSK Replicator mengoptimalkan untuk konsumen Anda di cluster sumber yang membaca dari posisi yang lebih dekat ke ujung aliran (akhir partisi topik). Jika grup konsumen Anda tertinggal di cluster sumber, Anda mungkin melihat lag yang lebih tinggi untuk kelompok konsumen pada target dibandingkan dengan sumbernya. Ini berarti setelah failover ke cluster target, konsumen Anda akan memproses ulang lebih banyak pesan duplikat. Untuk mengurangi lag ini, konsumen Anda di cluster sumber perlu mengejar ketinggalan dan mulai mengkonsumsi dari ujung aliran (akhir partisi topik). Saat konsumen Anda mengejar ketinggalan, MSK Replicator akan secara otomatis mengurangi lag.



Persyaratan dan pertimbangan untuk membuat Replikator MSK Amazon

Perhatikan persyaratan klaster MSK ini untuk menjalankan Replikator MSK Amazon.

Topik

- Izin yang diperlukan untuk membuat Replikator MSK
- Jenis dan versi cluster yang didukung
- MSK Konfigurasi klaster tanpa server
- Perubahan konfigurasi cluster

Izin yang diperlukan untuk membuat Replikator MSK

Berikut adalah contoh kebijakan IAM yang diperlukan untuk membuat Replikator MSK. Tindakan `kafka:TagResource` ini hanya diperlukan jika tag disediakan saat membuat Replikator MSK.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "iam:PassRole",
```

```
                "iam:CreateServiceLinkedRole",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeVpcs",
                "kafka:CreateReplicator",
                "kafka:TagResource"
            ],
            "Resource": "*"
        }
    ]
}
```

Berikut ini adalah contoh kebijakan IAM untuk menggambarkan replikator. Entah `kafka:DescribeReplicator` tindakan atau `kafka:ListTagsForResource` tindakan diperlukan, bukan keduanya.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "kafka:DescribeReplicator",
                "kafka:ListTagsForResource"
            ],
            "Resource": "*"
        }
    ]
}
```

Jenis dan versi cluster yang didukung

Ini adalah persyaratan untuk jenis instans yang didukung, versi Kafka, dan konfigurasi jaringan.

- MSK Replicator mendukung kluster yang disediakan MSK dan kluster MSK Tanpa Server dalam kombinasi apa pun sebagai cluster sumber dan target. Jenis cluster Kafka lainnya tidak didukung saat ini oleh MSK Replicator.
- MSK Kluster tanpa server memerlukan kontrol akses IAM, tidak mendukung replikasi Apache Kafka ACL dan dengan dukungan terbatas replikasi konfigurasi pada topik. Lihat MSK Tanpa Server.

- MSK Replicator hanya didukung pada cluster yang menjalankan Apache Kafka 2.7.0 atau lebih tinggi, terlepas dari apakah sumber dan kluster target Anda berada di Wilayah yang sama atau berbeda. AWS
- MSK Replicator mendukung cluster menggunakan tipe instance m5.large atau lebih besar. t3.small cluster tidak didukung.
- Jika Anda menggunakan MSK Replicator dengan kluster MSK Provisioned, Anda memerlukan minimal tiga broker di cluster sumber dan target. Anda dapat mereplikasi data di seluruh cluster di dua Availability Zone, tetapi Anda akan membutuhkan minimal empat broker di cluster tersebut.
- Kluster MSK sumber dan target Anda harus berada di akun yang sama AWS . Replikasi di seluruh cluster di akun yang berbeda tidak didukung.
- Jika cluster MSK sumber dan target berada di Wilayah yang berbeda (lintas AWS wilayah), MSK Replicator mengharuskan cluster sumber untuk mengaktifkan konektivitas pribadi multi-VPC untuk metode Kontrol Akses IAM-nya. Multi-VPC tidak diperlukan untuk metode otentikasi lain pada cluster sumber. Multi-VPC tidak diperlukan jika Anda mereplikasi data antar cluster di Wilayah yang sama. AWS Lihat the section called “Konektivitas pribadi multi-VPC dalam satu Wilayah”.

MSK Konfigurasi klaster tanpa server

- MSK Tanpa Server mendukung replikasi konfigurasi topik ini untuk kluster target MSK Tanpa Server selama pembuatan topik:,,,. `cleanup.policy` `compression.type` `max.message.bytes` `retention.bytes` `retention.ms`
- MSK Tanpa Server hanya mendukung konfigurasi topik ini selama sinkronisasi konfigurasi topik:`compression.type`,,,. `max.message.bytes` `retention.bytes` `retention.ms`
- Replicator menggunakan 83 partisi yang dipadatkan pada kluster MSK Serverless target. Pastikan bahwa target MSK Serverless cluster memiliki jumlah partisi yang dipadatkan yang cukup. Lihat MSK Kuota Tanpa Server.

Perubahan konfigurasi cluster

- Disarankan agar Anda tidak mengaktifkan atau menonaktifkan penyimpanan berjenjang setelah Replikator MSK dibuat. Jika kluster target Anda tidak berjenjang, maka MSK tidak akan menyalin konfigurasi penyimpanan berjenjang, terlepas dari apakah cluster sumber Anda berjenjang atau tidak. Jika Anda mengaktifkan penyimpanan berjenjang pada cluster target setelah Replicator dibuat, Replicator perlu dibuat ulang. Jika Anda ingin menyalin data dari klaster yang tidak

berjenjang ke klaster berjenjang, Anda tidak boleh menyalin konfigurasi topik. Lihat Mengaktifkan dan menonaktifkan penyimpanan berjenjang pada topik yang ada.

- Jangan ubah pengaturan konfigurasi cluster setelah pembuatan MSK Replicator. Pengaturan konfigurasi cluster divalidasi selama pembuatan MSK Replicator. Untuk menghindari masalah dengan MSK Replicator, jangan mengubah pengaturan berikut setelah MSK Replicator dibuat.
  - Ubah cluster MSK ke tipe instans t3.
  - Ubah izin peran eksekusi layanan.
  - Nonaktifkan konektivitas pribadi MSK Multi-VPC.
  - Ubah kebijakan berbasis sumber daya klaster terlampir.
  - Ubah aturan grup keamanan klaster.

Memulai menggunakan Amazon MSK Replicator

Tutorial ini menunjukkan kepada Anda cara mengatur cluster sumber dan claster target di Wilayah yang sama atau di AWS AWS Wilayah yang berbeda. Anda kemudian menggunakan cluster tersebut untuk membuat Replikator MSK Amazon.

Langkah 1: Siapkan cluster sumber MSK Amazon

Jika Anda sudah memiliki cluster sumber MSK yang dibuat untuk MSK Replicator, pastikan bahwa itu memenuhi persyaratan yang dijelaskan di bagian ini. Jika tidak, ikuti langkah-langkah ini untuk membuat kluster sumber yang disediakan MSK atau tanpa server.

Proses untuk membuat cluster sumber MSK Replicator lintas wilayah dan wilayah yang sama serupa. Perbedaan disebut dalam prosedur berikut.

1. Buat kluster MSK yang disediakan atau tanpa server dengan kontrol akses IAM diaktifkan di wilayah sumber. Cluster sumber Anda harus memiliki minimal tiga broker.
2. Untuk Replikator MSK lintas wilayah, jika sumbernya adalah kluster yang disediakan, konfigurasikan dengan konektivitas pribadi multi-VPC yang diaktifkan untuk skema kontrol akses IAM. Perhatikan bahwa jenis autentikasi yang tidak diautentikasi tidak didukung saat multi-VPC dihidupkan. Anda tidak perlu mengaktifkan konektivitas pribadi multi-VPC untuk skema otentikasi lainnya (mTLS atau SASL/SCRAM). Anda dapat secara bersamaan menggunakan skema autentikasi MTL atau SASL/SCRAM untuk klien Anda yang lain yang terhubung ke cluster MSK Anda. Anda dapat mengonfigurasi konektivitas pribadi multi-VPC di detail cluster konsol Pengaturan jaringan atau dengan API. `UpdateConnectivity` Lihat Pemilik cluster

mengaktifkan Multi-VPC. Jika cluster sumber Anda adalah kluster MSK Serverlesss, Anda tidak perlu mengaktifkan konektivitas pribadi multi-VPC.

Untuk Replikator MSK wilayah yang sama, kluster sumber MSK tidak memerlukan konektivitas pribadi multi-VPC dan cluster masih dapat diakses oleh klien lain menggunakan jenis autentikasi yang tidak diautentikasi.

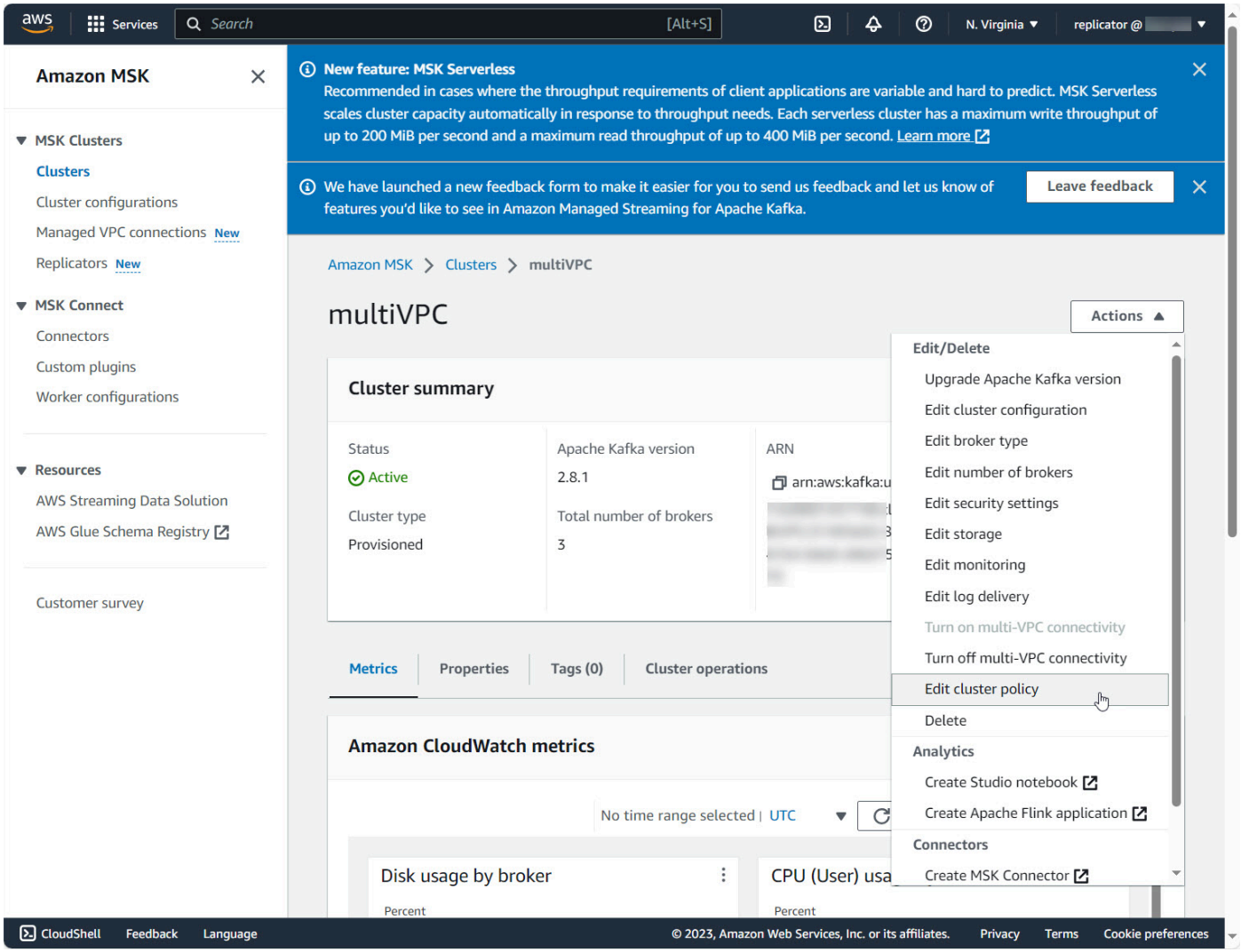
3. Untuk Replikator MSK lintas wilayah, Anda harus melampirkan kebijakan izin berbasis sumber daya ke kluster sumber. Hal ini memungkinkan MSK untuk terhubung ke cluster ini untuk mereplikasi data. Anda dapat melakukan ini menggunakan prosedur CLI atau AWS Konsol di bawah ini. Lihat juga, kebijakan berbasis sumber daya Amazon MSK. Anda tidak perlu melakukan langkah ini untuk Replikator MSK wilayah yang sama.

Console: create resource policy

Perbarui kebijakan cluster sumber dengan JSON berikut. Ganti placeholder dengan ARN dari cluster sumber Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Service": [
                "kafka.amazonaws.com"
            ]
        },
        "Action": [
            "kafka:CreateVpcConnection",
            "kafka:GetBootstrapBrokers",
            "kafka:DescribeClusterV2"
        ],
        "Resource": "<sourceClusterARN>"
    }
  ]
}
```

Gunakan opsi Edit kebijakan klaster di bawah menu Tindakan pada halaman detail klaster.



CLI: create resource policy

Catatan: Jika Anda menggunakan AWS konsol untuk membuat kluster sumber dan memilih opsi untuk membuat peran IAM baru, AWS lampirkan kebijakan kepercayaan yang diperlukan ke peran tersebut. Jika Anda ingin MSK menggunakan peran IAM yang ada atau jika Anda membuat peran sendiri, lampirkan kebijakan kepercayaan berikut ke peran tersebut sehingga MSK Replicator dapat mengasumsikan itu. Untuk informasi tentang cara mengubah hubungan kepercayaan suatu peran, lihat Mengubah Peran.

1. Dapatkan versi kebijakan cluster MSK saat ini menggunakan perintah ini. Ganti placeholder dengan ARN cluster yang sebenarnya.

```
aws kafka get-cluster-policy –cluster-arn <Cluster ARN>
{
"CurrentVersion": "K1PA6795UKM GR7",
```

```
"Policy": "..."
}
```

2. Buat kebijakan berbasis sumber daya untuk memungkinkan MSK Replicator mengakses kluster sumber Anda. Gunakan sintaks berikut sebagai template, menggantikan placeholder dengan ARN cluster sumber yang sebenarnya.

```
aws kafka put-cluster-policy --cluster-arn "<sourceClusterARN>" --policy '{
"Version": "2012-10-17",
"Statement": [
{
"Effect": "Allow",
"Principal": {
"Service": [
"kafka.amazonaws.com"
]
},
"Action": [
"kafka:CreateVpcConnection",
"kafka:GetBootstrapBrokers",
"kafka:DescribeClusterV2"
],
"Resource": "<sourceClusterARN>"
}
]
```

Langkah 2: Siapkan kluster target MSK Amazon

Buat kluster target MSK (disediakan atau tanpa server) dengan kontrol akses IAM diaktifkan. Cluster target tidak memerlukan konektivitas pribadi multi-VPC dihidupkan. Cluster target dapat berada di AWS Wilayah yang sama atau Wilayah yang berbeda sebagai cluster sumber. Cluster sumber dan target harus berada di AWS akun yang sama. Cluster target Anda harus memiliki minimal tiga broker.

Langkah 3: Buat Replikator MSK Amazon

Sebelum Anda membuat Amazon MSK Replicator, pastikan Anda memilikinya. Izin yang diperlukan untuk membuat Replikator MSK

Topik

- Buat replikator menggunakan AWS konsol di wilayah cluster target

- Pilih cluster sumber Anda
- Pilih klaster target Anda
- Konfigurasikan pengaturan dan izin replikator

Buat replikator menggunakan AWS konsol di wilayah cluster target

1. Di AWS Wilayah tempat cluster MSK target Anda berada, buka konsol MSK Amazon di https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Pilih Replikator untuk menampilkan daftar replikator di akun.
3. Pilih Buat replikator.
4. Di panel Detail Replicator, berikan replikator baru nama yang unik.

Pilih cluster sumber Anda

Cluster sumber berisi data yang ingin Anda salin ke kluster MSK target.

1. Di panel cluster Sumber, pilih AWS Wilayah tempat kluster sumber berada.

   Anda dapat mencari Region cluster dengan pergi ke MSK Clusters dan melihat rincian Cluster ARN. Nama Region disematkan dalam string ARN. Dalam contoh ARN berikut, `ap-southeast-2` adalah wilayah cluster.

   ```
   arn:aws:kafka:ap-southeast-2:123456789012:cluster/cluster-11/
   eec93c7f-4e8b-4baf-89fb-95de01ee639c-s1
   ```

2. Masukkan ARN cluster sumber Anda atau telusuri untuk memilih cluster sumber Anda.
3. Pilih subnet untuk cluster sumber Anda.

   Konsol menampilkan subnet yang tersedia di Wilayah cluster sumber untuk Anda pilih. Anda harus memilih minimal dua subnet. Untuk Replicator MSK wilayah yang sama, subnet yang Anda pilih disetel untuk mengakses cluster sumber dan subnet untuk mengakses kluster target harus berada di Availability Zone yang sama.
4. Pilih grup keamanan untuk replikator MSK untuk mengakses kluster sumber Anda.

   - Untuk replikasi lintas wilayah (CRR), Anda tidak perlu menyediakan grup keamanan untuk cluster sumber Anda.

- Untuk replikasi wilayah yang sama (SRR), buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/ dan pastikan bahwa grup keamanan yang akan Anda sediakan untuk Replicator memiliki aturan keluar untuk mengizinkan lalu lintas ke grup keamanan cluster sumber Anda. Selain itu, pastikan bahwa grup keamanan cluster sumber Anda memiliki aturan masuk yang memungkinkan lalu lintas dari grup keamanan Replicator yang disediakan untuk sumbernya.

  Untuk menambahkan aturan masuk ke grup keamanan cluster sumber Anda:

  1. Di AWS konsol, buka detail cluster sumber Anda dengan memilih nama Cluster.
  2. Pilih tab Properti, lalu gulir ke bawah ke panel Pengaturan jaringan untuk memilih nama grup Keamanan yang diterapkan.
  3. Buka aturan masuk dan pilih Edit aturan masuk.
  4. Pilih Tambahkan aturan.
  5. Di kolom Type untuk aturan baru, pilih Custom TCP.
  6. Di kolom rentang Port, ketik`9098`. MSK Replicator menggunakan kontrol akses IAM untuk terhubung ke cluster Anda yang menggunakan port 9098.
  7. Di kolom Sumber, ketikkan nama grup keamanan yang akan Anda berikan selama pembuatan Replicator untuk cluster sumber (ini mungkin sama dengan grup keamanan kluster sumber MSK), lalu pilih Simpan aturan.

  Untuk menambahkan aturan keluar ke grup keamanan Replicator yang disediakan untuk sumber:

  1. Di AWS konsol untuk Amazon EC2, buka grup keamanan yang akan Anda berikan selama pembuatan Replicator untuk sumbernya.
  2. Buka aturan keluar dan pilih Edit aturan keluar.
  3. Pilih Tambahkan aturan.
  4. Di kolom Type untuk aturan baru, pilih Custom TCP.
  5. Di kolom rentang Port, ketik`9098`. MSK Replicator menggunakan kontrol akses IAM untuk terhubung ke cluster Anda yang menggunakan port 9098.
  6. Di kolom Sumber, ketik nama grup keamanan kluster sumber MSK, lalu pilih Simpan aturan.

> **Note**
>
> Sebagai alternatif, jika Anda tidak ingin membatasi lalu lintas menggunakan grup keamanan Anda, Anda dapat menambahkan aturan masuk dan keluar yang memungkinkan Semua Lalu Lintas.
> 1. Pilih Tambahkan aturan.
> 2. Di kolom Type, pilih All Traffic.
> 3. Di kolom Sumber, ketik`0.0.0.0/0`, lalu pilih Simpan aturan.

Pilih klaster target Anda

Cluster target adalah kluster yang disediakan MSK atau tanpa server tempat data sumber disalin.

> **Note**
>
> MSK Replicator membuat topik baru di cluster target dengan awalan yang dibuat otomatis ditambahkan ke nama topik. Misalnya, MSK Replicator mereplikasi data dalam "`topic`" dari cluster sumber ke topik baru di cluster target yang disebut. `<sourceKafkaClusterAlias>.topic` Ini untuk membedakan topik yang berisi data yang direplikasi dari cluster sumber dari topik lain di cluster target dan untuk menghindari data yang direplikasi secara sirkuler antara cluster. Anda dapat menemukan awalan yang akan ditambahkan ke nama topik di kluster target di bawah bidang sourceKafkaClusterAlias menggunakan `DescribeReplicator` API atau halaman detail Replicator di Konsol MSK. Awalan di cluster target adalah **<** sourceKafkaCluster Alias**>**.

1. Di panel cluster Target, pilih AWS Wilayah tempat klaster target berada.
2. Masukkan ARN cluster target Anda atau telusuri untuk memilih cluster target Anda.
3. Pilih subnet untuk kluster target Anda.

   Konsol menampilkan subnet yang tersedia di Wilayah kluster target untuk Anda pilih. Pilih minimal dua subnet.
4. Pilih grup keamanan untuk replikator MSK untuk mengakses kluster target Anda.

   Grup keamanan yang tersedia di Wilayah kluster target ditampilkan untuk Anda pilih. Grup keamanan yang dipilih dikaitkan dengan setiap koneksi. Untuk informasi selengkapnya

tentang menggunakan grup keamanan, lihat Mengontrol lalu lintas ke AWS sumber daya Anda menggunakan grup keamanan di Panduan Pengguna Amazon VPC.

- Untuk replikasi lintas wilayah (CRR) dan replikasi wilayah yang sama (SRR), buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/ dan pastikan bahwa grup keamanan yang akan Anda berikan kepada Replicator memiliki aturan keluar untuk mengizinkan lalu lintas ke grup keamanan kluster target Anda. Pastikan juga bahwa grup keamanan kluster target Anda memiliki aturan masuk yang menerima lalu lintas dari grup keamanan Replicator yang disediakan untuk target.

  Untuk menambahkan aturan masuk ke grup keamanan klaster target Anda:

  1. Di AWS konsol, buka detail kluster target Anda dengan memilih nama Cluster.
  2. Pilih tab Properti, lalu gulir ke bawah ke panel Pengaturan jaringan untuk memilih nama grup Keamanan yang diterapkan.
  3. Buka aturan masuk dan pilih Edit aturan masuk.
  4. Pilih Tambahkan aturan.
  5. Di kolom Type untuk aturan baru, pilih Custom TCP.
  6. Di kolom rentang Port, ketik`9098`. MSK Replicator menggunakan kontrol akses IAM untuk terhubung ke cluster Anda yang menggunakan port 9098.
  7. Di kolom Sumber, ketikkan nama grup keamanan yang akan Anda berikan selama pembuatan Replicator untuk kluster target (ini mungkin sama dengan grup keamanan kluster target MSK), lalu pilih Simpan aturan.

  Untuk menambahkan aturan keluar ke grup keamanan Replicator yang disediakan untuk target:

  1. Di AWS konsol, buka grup keamanan yang akan Anda berikan selama pembuatan Replicator untuk target.
  2. Pilih tab Properti, lalu gulir ke bawah ke panel Pengaturan jaringan untuk memilih nama grup Keamanan yang diterapkan.
  3. Buka aturan keluar dan pilih Edit aturan keluar.
  4. Pilih Tambahkan aturan.
  5. Di kolom Type untuk aturan baru, pilih Custom TCP.

6. Di kolom rentang Port, ketik`9098`. MSK Replicator menggunakan kontrol akses IAM untuk terhubung ke cluster Anda yang menggunakan port 9098.
7. Di kolom Sumber, ketik nama grup keamanan kluster target MSK, lalu pilih Simpan aturan.

> **Note**
>
> Sebagai alternatif, jika Anda tidak ingin membatasi lalu lintas menggunakan grup keamanan Anda, Anda dapat menambahkan aturan masuk dan keluar yang memungkinkan Semua Lalu Lintas.
> 1. Pilih Tambahkan aturan.
> 2. Di kolom Type, pilih All Traffic.
> 3. Di kolom Sumber, ketik`0.0.0.0/0`, lalu pilih Simpan aturan.

Konfigurasikan pengaturan dan izin replikator

1. Di panel Pengaturan replikator, tentukan topik yang ingin Anda tiru menggunakan ekspresi reguler dalam daftar izinkan dan tolak. Secara default, semua topik direplikasi.

   > **Note**
   >
   > MSK Replicator hanya mereplikasi hingga 750 topik dalam urutan yang diurutkan. Jika Anda perlu mereplikasi lebih banyak topik, kami sarankan Anda membuat Replicator terpisah. Buka Pusat Dukungan AWS konsol dan buat kasus dukungan jika Anda memerlukan dukungan untuk lebih dari 750 topik per Replicator. Anda dapat memantau jumlah topik yang direplikasi menggunakan metrik TopicCount "". Lihat Kuota MSK Amazon.

2. Secara default, MSK Replicator memulai replikasi dari offset terbaru (terbaru) dalam topik yang dipilih. Atau, Anda dapat memulai replikasi dari offset paling awal (tertua) dalam topik yang dipilih jika Anda ingin mereplikasi data yang ada pada topik Anda. Setelah replikator dibuat, Anda tidak dapat mengubah pengaturan ini. Pengaturan ini sesuai dengan `startingPosition`bidang di API `CreateReplicator`permintaan dan `DescribeReplicator`respons.

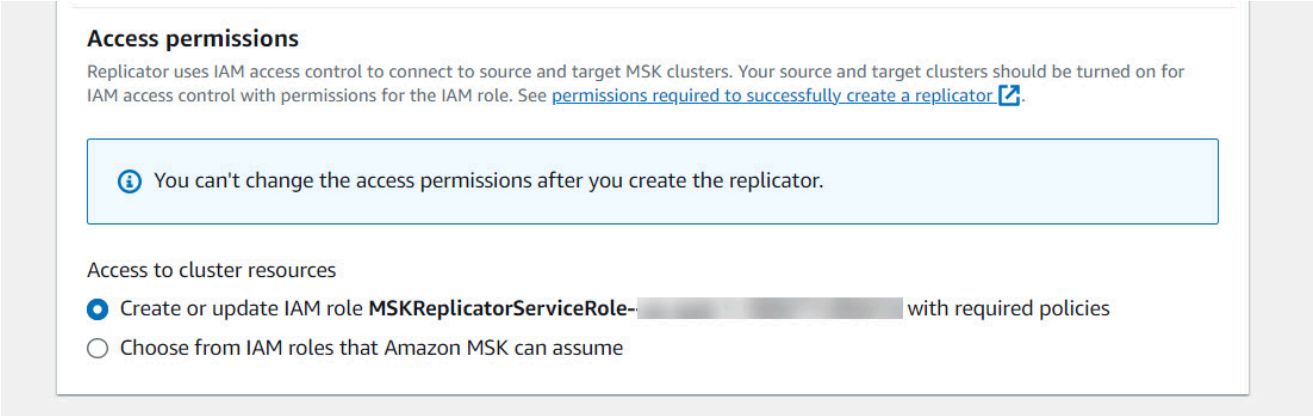
> **Note**
>
> MSK Replicator bertindak seperti konsumen baru untuk cluster sumber Anda. Bergantung pada jumlah data yang Anda replikasi dan kapasitas konsumsi yang Anda miliki di cluster sumber Anda, ini dapat menyebabkan konsumen lain di cluster sumber Anda terhambat. Jika Anda membuat Replicator yang disetel ke posisi awal paling awal, MSK Replicator akan membaca ledakan data di awal yang mungkin menghabiskan semua kapasitas konsumsi cluster sumber Anda. Setelah Replicator Anda menyusul, tingkat konsumsi akan turun agar sesuai dengan throughput topik cluster sumber Anda. Jika Anda mereplikasi dari posisi paling awal, kami sarankan Anda mengelola throughput Replicator menggunakan kuota Kafka untuk memastikan konsumen lain tidak dibatasi.

3. Secara default, MSK Replicator menyalin semua metadata termasuk konfigurasi topik, Access Control Lists (ACL) dan offset grup konsumen untuk failover yang mulus. Jika Anda tidak membuat Replicator untuk failover, Anda dapat memilih untuk mematikan satu atau beberapa pengaturan ini yang tersedia di bagian Pengaturan tambahan.

> **Note**
>
> MSK Replicator tidak mereplikasi ACL tulis karena produser Anda tidak boleh menulis langsung ke topik yang direplikasi di cluster target. Produser Anda harus menulis ke topik lokal di cluster target setelah failover. Lihat Melakukan failover yang direncanakan ke Wilayah sekunder AWS untuk detail.

4. Di panel replikasi grup konsumen, tentukan grup konsumen yang ingin direplikasi menggunakan ekspresi reguler dalam daftar izinkan dan tolak. Secara default, semua grup konsumen direplikasi.
5. Di panel Kompresi, Anda dapat memilih untuk mengompres data yang ditulis ke kluster target. Jika Anda akan menggunakan kompresi, kami sarankan Anda menggunakan metode kompresi yang sama dengan data di cluster sumber Anda.
6. Di panel Izin akses lakukan salah satu hal berikut:

    a. Pilih Buat atau perbarui peran IAM dengan kebijakan yang diperlukan. Konsol MSK akan secara otomatis melampirkan izin dan kebijakan kepercayaan yang diperlukan ke peran eksekusi layanan yang diperlukan untuk membaca dan menulis ke sumber dan target kluster MSK Anda.



b. Berikan peran IAM Anda sendiri dengan memilih Pilih dari peran IAM yang dapat diasumsikan oleh Amazon MSK. Kami menyarankan Anda melampirkan kebijakan IAM `AWSMSKReplicatorExecutionRole` terkelola ke peran eksekusi layanan Anda, alih-alih menulis kebijakan IAM Anda sendiri.

   - Buat peran IAM yang akan digunakan replikator untuk membaca dan menulis ke sumber Anda dan target kluster MSK dengan JSON di bawah ini sebagai bagian dari kebijakan kepercayaan dan terlampir pada peran tersebut`AWSMSKReplicatorExecutionRole`. Dalam kebijakan kepercayaan, ganti placeholder <yourAccountID>dengan ID akun Anda yang sebenarnya.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "kafka.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<yourAccountID>"
                }
            }
        }
    ]
}
```

7. Di panel tag Replicator, Anda dapat secara opsional menetapkan tag ke sumber daya MSK Replicator. Untuk informasi selengkapnya, lihat Menandai kluster MSK Amazon. Untuk

Replikator MSK lintas wilayah, tag disinkronkan ke Wilayah jarak jauh secara otomatis saat Replicator dibuat. Jika Anda mengubah tag setelah Replicator dibuat, perubahan tidak secara otomatis disinkronkan ke Wilayah jarak jauh, jadi Anda harus menyinkronkan replikator lokal dan referensi replikator jarak jauh secara manual.

8. Pilih Buat.

Jika Anda ingin membatasi `kafka-cluster:WriteData` izin, lihat bagian Buat kebijakan otorisasi dari Cara kerja kontrol akses IAM untuk Amazon MSK. Anda harus menambahkan `kafka-cluster:WriteDataIdempotently` izin ke cluster sumber dan target.

Dibutuhkan sekitar 30 menit agar Replikator MSK berhasil dibuat dan dialihkan ke status RUNNING.

Jika Anda membuat Replikator MSK baru untuk menggantikan yang Anda hapus, Replicator baru memulai replikasi dari offset terbaru.

Jika Replikator MSK Anda telah beralih ke status GAGAL, lihat bagian pemecahan masalah Pemecahan Masalah Replikator MSK.

Edit pengaturan MSK Replicator

Anda tidak dapat mengubah posisi awal cluster sumber, cluster target, atau Replicator setelah MSK Replicator dibuat. Namun, Anda dapat mengedit pengaturan Replicator lainnya, seperti topik dan grup konsumen untuk ditiru.

1. Masuk ke AWS Management Console, dan buka konsol MSK Amazon di https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Di panel navigasi kiri, pilih Replikator untuk menampilkan daftar replikator di akun dan pilih Replikator MSK yang ingin Anda edit.
3. Pilih tab Properti.
4. Di bagian Pengaturan replikator, pilih Edit replikator.
5. Anda dapat mengedit pengaturan MSK Replicator dengan mengubah salah satu pengaturan ini.

    - Tentukan topik yang ingin Anda tiru menggunakan ekspresi reguler dalam daftar izinkan dan tolak. Secara default, MSK Replicator menyalin semua metadata termasuk konfigurasi topik, Access Control Lists (ACL) dan offset grup konsumen untuk failover yang mulus. Jika Anda

tidak membuat Replicator untuk failover, Anda dapat memilih untuk mematikan satu atau beberapa pengaturan ini yang tersedia di bagian Pengaturan tambahan.

> **Note**
>
> MSK Replicator tidak mereplikasi ACL tulis karena produser Anda tidak boleh menulis langsung ke topik yang direplikasi di cluster target. Produser Anda harus menulis ke topik lokal di cluster target setelah failover. Lihat Melakukan failover yang direncanakan ke Wilayah sekunder AWS untuk detail.

- Untuk replikasi grup Konsumen, Anda dapat menentukan grup konsumen yang ingin direplikasi menggunakan ekspresi reguler dalam daftar izinkan dan tolak. Secara default, semua grup konsumen direplikasi. Jika daftar allow dan deny kosong, replikasi grup konsumen dimatikan.
- Di bawah Jenis kompresi target, Anda dapat memilih apakah akan mengompres data yang ditulis ke cluster target. Jika Anda akan menggunakan kompresi, kami sarankan Anda menggunakan metode kompresi yang sama dengan data di cluster sumber Anda.

6. Simpan perubahan Anda.

   Dibutuhkan sekitar 30 menit agar Replikator MSK berhasil dibuat dan dialihkan ke status berjalan. Jika Replikator MSK Anda telah beralih ke status GAGAL, lihat bagian pemecahan masalah. ???

Hapus Replikator MSK

Anda mungkin perlu menghapus Replikator MSK jika gagal membuat (status GAGAL). Cluster sumber dan target yang ditetapkan ke Replikator MSK tidak dapat diubah setelah Replikator MSK dibuat. Anda dapat menghapus Replikator MSK yang ada dan membuat yang baru. Jika Anda membuat Replikator MSK baru untuk menggantikan yang dihapus, Replicator baru memulai replikasi dari offset terbaru.

1. Di AWS Wilayah tempat cluster sumber Anda berada, masuk ke AWS Management Console, dan buka konsol MSK Amazon di https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Di panel navigasi, pilih Replicators.
3. Dari daftar Replikator MSK, pilih salah satu yang ingin Anda hapus dan pilih Hapus.

Pantau replikasi

Anda dapat menggunakan https://console.aws.amazon.com/cloudwatch/ di Wilayah kluster target untuk melihat metrik`ReplicationLatency`,`MessageLag`, dan `ReplicatorThroughput` pada tingkat topik dan agregat untuk setiap Replikator MSK Amazon. Metrik terlihat di bawah ReplicatorNamedi namespace “AWS/Kafka”. Anda juga dapat melihat`ReplicatorFailure`, `AuthError` dan `ThrottleTime` metrik untuk memeriksa masalah.

Konsol MSK menampilkan subset CloudWatch metrik untuk setiap Replikator MSK. Dari daftar Replicator konsol, pilih nama Replicator dan pilih tab Monitoring.

Metrik Replikator MSK

Metrik berikut menjelaskan metrik kinerja atau koneksi untuk Replikator MSK.

AuthError metrik tidak mencakup kesalahan autentikasi tingkat topik. Untuk memantau kesalahan autentikasi tingkat topik MSK Replicator Anda, pantau metrik Replicator dan ReplicationLatency metrik tingkat topik cluster sumber,. MessagesInPerSec Jika topik ReplicationLatency turun ke 0 tetapi topik masih memiliki data yang diproduksi untuk itu, ini menunjukkan bahwa Replicator memiliki masalah Auth dengan topik tersebut. Periksa apakah peran IAM eksekusi layanan Replicator memiliki izin yang cukup untuk mengakses topik.

Jenis metrik	Metrik	Deskripsi	Dimensi	Unit	Granularitas Metrik Mentah	Stat Agregasi Metrik Mentah	
Kinerja	ReplicationLatency	Waktu yang dibutuhkan catatan untuk mereplikasi dari sumber ke cluster target; durasi antara waktu produksi rekaman di sumber dan direplikasi	ReplicatorName	Milidetik	Partition	Maksimum	
			ReplicatorName, Topik	Milidetik	Partition	Maksimu	

Jenis metrik	Metrik	Deskripsi	Dimensi	Unit	Granularitas Metrik Mentah	Stat Agregasi Metrik Mentah	
		ke target. Jika ReplicationLatency meningkat, periksa apakah cluster memiliki partisi yang cukup untuk mendukung replikasi. Latensi replikasi tinggi dapat terjadi ketika jumlah partisi terlalu rendah untuk throughput tinggi.					

Jenis metrik	Metrik	Deskripsi	Dimensi	Unit	Granularitas Metrik Mentah	Stat Agregasi Metrik Mentah
Kinerja	MessageLag	Memantau sinkronisasi antara MSK Replicator dan cluster sumber. MessageLag menunjukkan jeda antara pesan yang dihasilkan ke cluster sumber dan pesan yang dikonsumsi oleh replikator. Ini bukan jeda antara cluster sumber dan target. Bahkan jika cluster sumber tidak tersedia/terputus, replikator akan selesai menulis pesan yang telah dikonsumsi ke cluster target. Setelah pemadaman, MessageLag menunjukk	ReplicatorName	Hitung	Partition	Jumlah
			ReplicatorName, Topik	Hitung	Partition	Jumlah

Jenis metrik	Metrik	Deskripsi	Dimensi	Unit	Granularitas Metrik Mentah	Stat Agregasi Metrik Mentah	
		an peningkatan yang menunjukkan jumlah pesan replikator berada di belakang cluster sumber dan ini dapat dipantau hingga jumlah pesan 0, menunjukkan bahwa replikator telah menyusul cluster sumber.					

Jenis metrik	Metrik	Deskripsi	Dimensi	Unit	Granularitas Metrik Mentah	Stat Agregasi Metrik Mentah	
Kinerja	ReplicatorThroughput	Rata-rata jumlah byte direplikasi per detik. Jika ReplicatorThroughput turun untuk topik, periksa KafkaClusterPingSuccessCount dan AuthError metrik untuk memastikan Replicator dapat berkomunikasi dengan cluster, lalu periksa metrik klaster untuk memastikan klaster tidak down.	ReplicatorName	BytesPerSecond	Partition	Jumlah	
			ReplicatorName, Topik	BytesPerSecond	Partition	Jumlah	

Jenis metrik	Metrik	Deskripsi	Dimensi	Unit	Granularitas Metrik Mentah	Stat Agregasi Metrik Mentah
Debug	AuthError	Jumlah koneksi dengan otentikasi gagal per detik. Jika metrik ini di atas 0, Anda dapat memeriksa apakah kebijakan peran eksekusi layanan untuk replikator valid dan pastikan tidak ada izin penolakan yang disetel untuk izin cluster. Berdasarkan dimensi ClusterAlias, Anda dapat mengidentifikasi apakah sumber atau kluster target mengalami kesalahan autentikasi.	ReplicatorName, ClusterAlias	Hitung	Pekerja	Jumlah

Jenis metrik	Metrik	Deskripsi	Dimensi	Unit	Granularitas Metrik Mentah	Stat Agregasi Metrik Mentah
Debug	ThrottleTime	Waktu rata-rata dalam ms permintaan dibatasi oleh broker di cluster. Atur throttling untuk menghindari MSK Replicator membanjiri cluster. Jika metrik ini 0, ReplicationLatency tidak tinggi, dan ReplicatorThroughput seperti yang diharapkan, maka throttling berfungsi seperti yang diharapkan. Jika metrik ini di atas 0, Anda dapat menyesuaikan pelambatan yang sesuai.	ReplicatorName, ClusterAlias	Milidetik	Pekerja	Maksimum
Debug	ReplicatorFailure	Jumlah kegagalan yang dialami replikator.	ReplicatorName	Hitung		Jumlah

Jenis metrik	Metrik	Deskripsi	Dimensi	Unit	Granularitas Metrik Mentah	Stat Agregasi Metrik Mentah	
Debug	KafkaClusterPingSuccessCount	Menunjukkan kesehatan koneksi replikator ke cluster kafka. Jika nilai ini 1, koneksi sehat. Jika nilainya 0 atau tidak ada titik data, koneksi tidak sehat. Jika nilainya 0, Anda dapat memeriksa pengaturan izin jaringan atau IAM untuk cluster Kafka. Berdasarkan ClusterAlias dimensi, Anda dapat mengidentifikasi apakah metrik ini untuk sumber atau cluster target.	ReplicatorName, ClusterAlias	Hitung		Jumlah	

Menggunakan replikasi untuk meningkatkan ketahanan aplikasi streaming Kafka di seluruh wilayah

Anda dapat menggunakan MSK Replicator untuk menyiapkan topologi cluster aktif-aktif atau aktif-pasif untuk meningkatkan ketahanan aplikasi Apache Kafka Anda di seluruh Wilayah. AWS Dalam pengaturan aktif-aktif, kedua kluster MSK secara aktif melayani membaca dan menulis. Dalam pengaturan aktif-pasif, hanya satu cluster MSK pada satu waktu yang secara aktif melayani data streaming, sementara cluster lainnya dalam keadaan siaga.

Pertimbangan untuk membangun aplikasi Apache Kafka multi-wilayah

Konsumen Anda harus dapat memproses ulang pesan duplikat tanpa dampak hilir. MSK Replicator mereplikasi data at-least-once yang dapat menghasilkan duplikat di cluster siaga. Ketika Anda beralih ke AWS Wilayah sekunder, konsumen Anda dapat memproses data yang sama lebih dari satu kali. MSK Replicator memprioritaskan penyalinan data daripada offset konsumen untuk kinerja yang lebih baik. Setelah failover, konsumen dapat mulai membaca dari offset sebelumnya yang menghasilkan pemrosesan duplikat.

Produsen dan konsumen juga harus mentolerir kehilangan data minimal. Karena MSK Replicator mereplikasi data secara asinkron, ketika AWS Region primer mulai mengalami kegagalan, tidak ada jaminan bahwa semua data direplikasi ke Region sekunder. Anda dapat menggunakan latensi replikasi untuk menentukan data maksimum yang tidak disalin ke Wilayah sekunder.

Menggunakan topologi cluster aktif-aktif versus aktif-pasif

Topologi cluster aktif-aktif menawarkan waktu pemulihan mendekati nol dan kemampuan aplikasi streaming Anda untuk beroperasi secara bersamaan di beberapa Wilayah. AWS Ketika sebuah cluster di satu Wilayah terganggu, aplikasi yang terhubung ke cluster di Wilayah lain terus memproses data.

Pengaturan pasif aktif cocok untuk aplikasi yang dapat berjalan hanya di satu AWS Wilayah pada satu waktu, atau ketika Anda membutuhkan kontrol lebih besar atas urutan pemrosesan data. Pengaturan aktif-pasif memerlukan lebih banyak waktu pemulihan daripada pengaturan aktif-aktif, karena Anda harus memulai seluruh pengaturan aktif-pasif Anda, termasuk produsen dan konsumen Anda, di Wilayah sekunder untuk melanjutkan streaming data setelah failover.

Membuat pengaturan cluster Kafka pasif aktif dan penamaan topik yang direplikasi

Untuk pengaturan aktif-pasif, kami menyarankan Anda untuk mengoperasikan pengaturan serupa dari produsen, klaster MSK, dan konsumen (dengan nama grup konsumen yang sama) di dua Wilayah yang berbeda. AWS Adalah penting bahwa kedua cluster MSK memiliki kapasitas baca dan tulis yang identik untuk memastikan replikasi data yang andal. Anda perlu membuat Replikator MSK untuk terus menyalin data dari primer ke cluster siaga. Anda juga perlu mengonfigurasi produsen Anda untuk menulis data ke dalam topik pada klaster di AWS Wilayah yang sama.

Untuk memastikan konsumen Anda dapat memulai ulang pemrosesan dengan andal dari klaster siaga, Anda perlu mengonfigurasi konsumen Anda untuk membaca data dari topik menggunakan operator wildcard '.*'. Misalnya, MSK Replicator mereplikasi “topic1" dari cluster utama ke topik baru di klaster siaga yang disebut “< alias>.topic1". sourceKafkaCluster Misalnya, Anda dapat mengonfigurasi produsen Anda untuk menulis ke “topic1" dan konsumen Anda untuk menggunakan”. *topik1” di kedua Wilayah. Contoh ini juga akan mencakup topik seperti footopic1, jadi sesuaikan operator wildcard sesuai dengan kebutuhan Anda.

Kapan harus failover ke Wilayah sekunder AWS

Kami menyarankan Anda memantau latensi replikasi di AWS Wilayah sekunder menggunakan. CloudWatch Selama acara layanan di AWS Wilayah primer, latensi replikasi dapat tiba-tiba meningkat. Jika latensi terus meningkat, gunakan AWS Service Health Dashboard untuk memeriksa kejadian layanan di AWS Wilayah utama. Jika ada acara, Anda dapat melakukan failover ke AWS Region sekunder.

Melakukan failover yang direncanakan ke Wilayah sekunder AWS

Anda dapat melakukan failover yang direncanakan untuk menguji ketahanan aplikasi Anda terhadap peristiwa tak terduga di AWS wilayah utama Anda yang memiliki kluster MSK sumber Anda. Failover yang direncanakan seharusnya tidak mengakibatkan kehilangan data.

1. Matikan semua produsen dan konsumen yang terhubung ke cluster sumber Anda.
2. Buat Replikator MSK baru untuk mereplikasi data dari klaster MSK Anda di Wilayah sekunder ke kluster MSK Anda di Wilayah utama. Ini diperlukan untuk menyalin data yang akan Anda tulis ke wilayah sekunder kembali ke Wilayah utama sehingga Anda dapat gagal kembali ke Wilayah utama setelah peristiwa tak terduga berakhir.

3. Mulai produsen pada cluster target di AWS Wilayah sekunder.
4. Bergantung pada persyaratan pemesanan pesan aplikasi Anda, ikuti langkah-langkah di salah satu tab berikut.

No message ordering

Jika aplikasi Anda tidak memerlukan pemesanan pesan, mulailah konsumen di AWS Wilayah sekunder yang membaca dari topik lokal (misalnya,`topic`) dan topik yang direplikasi (misalnya,`<sourceKafkaClusterAlias>.topic`) menggunakan operator wildcard (misalnya,. * topik).

Message ordering

Jika aplikasi Anda memerlukan pemesanan pesan, mulailah konsumen hanya untuk topik yang direplikasi pada klaster target (misalnya,`<sourceKafkaClusterAlias>.topic`) tetapi bukan topik lokal (misalnya,`topic`).

1. Tunggu semua konsumen topik yang direplikasi pada cluster MSK target untuk menyelesaikan pemrosesan semua data, sehingga kelambatan konsumen adalah 0 dan jumlah catatan yang diproses juga 0. Kemudian, hentikan konsumen untuk topik yang direplikasi pada cluster target. Pada titik ini, semua catatan yang direplikasi dari cluster MSK sumber untuk menargetkan klaster MSK telah dikonsumsi.
2. Mulai konsumen untuk topik lokal (misalnya,`topic`) pada cluster MSK target.

Melakukan failover yang tidak direncanakan ke Wilayah sekunder AWS

Anda dapat melakukan failover yang tidak direncanakan ketika ada acara layanan di AWS Wilayah utama yang memiliki kluster MSK sumber Anda dan Anda ingin mengalihkan sementara lalu lintas Anda ke AWS Wilayah sekunder yang memiliki kluster MSK target Anda. Failover yang tidak direncanakan dapat mengakibatkan beberapa kehilangan data.

1. Mencoba untuk menutup semua produsen dan konsumen yang terhubung ke cluster MSK sumber di Wilayah utama. Ini mungkin gagal.
2. Mulai produsen yang terhubung ke cluster MSK target di Wilayah sekunder.
3. Bergantung pada persyaratan pemesanan pesan aplikasi Anda, ikuti langkah-langkah di salah satu tab berikut.

No message ordering

Jika aplikasi Anda tidak memerlukan pemesanan pesan, mulailah konsumen di AWS Wilayah target yang membaca dari topik lokal (misalnya,`topic`) dan topik yang direplikasi (misalnya,`<sourceKafkaClusterAlias>.topic`) menggunakan operator wildcard (misalnya,`.*topic`).

Message ordering

1. Mulai konsumen hanya untuk topik yang direplikasi pada cluster target (misalnya,`<sourceKafkaClusterAlias>.topic`) tetapi bukan topik lokal (misalnya,`topic`).
2. Tunggu semua konsumen topik yang direplikasi pada cluster MSK target untuk menyelesaikan pemrosesan semua data, sehingga offset lag adalah 0 dan jumlah catatan yang diproses juga 0. Kemudian, hentikan konsumen untuk topik yang direplikasi pada cluster target. Pada titik ini, semua catatan yang direplikasi dari cluster MSK sumber untuk menargetkan klaster MSK telah dikonsumsi.
3. Mulai konsumen untuk topik lokal (misalnya,`topic`) pada cluster MSK target.

4. Setelah acara layanan berakhir di Wilayah utama, buat Replikator MSK baru untuk mereplikasi data dari klaster MSK Anda di Wilayah sekunder ke kluster MSK Anda di Wilayah primer dengan posisi awal Replicator disetel ke paling awal. Ini diperlukan untuk menyalin data yang akan Anda tulis ke Wilayah sekunder kembali ke Wilayah utama sehingga Anda dapat gagal kembali ke Wilayah utama setelah acara layanan berakhir. Jika Anda tidak menyetel posisi awal Replicator ke yang paling awal, data apa pun yang Anda hasilkan ke cluster di wilayah sekunder selama peristiwa layanan di wilayah primer tidak akan disalin kembali ke cluster di wilayah primer.

Melakukan kegagalan kembali ke Wilayah utama AWS

Anda dapat gagal kembali ke AWS wilayah utama setelah acara layanan di wilayah tersebut berakhir. MSK Replicator secara otomatis melewatkan topik yang memiliki alias cluster sumber sebagai awalan saat mereplikasi data kembali ke Wilayah utama selama failback.

Jika Anda mengikuti langkah-langkah failover yang tidak direncanakan, Anda seharusnya sudah membuat Replikator failback sebagai bagian dari langkah terakhir kegagalan dari wilayah primer ke sekunder.

Jika Anda tidak mengikuti langkah-langkah failover yang tidak direncanakan, setelah peristiwa layanan berakhir di Wilayah utama, buat Replikator MSK baru untuk mereplikasi data dari klaster MSK Anda di Wilayah sekunder ke kluster MSK Anda di Wilayah primer dengan posisi awal Replicator disetel ke yang paling awal. Ini diperlukan untuk menyalin data yang akan Anda tulis ke Wilayah sekunder kembali ke Wilayah utama sehingga Anda dapat gagal kembali ke Wilayah utama setelah acara layanan berakhir. Jika Anda tidak mengubah posisi awal Replicator dari nilai default terbaru ke yang paling awal, data apa pun yang Anda hasilkan ke cluster di wilayah sekunder selama peristiwa layanan di wilayah primer tidak akan disalin kembali ke cluster di wilayah primer.

Anda harus memulai langkah failback hanya setelah replikasi dari cluster di Region sekunder ke cluster di Region primer telah menyusul dan MessageLag metrik di mendekati CloudWatch 0. Failback yang direncanakan seharusnya tidak mengakibatkan kehilangan data.

1. Matikan semua produsen dan konsumen yang terhubung ke cluster MSK di Wilayah sekunder.
2. Untuk topologi aktif-pasif, hapus Replicator yang mereplikasi data dari cluster di Region sekunder ke Region primer. Anda tidak perlu menghapus Replicator untuk topologi aktif-aktif.
3. Mulai produsen yang terhubung ke cluster MSK di Wilayah utama.
4. Bergantung pada persyaratan pemesanan pesan aplikasi Anda, ikuti langkah-langkah di salah satu tab berikut.

   No message ordering

   Jika aplikasi Anda tidak memerlukan pemesanan pesan, mulailah konsumen di AWS Wilayah utama yang membaca dari topik lokal (misalnya,`topic`) dan topik yang direplikasi (misalnya,`<sourceKafkaClusterAlias>.topic`) menggunakan operator wildcard (misalnya,`.*topic`). Konsumen pada topik lokal (misalnya: topik) akan melanjutkan dari offset terakhir yang mereka konsumsi sebelum failover. Jika ada data yang belum diproses dari sebelum failover, itu akan diproses sekarang. Dalam kasus failover yang direncanakan, seharusnya tidak ada catatan seperti itu.

   Message ordering

   1. Mulai konsumen hanya untuk topik yang direplikasi di Wilayah primer (misalnya,`<sourceKafkaClusterAlias>.topic`) tetapi bukan topik lokal (misalnya,`topic`).
   2. Tunggu semua konsumen topik yang direplikasi pada cluster di Wilayah primer untuk menyelesaikan pemrosesan semua data, sehingga offset lag adalah 0 dan jumlah catatan yang diproses juga 0. Kemudian, hentikan konsumen untuk topik yang direplikasi

pada cluster di Wilayah utama. Pada titik ini, semua catatan yang diproduksi di Wilayah sekunder setelah failover telah dikonsumsi di Wilayah primer.

3. Mulai konsumen untuk topik lokal (misalnya,`topic`) pada cluster di Wilayah utama.

5. Verifikasi bahwa Replicator yang ada dari cluster di primer ke cluster di Region sekunder berada dalam status RUNNING dan berfungsi seperti yang diharapkan menggunakan metrik `ReplicatorThroughput` dan latensi.

Membuat pengaturan aktif-aktif menggunakan MSK Replicator

Ikuti langkah-langkah berikut untuk menyiapkan topologi aktif-aktif antara kluster MSK sumber A dan kluster MSK target B.

1. Buat Replikator MSK dengan MSK cluster A sebagai sumber dan MSK cluster B sebagai target.
2. Setelah MSK Replicator di atas berhasil dibuat, buat Replicator dengan cluster B sebagai sumber dan cluster A sebagai target.
3. Buat dua set produsen, masing-masing menulis data pada saat yang sama ke dalam topik lokal (misalnya, “topik”) di cluster di wilayah yang sama dengan produser.
4. Buat dua set konsumen, masing-masing membaca data menggunakan langganan wildcard (seperti”. *topic”) dari cluster MSK di AWS Wilayah yang sama dengan konsumen. Dengan cara ini konsumen Anda akan secara otomatis membaca data yang dihasilkan secara lokal di Wilayah dari topik lokal (misalnya,`topic`), serta data yang direplikasi dari Wilayah lain dalam topik dengan awalan`<sourceKafkaClusterAlias>.topic`). Kedua set konsumen ini harus memiliki ID grup konsumen yang berbeda sehingga offset grup konsumen tidak ditimpa ketika MSK Replicator menyalinnya ke cluster lain.

Pemecahan Masalah MSK Replicator

Topik

- Latensi replikasi tinggi atau terus meningkat

Informasi berikut dapat membantu Anda memecahkan masalah yang mungkin Anda miliki dengan MSK Replicator. Anda juga dapat memposting masalah Anda ke AWS re:Post.

Status MSK Replicator berubah dari CREATING menjadi FAILED

Berikut adalah beberapa penyebab umum kegagalan pembuatan MSK Replicator.

1. Verifikasi bahwa grup keamanan yang Anda berikan untuk pembuatan Replicator di bagian klaster Target memiliki aturan keluar untuk mengizinkan lalu lintas ke grup keamanan kluster target Anda. Juga verifikasi bahwa grup keamanan kluster target Anda memiliki aturan masuk yang menerima lalu lintas dari grup keamanan yang Anda sediakan untuk pembuatan Replicator di bagian klaster Target. Lihat Pilih klaster target Anda.
2. Jika Anda membuat Replicator untuk replikasi lintas wilayah, verifikasi bahwa cluster sumber Anda telah mengaktifkan konektivitas multi-VPC untuk metode autentikasi Kontrol Akses IAM. Lihat Amazon MSK Multi-VPC konektivitas pribadi dalam satu Wilayah. Juga verifikasi bahwa kebijakan cluster diatur pada cluster sumber sehingga MSK Replicator dapat terhubung ke cluster sumber. Lihat Langkah 1: Siapkan cluster sumber MSK Amazon.
3. Verifikasi bahwa peran IAM yang Anda berikan selama pembuatan MSK Replicator memiliki izin yang diperlukan untuk membaca dan menulis ke sumber dan kluster target Anda. Juga, verifikasi bahwa peran IAM memiliki izin untuk menulis ke topik. Lihat Konfigurasikan pengaturan dan izin replikator
4. Verifikasi bahwa ACL jaringan Anda tidak memblokir koneksi antara Replikator MSK dan kluster sumber dan target Anda.
5. Ada kemungkinan bahwa sumber atau cluster target tidak sepenuhnya tersedia ketika MSK Replicator mencoba untuk terhubung ke mereka. Ini mungkin karena beban yang berlebihan, penggunaan disk atau penggunaan CPU, yang menyebabkan Replicator tidak dapat terhubung ke broker. Perbaiki masalah dengan broker dan coba lagi pembuatan Replicator.

Setelah Anda melakukan validasi di atas, buat MSK Replicator lagi.

MSK Replicator tampak macet dalam status CREATING

Terkadang pembuatan MSK Replicator dapat memakan waktu hingga 30 menit. Tunggu selama 30 menit dan periksa status Replicator lagi.

MSK Replicator tidak mereplikasi data atau mereplikasi hanya sebagian data

Ikuti langkah-langkah ini untuk memecahkan masalah replikasi data.

1. Verifikasi bahwa Replicator Anda tidak mengalami kesalahan otentikasi menggunakan AuthError metrik yang disediakan oleh MSK Replicator di. CloudWatch Jika metrik ini di atas 0, periksa apakah kebijakan peran IAM yang Anda berikan untuk replikator valid dan tidak ada izin penolakan yang ditetapkan untuk izin klaster. Berdasarkan dimensi ClusterAlias, Anda dapat mengidentifikasi apakah sumber atau kluster target mengalami kesalahan otentikasi.
2. Verifikasi bahwa sumber dan kluster target Anda tidak mengalami masalah apa pun. Ada kemungkinan bahwa Replicator tidak dapat terhubung ke sumber atau cluster target Anda. Ini mungkin terjadi karena terlalu banyak koneksi, disk pada kapasitas penuh atau penggunaan CPU yang tinggi.
3. Verifikasi bahwa sumber dan kluster target Anda dapat dijangkau dari MSK Replicator menggunakan metrik di. KafkaClusterPingSuccessCount CloudWatch Berdasarkan dimensi ClusterAlias, Anda dapat mengidentifikasi apakah sumber atau kluster target mengalami kesalahan autentikasi. Jika metrik ini 0 atau tidak memiliki titik data, koneksi tidak sehat. Anda harus memeriksa izin peran jaringan dan IAM yang digunakan MSK Replicator untuk terhubung ke cluster Anda.
4. Verifikasi bahwa Replicator Anda tidak mengalami kegagalan karena izin tingkat topik yang hilang menggunakan metrik di. ReplicatorFailure CloudWatch Jika metrik ini di atas 0, periksa peran IAM yang Anda berikan untuk izin tingkat topik.
5. Verifikasi bahwa ekspresi reguler yang Anda berikan dalam daftar izinkan saat membuat Replikator cocok dengan nama topik yang ingin Anda tiru. Juga, verifikasi bahwa topik tidak dikecualikan dari replikasi karena ekspresi reguler dalam daftar penolakan.
6. Perhatikan bahwa mungkin diperlukan waktu hingga 30 detik bagi Replicator untuk mendeteksi dan membuat topik baru atau partisi topik pada cluster target. Setiap pesan yang dihasilkan ke topik sumber sebelum topik dibuat di kluster target tidak akan direplikasi jika posisi awal replikator terbaru (default). Atau, Anda dapat memulai replikasi dari offset paling awal di partisi topik cluster sumber jika Anda ingin mereplikasi pesan yang ada pada topik Anda di kluster target. Lihat Konfigurasikan pengaturan dan izin replikator.

Offset pesan di cluster target berbeda dari cluster sumber

Sebagai bagian dari mereplikasi data, MSK Replicator mengkonsumsi pesan dari cluster sumber dan memproduksinya ke cluster target. Hal ini dapat menyebabkan pesan memiliki offset yang berbeda pada sumber dan kluster target Anda. Namun, jika Anda telah mengaktifkan sinkronisasi offset grup konsumen selama pembuatan Replicator, MSK Replicator akan secara otomatis menerjemahkan offset saat menyalin metadata sehingga setelah gagal ke cluster target, konsumen Anda dapat melanjutkan pemrosesan dari dekat tempat mereka tinggalkan di cluster sumber.

MSK Replicator tidak menyinkronkan offset grup konsumen atau grup konsumen tidak ada pada cluster target

Ikuti langkah-langkah ini untuk memecahkan masalah replikasi metadata.

1. Verifikasi bahwa replikasi data Anda berfungsi seperti yang diharapkan. Jika belum, lihat MSK Replicator tidak mereplikasi data atau mereplikasi hanya sebagian data.
2. Verifikasi bahwa ekspresi reguler yang Anda berikan dalam daftar izinkan saat membuat Replikator cocok dengan nama grup konsumen yang ingin Anda tiru. Juga, verifikasi bahwa kelompok konsumen tidak dikecualikan dari replikasi karena ekspresi reguler dalam daftar penolakan.
3. Verifikasi bahwa MSK Replicator telah membuat topik pada cluster target. Diperlukan waktu hingga 30 detik bagi Replicator untuk mendeteksi dan membuat topik atau partisi topik baru pada cluster target. Setiap pesan yang dihasilkan ke topik sumber sebelum topik dibuat di kluster target tidak akan direplikasi jika posisi awal replikator terbaru (default). Jika kelompok konsumen Anda di cluster sumber hanya mengkonsumsi mesages yang belum direplikasi oleh MSK Replicator, grup konsumen tidak akan direplikasi ke cluster target. Setelah topik berhasil dibuat di cluster target, MSK Replicator akan mulai mereplikasi pesan yang baru ditulis di cluster sumber ke target. Setelah grup konsumen Anda mulai membaca pesan-pesan ini dari sumbernya, MSK Replicator akan secara otomatis mereplikasi grup konsumen ke cluster target. Atau, Anda dapat memulai replikasi dari offset paling awal di partisi topik cluster sumber jika Anda ingin mereplikasi pesan yang ada pada topik Anda di kluster target. Lihat Konfigurasikan pengaturan dan izin replikator.

> **Note**
>
> MSK Replicator mengoptimalkan sinkronisasi offset grup konsumen untuk konsumen Anda di cluster sumber yang membaca dari posisi yang lebih dekat ke akhir partisi topik. Jika

grup konsumen Anda tertinggal di cluster sumber, Anda mungkin melihat lag yang lebih tinggi untuk kelompok konsumen pada target dibandingkan dengan sumbernya. Ini berarti setelah failover ke cluster target, konsumen Anda akan memproses ulang lebih banyak pesan duplikat. Untuk mengurangi lag ini, konsumen Anda di cluster sumber perlu mengejar ketinggalan dan mulai mengkonsumsi dari ujung aliran (akhir partisi topik). Saat konsumen Anda mengejar ketinggalan, MSK Replicator akan secara otomatis mengurangi lag.

Latensi replikasi tinggi atau terus meningkat

Berikut adalah beberapa penyebab umum latensi replikasi yang tinggi.

1. Verifikasi bahwa Anda memiliki jumlah partisi yang tepat pada sumber dan target kluster MSK Anda. Memiliki terlalu sedikit atau terlalu banyak partisi dapat memengaruhi kinerja. Untuk panduan memilih jumlah partisi, lihatPraktik terbaik untuk menggunakan MSK Replicator. Tabel berikut menunjukkan jumlah minimum partisi yang disarankan untuk mendapatkan throughput yang Anda inginkan dengan MSK Replicator.

   Throughput dan jumlah minimum partisi yang disarankan

Throughput (MB/s)	Jumlah minimum partisi yang diperlukan
50	167
100	334
250	833
500	1666
1000	3333

2. Verifikasi bahwa Anda memiliki kapasitas baca dan tulis yang cukup di sumber dan target kluster MSK Anda untuk mendukung lalu lintas replikasi. MSK Replicator bertindak sebagai konsumen untuk cluster sumber Anda (jalan keluar) dan sebagai produsen untuk cluster target Anda (ingress). Oleh karena itu, Anda harus menyediakan kapasitas cluster untuk mendukung lalu lintas replikasi serta lalu lintas lain di cluster Anda. Lihat ??? panduan tentang ukuran kluster MSK Anda.
3. Latensi replikasi dapat bervariasi untuk kluster MSK di pasangan AWS Wilayah sumber dan tujuan yang berbeda, tergantung pada seberapa jauh jarak cluster secara geografis satu sama lain.

Misalnya, latensi replikasi biasanya lebih rendah ketika mereplikasi antara kluster di Wilayah Eropa (Irlandia) dan Eropa (London) dibandingkan dengan replikasi antara kluster di Wilayah Eropa (Irlandia) dan Asia Pasifik (Sydney).

4. Verifikasi bahwa Replicator Anda tidak terhambat karena kuota terlalu agresif yang ditetapkan pada sumber atau kluster target Anda. Anda dapat menggunakan ThrottleTime metrik yang disediakan oleh MSK Replicator CloudWatch untuk melihat waktu rata-rata dalam milidetik permintaan dibatasi oleh broker di cluster sumber/target Anda. Jika metrik ini di atas 0, Anda harus menyesuaikan kuota Kafka untuk mengurangi pelambatan sehingga Replicator dapat mengejar ketinggalan. Lihat Mengelola throughput MSK Replicator menggunakan kuota Kafka untuk informasi tentang mengelola kuota Kafka untuk Replicator.
5. ReplicationLatency dan MessageLag mungkin meningkat ketika suatu AWS Wilayah menjadi terdegradasi. Gunakan AWS Service Health Dashboard untuk memeriksa acara layanan MSK di Wilayah tempat klaster MSK utama Anda berada. Jika ada acara layanan, Anda dapat mengalihkan sementara aplikasi Anda membaca dan menulis ke Wilayah lain.

Praktik terbaik untuk menggunakan MSK Replicator

Bagian ini mencakup praktik terbaik umum dan strategi implementasi untuk menggunakan &MSK; Replicator.

Topik

- Mengelola throughput MSK Replicator menggunakan kuota Kafka
- Mengatur periode retensi cluster

Mengelola throughput MSK Replicator menggunakan kuota Kafka

Karena MSK Replicator bertindak sebagai konsumen untuk cluster sumber Anda, replikasi dapat menyebabkan konsumen lain terhambat pada cluster sumber Anda. Jumlah throttling tergantung pada kapasitas baca yang Anda miliki di cluster sumber Anda dan throughput data yang Anda replikasi. Kami menyarankan agar penyediaan Anda memiliki kapasitas yang sama untuk cluster sumber dan target Anda, dan memperhitungkan throughput replikasi saat menghitung berapa banyak kapasitas yang Anda butuhkan.

Anda juga dapat mengatur kuota Kafka untuk Replicator pada sumber dan kluster target Anda untuk mengontrol berapa banyak kapasitas yang dapat digunakan oleh Replikator MSK. Direkomendasikan kuota bandwidth jaringan. Kuota bandwidth jaringan mendefinisikan ambang batas byte, didefinisikan

sebagai byte per detik, untuk satu atau lebih klien yang berbagi kuota. Kuota ini didefinisikan berdasarkan per-broker.

Ikuti langkah-langkah berikut untuk menerapkan kuota.

1. Ambil string server bootstrap untuk cluster sumber. Lihat Mendapatkan broker bootstrap untuk cluster MSK Amazon.
2. Ambil peran eksekusi layanan (SER) yang digunakan oleh MSK Replicator. Ini adalah SER yang Anda gunakan untuk `CreateReplicator` permintaan. Anda juga dapat menarik SER dari DescribeReplicator respons dari Replicator yang ada.
3. Menggunakan alat Kafka CLI, jalankan perintah berikut terhadap cluster sumber.

```
./kafka-configs.sh --bootstrap-server <source-cluster-bootstrap-server> --alter --
add-config 'consumer_byte_
rate=<quota_in_bytes_per_second>' --entity-type users --entity-name
 arn:aws:sts::<customer-account-id>:assumed-role/<ser-role-name>/<customer-account-
id> --command-config <client-properties-for-iam-auth></programlisting>
```

4. Setelah menjalankan perintah di atas, verifikasi bahwa `ReplicatorThroughput` metrik tidak melewati kuota yang telah Anda tetapkan.

Perhatikan bahwa jika Anda menggunakan kembali peran eksekusi layanan antara beberapa Replikator MSK, semuanya tunduk pada kuota ini. Jika Anda ingin mempertahankan kuota terpisah per Replicator, gunakan peran eksekusi layanan terpisah.

Untuk informasi lebih lanjut tentang menggunakan otentikasi MSK IAM dengan kuota, lihat Multi-tenancy Apache Kafka cluster di Amazon MSK dengan kontrol akses IAM dan Kafka Quota - Bagian 1.

> **Warning**
>
> Menyetel consumer_byte_rate yang sangat rendah dapat menyebabkan Replikator MSK Anda bertindak dengan cara yang tidak terduga.

Mengatur periode retensi cluster

Anda dapat mengatur periode retensi log untuk kluster yang disediakan MSK dan tanpa server. Periode retensi yang disarankan adalah 7 hari. Lihat Perubahan konfigurasi cluster atau MSK Konfigurasi klaster tanpa server.

Negara cluster

Tabel berikut menunjukkan kemungkinan keadaan cluster dan menjelaskan apa artinya. Ini juga menjelaskan tindakan apa yang dapat dan tidak dapat Anda lakukan ketika cluster berada di salah satu status ini. Untuk mengetahui keadaan cluster, Anda dapat mengunjungi AWS Management Console. Anda juga dapat menggunakan perintah describe-cluster-v2 atau operasi DescribeClusterV2 untuk menggambarkan cluster. Deskripsi cluster mencakup keadaannya.

Status cluster	Makna dan kemungkinan tindakan
AKTIF	Anda dapat menghasilkan dan mengkonsu msi data. Anda juga dapat melakukan Amazon MSK API dan AWS CLI operasi di cluster.
CREATING	Amazon MSK sedang menyiapkan cluster. Anda harus menunggu klaster mencapai status ACTIVE sebelum Anda dapat menggunak annya untuk menghasilkan atau menggunakan data atau untuk menjalankan Amazon MSK API atau AWS CLI operasi di dalamnya.
DELETING	Klaster sedang dihapus. Anda tidak dapat menggunakannya untuk menghasilkan atau mengkonsumsi data. Anda juga tidak dapat melakukan Amazon MSK API atau AWS CLI operasi di atasnya.
FAILED	Proses pembuatan atau penghapusan klaster gagal. Anda tidak dapat menggunakan cluster untuk menghasilkan atau mengkonsumsi data. Anda dapat menghapus cluster tetapi tidak dapat melakukan Amazon MSK API atau AWS CLI memperbarui operasi di dalamnya.
SEMBUH	Amazon MSK menjalankan operasi internal, seperti mengganti broker yang tidak sehat. Misalnya, broker mungkin tidak responsif

Status cluster	Makna dan kemungkinan tindakan
	. Anda masih dapat menggunakan cluster untuk memproduksi dan mengkonsumsi data. Namun, Anda tidak dapat menjalankan Amazon MSK API atau AWS CLI memperbarui operasi di cluster hingga kembali ke status AKTIF.
PERAWATAN	Amazon MSK melakukan operasi pemeliharaan rutin di cluster. Operasi pemeliharaan tersebut termasuk penambalan keamanan. Anda masih dapat menggunakan cluster untuk memproduksi dan mengkonsumsi data. Namun, Anda tidak dapat menjalankan Amazon MSK API atau AWS CLI memperbarui operasi di cluster hingga kembali ke status AKTIF.
REBOOTING_BROKER	Amazon MSK me-reboot broker. Anda masih dapat menggunakan cluster untuk memproduksi dan mengkonsumsi data. Namun, Anda tidak dapat menjalankan Amazon MSK API atau AWS CLI memperbarui operasi di cluster hingga kembali ke status AKTIF.
UPDATING	API AWS CLI atau operasi MSK Amazon yang diprakarsai pengguna memperbarui cluster. Anda masih dapat menggunakan cluster untuk memproduksi dan mengkonsumsi data. Namun, Anda tidak dapat melakukan API MSK Amazon tambahan atau operasi AWS CLI pembaruan di klaster hingga kembali ke status AKTIF.

Keamanan di Amazon Managed Streaming for Apache Kafka

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 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 Kepatuhan . Untuk mempelajari tentang program kepatuhan yang berlaku untuk Amazon Managed Streaming for Apache Kafka, lihat Amazon Web Services in Scope by Compliance Program Amazon Web Services in Scope by Compliance Program by Compliance Program.
- 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 Amazon MSK. Topik berikut menunjukkan cara mengonfigurasi Amazon MSK untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga mempelajari cara menggunakan Amazon Web Services lain yang membantu Anda memantau dan mengamankan sumber daya MSK Amazon Anda.

Topik

- Validasi kepatuhan untuk Amazon Managed Streaming for Apache Kafka
- Ketahanan di Amazon Managed Streaming untuk Apache Kafka Kafka
- Keamanan infrastruktur di Amazon Managed Streaming for Apache Kafka

Perlindungan data di Amazon Managed Streaming for Apache Kafka

Model tanggung jawab AWS bersama model berlaku untuk perlindungan data di Amazon Managed Streaming for Apache Kafka. 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 kredensyal 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 sumber daya. AWS Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan logging aktivitas pengguna dengan AWS CloudTrail.
- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-2 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi yang lebih lengkap tentang titik akhir FIPS yang tersedia di Standar Pemrosesan Informasi Federal (FIPS) 140-2.

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 Amazon MSK atau lainnya Layanan AWS menggunakan konsol, API AWS CLI, atau AWS SDK. 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.

Topik

Enkripsi MSK Amazon

Amazon MSK menyediakan opsi enkripsi data yang dapat Anda gunakan untuk memenuhi persyaratan manajemen data yang ketat. Sertifikat yang digunakan Amazon MSK untuk enkripsi harus diperbarui setiap 13 bulan. Amazon MSK secara otomatis memperbarui sertifikat ini untuk semua cluster. Ini menetapkan status cluster `MAINTENANCE` saat memulai operasi pembaruan sertifikat. Ini mengaturnya kembali ke `ACTIVE` saat pembaruan selesai. Saat klaster berada dalam `MAINTENANCE` status, Anda dapat terus memproduksi dan mengkonsumsi data, tetapi Anda tidak dapat melakukan operasi pembaruan apa pun di dalamnya.

Enkripsi diam

Amazon MSK terintegrasi dengan AWS Key Management Service(KMS) untuk menawarkan enkripsi sisi server yang transparan. Amazon MSK selalu mengenkripsi data Anda saat istirahat. Saat Anda membuat klaster MSK, Anda dapat menentukan AWS KMS key yang Anda ingin Amazon MSK gunakan untuk mengenkripsi data Anda saat istirahat. Jika Anda tidak menentukan kunci KMS, Amazon MSK membuat Kunci yang dikelola AWSuntuk Anda dan menggunakannya atas nama Anda. Untuk informasi selengkapnya tentang kunci KMS, lihat AWS KMS keys dalam Panduan Developer AWS Key Management Service .

Enkripsi bergerak

Amazon MSK menggunakan TLS 1.2. Secara default, ini mengenkripsi data dalam perjalanan antara broker cluster MSK Anda. Anda dapat mengganti default ini pada saat Anda membuat cluster.

Untuk komunikasi antara klien dan broker, Anda harus menentukan salah satu dari tiga pengaturan berikut:

- Hanya izinkan data terenkripsi TLS. Ini adalah pengaturan default.
- Izinkan plaintext, serta data terenkripsi TLS.
- Hanya izinkan data plaintext.

Broker MSK Amazon menggunakan AWS Certificate Manager sertifikat publik. Oleh karena itu, setiap truststore yang mempercayai Amazon Trust Services juga mempercayai sertifikat broker MSK Amazon.

Meskipun kami sangat menyarankan untuk mengaktifkan enkripsi dalam transit, ini dapat menambahkan overhead CPU tambahan dan latensi beberapa milidetik. Namun, sebagian besar kasus penggunaan tidak sensitif terhadap perbedaan ini, dan besarnya dampaknya bergantung pada konfigurasi klaster, klien, dan profil penggunaan Anda.

Bagaimana cara memulai dengan enkripsi?

Saat membuat cluster MSK, Anda dapat menentukan pengaturan enkripsi dalam format JSON. Berikut adalah contohnya.

```
{
   "EncryptionAtRest": {
       "DataVolumeKMSKeyId": "arn:aws:kms:us-east-1:123456789012:key/abcdabcd-1234-
abcd-1234-abcd123e8e8e"
    },
   "EncryptionInTransit": {
        "InCluster": true,
        "ClientBroker": "TLS"
    }
}
```

Untuk`DataVolumeKMSKeyId`, Anda dapat menentukan kunci yang dikelola pelanggan atau MSK Kunci yang dikelola AWS untuk di akun Anda (`alias/aws/kafka`). Jika Anda tidak menentukan`EncryptionAtRest`, Amazon MSK masih mengenkripsi data Anda saat istirahat di bawah. Kunci yang dikelola AWS Untuk menentukan kunci mana yang digunakan klaster Anda, kirim `GET` permintaan atau panggil operasi `DescribeCluster` API.

Untuk`EncryptionInTransit`, nilai default `InCluster` adalah true, tetapi Anda dapat mengaturnya ke false jika Anda tidak ingin Amazon MSK mengenkripsi data Anda saat melewati antara broker.

Untuk menentukan mode enkripsi untuk data dalam perjalanan antara klien dan broker, atur `ClientBroker` ke salah satu dari tiga nilai:`TLS`,`TLS_PLAINTEXT`, atau`PLAINTEXT`.

Untuk menentukan pengaturan enkripsi saat membuat cluster

1. Simpan isi dari contoh sebelumnya dalam file dan berikan file nama apa pun yang Anda inginkan. Misalnya, sebut saja`encryption-settings.json`.
2. Jalankan `create-cluster` perintah dan gunakan `encryption-info` opsi untuk menunjuk ke file tempat Anda menyimpan konfigurasi JSON Anda. Berikut adalah contohnya. Ganti *`{YOUR MSK VERSION}`* dengan versi yang cocok dengan versi klien Apache Kafka. Untuk informasi tentang cara menemukan versi klaster MSK Anda, lihatTo find the version of your MSK cluster. Ketahuilah bahwa menggunakan versi klien Apache Kafka yang tidak sama dengan versi cluster MSK Anda dapat menyebabkan kerusakan, kehilangan, dan waktu henti data Apache Kafka.

   ```
   aws kafka create-cluster --cluster-name "ExampleClusterName" --broker-node-group-
   info file://brokernodegroupinfo.json --encryption-info file://encryptioninfo.json
    --kafka-version "{YOUR MSK VERSION}" --number-of-broker-nodes 3
   ```

   Berikut ini adalah contoh respons yang berhasil setelah menjalankan perintah ini.

   ```
   {
       "ClusterArn": "arn:aws:kafka:us-east-1:123456789012:cluster/SecondTLSTest/
   abcdabcd-1234-abcd-1234-abcd123e8e8e",
       "ClusterName": "ExampleClusterName",
       "State": "CREATING"
   }
   ```

Untuk menguji enkripsi TLS

1. Buat mesin klien mengikuti panduan dithe section called “Langkah 3: Buat mesin klien”.
2. Instal Apache Kafka di mesin klien.
3. Dalam contoh ini kita menggunakan truststore JVM untuk berbicara dengan cluster MSK. Untuk melakukan ini, pertama buat folder bernama `/tmp` pada mesin klien. Kemudian, buka `bin` folder instalasi Apache Kafka, dan jalankan perintah berikut. (Jalur JVM Anda mungkin berbeda.)

```
cp /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.201.b09-0.amzn2.x86_64/jre/lib/security/
cacerts /tmp/kafka.client.truststore.jks
```

4. Saat masih dalam `bin` folder instalasi Apache Kafka di mesin klien, buat file teks bernama `client.properties` dengan konten berikut.

```
security.protocol=SSL
ssl.truststore.location=/tmp/kafka.client.truststore.jks
```

5. Jalankan perintah berikut pada mesin yang telah AWS CLI diinstal, menggantikan *`ClusterARN`* dengan ARN cluster Anda.

```
aws kafka get-bootstrap-brokers --cluster-arn clusterARN
```

Hasil yang sukses terlihat seperti berikut ini. Simpan hasil ini karena Anda membutuhkannya untuk langkah selanjutnya.

```
{
    "BootstrapBrokerStringTls": "a-1.example.g7oein.c2.kafka.us-
east-1.amazonaws.com:0123,a-3.example.g7oein.c2.kafka.us-
east-1.amazonaws.com:0123,a-2.example.g7oein.c2.kafka.us-east-1.amazonaws.com:0123"
}
```

6. Jalankan perintah berikut, ganti *`BootstrapBrokerStringTls`*dengan salah satu titik akhir broker yang Anda peroleh pada langkah sebelumnya.

```
<path-to-your-kafka-installation>/bin/kafka-console-producer.sh --broker-
list BootstrapBrokerStringTls --producer.config client.properties --topic
 TLSTestTopic
```

7. Buka jendela perintah baru dan sambungkan ke mesin klien yang sama. Kemudian, jalankan perintah berikut untuk membuat konsumen konsol.

```
<path-to-your-kafka-installation>/bin/kafka-console-consumer.sh --bootstrap-
server BootstrapBrokerStringTls --consumer.config client.properties --topic
 TLSTestTopic
```

8. Di jendela produser, ketik pesan teks diikuti dengan pengembalian, dan cari pesan yang sama di jendela konsumen. Amazon MSK mengenkripsi pesan ini dalam perjalanan.

Untuk informasi selengkapnya tentang mengonfigurasi klien Apache Kafka agar bekerja dengan data terenkripsi, lihat Mengonfigurasi Klien Kafka.

Otentikasi dan otorisasi untuk Amazon MSK API

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 MSK Amazon. IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Halaman ini menjelaskan bagaimana Anda dapat menggunakan IAM untuk mengontrol siapa yang dapat melakukan operasi MSK Amazon di klaster Anda. Untuk informasi tentang cara mengontrol siapa yang dapat melakukan operasi Apache Kafka di cluster Anda, lihat. the section called "Otentikasi dan otorisasi untuk Apache Kafka API"

Topik

- Bagaimana Amazon MSK bekerja dengan IAM
- Contoh kebijakan berbasis identitas MSK Amazon
- Menggunakan peran terkait layanan untuk Amazon MSK
- AWS kebijakan terkelola untuk Amazon MSK
- Memecahkan masalah identitas dan akses MSK Amazon

Bagaimana Amazon MSK bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ke Amazon MSK, Anda harus memahami fitur IAM apa yang tersedia untuk digunakan dengan Amazon MSK. Untuk mendapatkan tampilan tingkat tinggi tentang cara Amazon MSK dan AWS layanan lainnya bekerja dengan IAM, lihat AWS Layanan yang Bekerja dengan IAM di Panduan Pengguna IAM.

Topik

- Kebijakan berbasis identitas MSK Amazon
- Kebijakan berbasis sumber daya Amazon MSK
- AWS kebijakan terkelola
- Otorisasi berdasarkan tag MSK Amazon

- Peran Amazon MSK IAM

Kebijakan berbasis identitas MSK Amazon

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. Amazon MSK mendukung tindakan, sumber daya, dan kunci kondisi tertentu. Untuk mempelajari semua elemen yang Anda gunakan dalam kebijakan JSON, lihat Referensi Elemen Kebijakan JSON IAM dalam Panduan Pengguna IAM.

Tindakan

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama dengan operasi AWS API terkait. Ada beberapa pengecualian, misalnya tindakan hanya izin yang tidak memiliki operasi API yang cocok. Ada juga beberapa operasi yang memerlukan beberapa tindakan dalam suatu kebijakan. Tindakan tambahan ini disebut tindakan dependen.

Menyertakan tindakan dalam kebijakan untuk memberikan izin untuk melakukan operasi terkait.

Tindakan kebijakan di Amazon MSK menggunakan awalan berikut sebelum tindakan:. `kafka:` Misalnya, untuk memberikan izin kepada seseorang untuk mendeskripsikan klaster MSK dengan operasi Amazon MSK `DescribeCluster` API, Anda menyertakan `kafka:DescribeCluster` tindakan tersebut dalam kebijakan mereka. Pernyataan kebijakan harus memuat elemen `Action` atau `NotAction`. Amazon MSK mendefinisikan serangkaian tindakannya sendiri yang menggambarkan tugas yang dapat Anda lakukan dengan layanan ini.

Untuk menetapkan beberapa tindakan dalam satu pernyataan, pisahkan dengan koma seperti berikut:

```
"Action": ["kafka:action1", "kafka:action2"]
```

Anda dapat menentukan beberapa tindakan menggunakan wildcard (*). Sebagai contoh, untuk menentukan semua tindakan yang dimulai dengan kata `Describe`, sertakan tindakan berikut:

```
"Action": "kafka:Describe*"
```

Untuk melihat daftar tindakan MSK Amazon, lihat Tindakan, sumber daya, dan kunci kondisi untuk Amazon Managed Streaming for Apache Kafka di Panduan Pengguna IAM.

Sumber daya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek yang menjadi target penerapan tindakan. Pernyataan harus menyertakan elemen `Resource` atau `NotResource`. Praktik terbaiknya, tentukan sumber daya menggunakan Amazon Resource Name (ARN). Anda dapat melakukan ini untuk tindakan yang mendukung jenis sumber daya tertentu, yang dikenal sebagai izin tingkat sumber daya.

Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, misalnya operasi pencantuman, gunakan wildcard (*) untuk menunjukkan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

```
"Resource": "*"
```

Sumber daya instans MSK Amazon memiliki ARN berikut:

```
arn:${Partition}:kafka:${Region}:${Account}:cluster/${ClusterName}/${UUID}
```

Untuk informasi selengkapnya tentang format ARN, lihat Nama Sumber Daya Amazon (ARN) dan Ruang Nama AWS Layanan.

Misalnya, untuk menentukan instans `CustomerMessages` dalam pernyataan Anda, gunakan ARN berikut:

```
"Resource": "arn:aws:kafka:us-east-1:123456789012:cluster/CustomerMessages/abcd1234-
abcd-dcba-4321-a1b2abcd9f9f-2"
```

Untuk menentukan semua instans milik akun tertentu, gunakan wildcard (*):

```
"Resource": "arn:aws:kafka:us-east-1:123456789012:cluster/*"
```

Beberapa tindakan MSK Amazon, seperti untuk membuat sumber daya, tidak dapat dilakukan pada sumber daya tertentu. Dalam kasus tersebut, Anda harus menggunakan wildcard (*).

```
"Resource": "*"
```

Untuk menentukan beberapa sumber daya dalam satu pernyataan, pisahkan ARN dengan koma.

```
"Resource": ["resource1", "resource2"]
```

Untuk melihat daftar jenis sumber daya MSK Amazon dan ARNnya, lihat Sumber Daya yang Ditentukan oleh Amazon Managed Streaming for Apache Kafka di Panduan Pengguna IAM. Untuk mempelajari tindakan mana yang dapat Anda tentukan ARN dari setiap sumber daya, lihat Tindakan yang Ditentukan oleh Amazon Managed Streaming for Apache Kafka.

Kunci syarat

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Condition` (atau blok `Condition`) akan memungkinkan Anda menentukan kondisi yang menjadi dasar suatu pernyataan berlaku. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi bersyarat yang menggunakan operator kondisi, misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta.

Jika Anda menentukan beberapa elemen `Condition` dalam sebuah pernyataan, atau beberapa kunci dalam elemen `Condition` tunggal, maka AWS akan mengevaluasinya menggunakan operasi `AND` logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS mengevaluasi kondisi menggunakan `OR` operasi logis. Semua kondisi harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan kondisi. Sebagai contoh, Anda dapat memberikan izin kepada pengguna IAM untuk mengakses sumber daya hanya jika izin tersebut mempunyai tag yang sesuai dengan nama pengguna IAM mereka. Untuk informasi selengkapnya, lihat Elemen kebijakan IAM: variabel dan tag dalam Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi khusus layanan. Untuk melihat semua kunci kondisi AWS global, lihat kunci konteks kondisi AWS global di Panduan Pengguna IAM.

Amazon MSK mendefinisikan rangkaian kunci kondisinya sendiri dan juga mendukung penggunaan beberapa kunci kondisi global. Untuk melihat semua kunci kondisi AWS global, lihat Kunci Konteks Kondisi AWS Global di Panduan Pengguna IAM.

Untuk melihat daftar kunci kondisi MSK Amazon, lihat Kunci Kondisi untuk Amazon Managed Streaming for Apache Kafka di Panduan Pengguna IAM. Untuk mempelajari tindakan dan sumber daya yang dapat Anda gunakan kunci kondisi, lihat Tindakan yang Ditentukan oleh Amazon Managed Streaming for Apache Kafka.

Contoh

Untuk melihat contoh kebijakan berbasis identitas MSK Amazon, lihat. Contoh kebijakan berbasis identitas MSK Amazon

Kebijakan berbasis sumber daya Amazon MSK

Amazon MSK mendukung kebijakan klaster (juga dikenal sebagai kebijakan berbasis sumber daya) untuk digunakan dengan kluster MSK Amazon. Anda dapat menggunakan kebijakan klaster untuk menentukan prinsipal IAM mana yang memiliki izin lintas akun untuk menyiapkan konektivitas pribadi ke kluster MSK Amazon Anda. Saat digunakan dengan otentikasi klien IAM, Anda juga dapat menggunakan kebijakan klaster untuk secara terperinci menentukan izin bidang data Kafka untuk klien yang menghubungkan.

Untuk melihat contoh cara mengonfigurasi kebijakan klaster, lihatLangkah 2: Lampirkan kebijakan klaster ke klaster MSK.

AWS kebijakan terkelola

Otorisasi berdasarkan tag MSK Amazon

Anda dapat melampirkan tag ke cluster MSK Amazon. Untuk mengendalikan akses berdasarkan tag, berikan informasi tentang tag di elemen kondisi dari kebijakan menggunakan kunci kondisi `kafka:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, atau `aws:TagKeys`. Untuk informasi selengkapnya tentang menandai sumber daya MSK Amazon, lihat. the section called “Menandai klaster”

Untuk melihat contoh kebijakan berbasis identitas untuk membatasi akses ke klaster berdasarkan tag pada klaster tersebut, lihat. Mengakses kluster MSK Amazon berdasarkan tag

Peran Amazon MSK IAM

IAM role adalah entitas di dalam akun Amazon Web Services Anda yang memiliki izin khusus.

Menggunakan kredensi sementara dengan Amazon MSK

Anda dapat menggunakan kredensial sementara untuk masuk dengan gabungan, menjalankan IAM role, atau menjalankan peran lintas akun. Anda memperoleh kredensyal keamanan sementara dengan memanggil operasi AWS STS API seperti AssumeRoleatau GetFederation Token.

Amazon MSK mendukung penggunaan kredensil sementara.

Peran terkait layanan

Peran terkait layanan memungkinkan Amazon Web Services mengakses sumber daya di layanan lain untuk menyelesaikan tindakan atas nama Anda. Peran tertaut layanan muncul di akun IAM Anda dan dimiliki oleh layanan tersebut. Administrator dapat melihat tetapi tidak dapat mengedit izin untuk peran yang terkait dengan layanan.

Amazon MSK mendukung peran terkait layanan. Untuk detail tentang membuat atau mengelola peran terkait layanan MSK Amazon,. the section called “Peran terkait layanan”

Contoh kebijakan berbasis identitas MSK Amazon

Secara default, pengguna dan peran IAM tidak memiliki izin untuk menjalankan tindakan Amazon MSK API. Administrator harus membuat kebijakan IAM yang memberikan izin kepada pengguna dan peran untuk melakukan operasi API tertentu pada sumber daya tertentu yang mereka butuhkan. Administrator kemudian harus melampirkan kebijakan tersebut ke pengguna IAM atau grup yang memerlukan izin tersebut.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM menggunakan contoh dokumen kebijakan JSON ini, lihat Membuat Kebijakan pada Tab JSON dalam Panduan Pengguna IAM.

Topik

- Praktik terbaik kebijakan
- Mengizinkan pengguna melihat izin mereka sendiri
- Mengakses satu cluster MSK Amazon
- Mengakses kluster MSK Amazon berdasarkan tag

Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus sumber daya MSK Amazon di akun Anda. Tindakan ini membuat Akun AWS Anda dikenai biaya. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit — Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Anda Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola AWS pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat Kebijakan yang dikelola AWS atau Kebijakan yang dikelola AWS untuk fungsi tugas dalam Panduan Pengguna IAM.
- Menerapkan izin dengan hak akses paling rendah – Ketika Anda menetapkan izin dengan kebijakan IAM, hanya berikan izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk mengajukan izin, lihat Kebijakan dan izin dalam IAM dalam Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan jika digunakan melalui yang spesifik Layanan AWS, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat Elemen kebijakan JSON IAM: Kondisi dalam Panduan Pengguna IAM.
- Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional – IAM Access Analyzer memvalidasi kebijakan baru dan yang sudah ada sehingga kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM. IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, lihat Validasi kebijakan IAM Access Analyzer dalam Panduan Pengguna IAM.
- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Anda, Akun AWS aktifkan MFA untuk keamanan tambahan.

Untuk meminta MFA ketika operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi akses API yang dilindungi MFA dalam Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat Praktik terbaik keamanan dalam IAM dalam Panduan Pengguna IAM.

Mengizinkan pengguna melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan ini di konsol atau menggunakan API atau secara terprogram. AWS CLI AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

Mengakses satu cluster MSK Amazon

Dalam contoh ini, Anda ingin memberikan pengguna IAM di akun Amazon Web Services Anda akses ke salah satu cluster Anda. `purchaseQueriesCluster` Kebijakan ini memungkinkan pengguna untuk mendeskripsikan cluster, mendapatkan broker bootstrap, daftar node brokernya, dan memperbaruinya.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"UpdateCluster",
         "Effect":"Allow",
         "Action":[
            "kafka:Describe*",
            "kafka:Get*",
            "kafka:List*",
            "kafka:Update*"
         ],
         "Resource":"arn:aws:kafka:us-east-1:012345678012:cluster/
purchaseQueriesCluster/abcdefab-1234-abcd-5678-cdef0123ab01-2"
      }
   ]
}
```

Mengakses kluster MSK Amazon berdasarkan tag

Anda dapat menggunakan kondisi dalam kebijakan berbasis identitas untuk mengontrol akses ke sumber daya MSK Amazon berdasarkan tag. Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan yang memungkinkan pengguna untuk mendeskripsikan cluster, mendapatkan broker bootstrap, daftar node brokernya, memperbaruinya, dan menghapusnya. Namun, izin diberikan hanya jika tag cluster `Owner` memiliki nilai nama pengguna pengguna tersebut.

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

```
    {
      "Sid": "AccessClusterIfOwner",
      "Effect": "Allow",
      "Action": [
        "kafka:Describe*",
        "kafka:Get*",
        "kafka:List*",
        "kafka:Update*",
        "kafka:Delete*"
      ],
      "Resource": "arn:aws:kafka:us-east-1:012345678012:cluster/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Owner": "${aws:username}"
        }
      }
    }
  ]
}
```

Anda dapat melampirkan kebijakan ini ke pengguna IAM di akun Anda. Jika pengguna bernama `richard-roe` mencoba memperbarui kluster MSK, cluster harus diberi tag `Owner=richard-roe` atau. `owner=richard-roe` Jika tidak, aksesnya akan ditolak. Kunci tanda syarat `Owner` sama dengan kedua `Owner` dan `owner` karena nama kunci syarat tidak terpengaruh huruf besar/kecil. Untuk informasi lebih lanjut, lihat Elemen Kebijakan IAM JSON: Persyaratan dalam Panduan Pengguna IAM.

Menggunakan peran terkait layanan untuk Amazon MSK

Amazon MSK menggunakan peran terkait layanan AWS Identity and Access Management (IAM). Peran terkait layanan adalah jenis peran IAM unik yang ditautkan langsung ke Amazon MSK. Peran terkait layanan telah ditentukan sebelumnya oleh Amazon MSK dan mencakup semua izin yang diperlukan layanan untuk memanggil layanan lain AWS atas nama Anda.

Peran terkait layanan membuat pengaturan Amazon MSK lebih mudah karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Amazon MSK mendefinisikan izin dari peran terkait layanannya. Kecuali ditentukan lain, hanya Amazon MSK yang dapat mengambil perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Untuk informasi tentang layanan lain yang mendukung peran terkait layanan, lihat Amazon Web Services yang Bekerja dengan IAM, dan cari layanan yang memiliki Ya di kolom Peran Tertaut Layanan. Pilih Ya bersama tautan untuk melihat dokumentasi peran tertaut layanan untuk layanan tersebut.

Topik

- Izin peran terkait layanan untuk Amazon MSK
- Membuat peran terkait layanan untuk Amazon MSK
- Mengedit peran terkait layanan untuk Amazon MSK
- Wilayah yang Didukung untuk peran terkait layanan MSK Amazon

Izin peran terkait layanan untuk Amazon MSK

Amazon MSK menggunakan peran terkait layanan bernama. AWSServiceRoleForKafka Amazon MSK menggunakan peran ini untuk mengakses sumber daya Anda dan melakukan operasi seperti:

- `*NetworkInterface`— membuat dan mengelola antarmuka jaringan di akun pelanggan yang membuat broker cluster dapat diakses oleh klien di VPC pelanggan.
- `*VpcEndpoints`— mengelola titik akhir VPC di akun pelanggan yang membuat broker klaster dapat diakses oleh klien di VPC pelanggan yang menggunakan. AWS PrivateLink Amazon MSK menggunakan izin untuk`DescribeVpcEndpoints`, `ModifyVpcEndpoint` dan. `DeleteVpcEndpoints`
- `secretsmanager`— mengelola kredensi klien dengan. AWS Secrets Manager
- `GetCertificateAuthorityCertificate`— mengambil sertifikat untuk otoritas sertifikat pribadi Anda.

Peran terkait layanan ini dilampirkan ke kebijakan terkelola berikut ini: `KafkaServiceRolePolicy`. Untuk pembaruan kebijakan ini, lihat KafkaServiceRolePolicy.

AWSServiceRoleForKafka peran terkait layanan memercayakan layanan berikut untuk menjalankan peran tersebut:

- `kafka.amazonaws.com`

Kebijakan izin peran memungkinkan Amazon MSK menyelesaikan tindakan berikut pada sumber daya.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Action": [
    "ec2:CreateNetworkInterface",
    "ec2:DescribeNetworkInterfaces",
    "ec2:CreateNetworkInterfacePermission",
    "ec2:AttachNetworkInterface",
    "ec2:DeleteNetworkInterface",
    "ec2:DetachNetworkInterface",
    "ec2:DescribeVpcEndpoints",
    "acm-pca:GetCertificateAuthorityCertificate",
    "secretsmanager:ListSecrets"
   ],
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "ec2:ModifyVpcEndpoint"
   ],
   "Resource": "arn:*:ec2:*:*:subnet/*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "ec2:DeleteVpcEndpoints",
    "ec2:ModifyVpcEndpoint"
   ],
   "Resource": "arn:*:ec2:*:*:vpc-endpoint/*",
   "Condition": {
    "StringEquals": {
     "ec2:ResourceTag/AWSMSKManaged": "true"
    },
    "StringLike": {
     "ec2:ResourceTag/ClusterArn": "*"
    }
   }
  },
  {
   "Effect": "Allow",
```

```
    "Action": [
     "secretsmanager:GetResourcePolicy",
     "secretsmanager:PutResourcePolicy",
     "secretsmanager:DeleteResourcePolicy",
     "secretsmanager:DescribeSecret"
    ],
    "Resource": "*",
    "Condition": {
     "ArnLike": {
      "secretsmanager:SecretId": "arn:*:secretsmanager:*:*:secret:AmazonMSK_*"
     }
    }
   }
  ]
}
```

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, silakan lihat Izin Peran Tertaut Layanan di Panduan Pengguna IAM.

Membuat peran terkait layanan untuk Amazon MSK

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda membuat klaster MSK Amazon di AWS Management Console, API AWS CLI, atau AWS API, Amazon MSK membuat peran terkait layanan untuk Anda.

Jika Anda menghapus peran tertaut layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Saat Anda membuat kluster MSK Amazon, Amazon MSK membuat peran terkait layanan untuk Anda lagi.

Mengedit peran terkait layanan untuk Amazon MSK

Amazon MSK tidak mengizinkan Anda mengedit peran AWSServiceRoleForKafka terkait layanan. Setelah Anda membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat mengedit penjelasan peran menggunakan IAM. Untuk informasi selengkapnya, lihat Mengedit Peran Tertaut Layanan dalam Panduan Pengguna IAM.

Wilayah yang Didukung untuk peran terkait layanan MSK Amazon

Amazon MSK mendukung penggunaan peran terkait layanan di semua Wilayah tempat layanan tersedia. Untuk informasi selengkapnya, lihat AWS Wilayah dan Titik Akhir.

AWS kebijakan terkelola untuk Amazon MSK

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 AWS kebijakan yang dikelola dalam Panduan Pengguna IAM.

AWS kebijakan terkelola: AmazonMSK FullAccess

Kebijakan ini memberikan izin administratif yang memungkinkan akses penuh utama ke semua tindakan MSK Amazon. Izin dalam kebijakan ini dikelompokkan sebagai berikut:

- Izin MSK Amazon memungkinkan semua tindakan MSK Amazon.
- **`Amazon EC2`**izin — dalam kebijakan ini diperlukan untuk memvalidasi sumber daya yang diteruskan dalam permintaan API. Ini untuk memastikan Amazon MSK dapat berhasil menggunakan sumber daya dengan cluster. Izin Amazon EC2 lainnya dalam kebijakan ini memungkinkan Amazon MSK membuat AWS sumber daya yang diperlukan untuk memungkinkan Anda terhubung ke cluster Anda.
- **`AWS KMS`**izin - digunakan selama panggilan API untuk memvalidasi sumber daya yang diteruskan dalam permintaan. Mereka diperlukan untuk Amazon MSK untuk dapat menggunakan kunci yang diteruskan dengan cluster MSK Amazon.
- **`CloudWatch Logs, Amazon S3, and Amazon Data Firehose`**izin - diperlukan untuk Amazon MSK untuk dapat memastikan bahwa tujuan pengiriman log dapat dijangkau, dan valid untuk penggunaan log broker.
- **`IAM`**izin - diperlukan agar Amazon MSK dapat membuat peran terkait layanan di akun Anda dan memungkinkan Anda meneruskan peran eksekusi layanan ke Amazon MSK.

```
{
 "Version": "2012-10-17",
 "Statement": [{
   "Effect": "Allow",
   "Action": [
    "kafka:*",
    "ec2:DescribeSubnets",
    "ec2:DescribeVpcs",
    "ec2:DescribeSecurityGroups",
    "ec2:DescribeRouteTables",
    "ec2:DescribeVpcEndpoints",
    "ec2:DescribeVpcAttribute",
    "kms:DescribeKey",
    "kms:CreateGrant",
    "logs:CreateLogDelivery",
    "logs:GetLogDelivery",
    "logs:UpdateLogDelivery",
    "logs:DeleteLogDelivery",
    "logs:ListLogDeliveries",
    "logs:PutResourcePolicy",
    "logs:DescribeResourcePolicies",
    "logs:DescribeLogGroups",
    "S3:GetBucketPolicy",
    "firehose:TagDeliveryStream"
   ],
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "ec2:CreateVpcEndpoint"
   ],
   "Resource": [
    "arn:*:ec2:*:*:vpc/*",
    "arn:*:ec2:*:*:subnet/*",
    "arn:*:ec2:*:*:security-group/*"
   ]
  },
  {
   "Effect": "Allow",
   "Action": [
    "ec2:CreateVpcEndpoint"
   ],
```

```
        "Resource": [
         "arn:*:ec2:*:*:vpc-endpoint/*"
        ],
        "Condition": {
         "StringEquals": {
          "aws:RequestTag/AWSMSKManaged": "true"
         },
         "StringLike": {
          "aws:RequestTag/ClusterArn": "*"
         }
        }
       },
       {
        "Effect": "Allow",
        "Action": [
         "ec2:CreateTags"
        ],
        "Resource": "arn:*:ec2:*:*:vpc-endpoint/*",
        "Condition": {
         "StringEquals": {
          "ec2:CreateAction": "CreateVpcEndpoint"
         }
        }
       },
       {
        "Effect": "Allow",
        "Action": [
         "ec2:DeleteVpcEndpoints"
        ],
        "Resource": "arn:*:ec2:*:*:vpc-endpoint/*",
        "Condition": {
         "StringEquals": {
          "ec2:ResourceTag/AWSMSKManaged": "true"
         },
         "StringLike": {
          "ec2:ResourceTag/ClusterArn": "*"
         }
        }
       },
       {
        "Effect": "Allow",
        "Action": "iam:PassRole",
        "Resource": "*",
        "Condition": {
```

```
        "StringEquals": {
         "iam:PassedToService": "kafka.amazonaws.com"
        }
       }
      },
      {
       "Effect": "Allow",
       "Action": "iam:CreateServiceLinkedRole",
       "Resource": "arn:aws:iam::*:role/aws-service-role/kafka.amazonaws.com/
AWSServiceRoleForKafka*",
       "Condition": {
        "StringLike": {
         "iam:AWSServiceName": "kafka.amazonaws.com"
        }
       }
      },
      {
       "Effect": "Allow",
       "Action": [
        "iam:AttachRolePolicy",
        "iam:PutRolePolicy"
       ],
       "Resource": "arn:aws:iam::*:role/aws-service-role/kafka.amazonaws.com/
AWSServiceRoleForKafka*"
      },
      {
       "Effect": "Allow",
       "Action": "iam:CreateServiceLinkedRole",
       "Resource": "arn:aws:iam::*:role/aws-service-role/delivery.logs.amazonaws.com/
AWSServiceRoleForLogDelivery*",
       "Condition": {
        "StringLike": {
         "iam:AWSServiceName": "delivery.logs.amazonaws.com"
        }
       }
      }

     ]
    }
```

AWS kebijakan terkelola: Akses ReadOnly AmazonMSK

Kebijakan ini memberikan izin hanya-baca yang memungkinkan pengguna melihat informasi di Amazon MSK. Prinsipal dengan kebijakan ini terlampir tidak dapat membuat pembaruan atau menghapus sumber daya yang keluar, juga tidak dapat membuat sumber daya MSK Amazon baru. Misalnya, prinsipal dengan izin ini dapat melihat daftar cluster dan konfigurasi yang terkait dengan akun mereka, tetapi tidak dapat mengubah konfigurasi atau pengaturan cluster apa pun. Izin dalam kebijakan ini dikelompokkan sebagai berikut:

- **`Amazon MSK`**izin - memungkinkan Anda mencantumkan sumber daya MSK Amazon, menjelaskannya, dan mendapatkan informasi tentangnya.
- **`Amazon EC2`**izin - digunakan untuk menggambarkan VPC Amazon, subnet, grup keamanan, dan ENI yang terkait dengan kluster.
- **`AWS KMS`**izin — digunakan untuk menggambarkan kunci yang terkait dengan cluster.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "kafka:Describe*",
                "kafka:List*",
                "kafka:Get*",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "kms:DescribeKey"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

AWS kebijakan terkelola: KafkaServiceRolePolicy

Anda tidak dapat melampirkan KafkaServiceRolePolicy ke entitas IAM Anda. Kebijakan ini dilampirkan ke peran terkait layanan yang memungkinkan Amazon MSK melakukan tindakan

seperti mengelola titik akhir (konektor) VPC pada kluster MSK, mengelola antarmuka jaringan, dan mengelola kredensyal klaster. AWS Secrets Manager Untuk informasi selengkapnya, lihat the section called “Peran terkait layanan”.

AWS kebijakan terkelola: AWSMSKReplicatorExecutionRole

`AWSMSKReplicatorExecutionRole`Kebijakan ini memberikan izin kepada replikator MSK Amazon untuk mereplikasi data antar kluster MSK. Izin dalam kebijakan ini dikelompokkan sebagai berikut:

- **`cluster`**— Memberikan izin Amazon MSK Replicator untuk terhubung ke cluster menggunakan otentikasi IAM. Juga memberikan izin untuk mendeskripsikan dan mengubah cluster.
- **`topic`**— Memberikan izin Amazon MSK Replicator untuk mendeskripsikan, membuat, dan mengubah topik, dan untuk mengubah konfigurasi dinamis topik.
- **`consumer group`**— Memberikan izin Amazon MSK Replicator untuk mendeskripsikan dan mengubah grup konsumen, membaca dan menulis tanggal dari kluster MSK, dan untuk menghapus topik internal yang dibuat oleh replikator.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "ClusterPermissions",
   "Effect": "Allow",
   "Action": [
    "kafka-cluster:Connect",
    "kafka-cluster:DescribeCluster",
    "kafka-cluster:AlterCluster",
    "kafka-cluster:DescribeTopic",
    "kafka-cluster:CreateTopic",
    "kafka-cluster:AlterTopic",
    "kafka-cluster:WriteData",
    "kafka-cluster:ReadData",
    "kafka-cluster:AlterGroup",
    "kafka-cluster:DescribeGroup",
    "kafka-cluster:DescribeTopicDynamicConfiguration",
    "kafka-cluster:AlterTopicDynamicConfiguration",
    "kafka-cluster:WriteDataIdempotently"
   ],
   "Resource": [
    "arn:aws:kafka:*:*:cluster/*"
```

```
    ]
   },
   {
    "Sid": "TopicPermissions",
    "Effect": "Allow",
    "Action": [
     "kafka-cluster:DescribeTopic",
     "kafka-cluster:CreateTopic",
     "kafka-cluster:AlterTopic",
     "kafka-cluster:WriteData",
     "kafka-cluster:ReadData",
     "kafka-cluster:DescribeTopicDynamicConfiguration",
     "kafka-cluster:AlterTopicDynamicConfiguration",
     "kafka-cluster:AlterCluster"
    ],
    "Resource": [
     "arn:aws:kafka:*:*:topic/*/*"
    ]
   },
   {
    "Sid": "GroupPermissions",
    "Effect": "Allow",
    "Action": [
     "kafka-cluster:AlterGroup",
     "kafka-cluster:DescribeGroup"
    ],
    "Resource": [
     "arn:aws:kafka:*:*:group/*/*"
    ]
   }
  ]
}
```

Amazon MSK memperbarui kebijakan AWS terkelola

Lihat detail tentang pembaruan kebijakan AWS terkelola untuk Amazon MSK sejak layanan ini mulai melacak perubahan ini.

Perubahan	Deskripsi	Tanggal
WriteDataIdempotently izin ditambahkan ke AWSMSKRep	Amazon MSK menambahkan WriteDataIdempotently	Maret 12, 2024

Perubahan	Deskripsi	Tanggal
licatorExecutionRole - Perbarui ke kebijakan yang ada	izin ke AWSMSKReplicatorEx ecutionRole kebijakan untuk mendukung replikasi data antara kluster MSK.	
AWSMSKReplicatorEx ecutionRole – Kebijakan baru	Amazon MSK menambahk an AWSMSKReplicatorEx ecutionRole kebijakan untuk mendukung Amazon MSK Replicator.	Desember 4, 2023
AmazonMSK FullAccess - Update ke kebijakan yang ada	Amazon MSK menambahkan izin untuk mendukung Amazon MSK Replicator.	28 September 2023
KafkaServiceRolePolicy – Pembaruan ke kebijakan yang ada	Amazon MSK menambahk an izin untuk mendukung konektivitas pribadi multi-VPC.	8 Maret 2023
AmazonMSK FullAccess - Update ke kebijakan yang ada	Amazon MSK menambahkan izin Amazon EC2 baru untuk memungkinkan terhubung ke cluster.	30 November 2021
AmazonMSK FullAccess - Update ke kebijakan yang ada	Amazon MSK menambahkan izin baru untuk memungkin kannya menggambarkan tabel rute Amazon EC2.	November 19, 2021
Amazon MSK mulai melacak perubahan	Amazon MSK mulai melacak perubahan untuk kebijakan yang AWS dikelola.	November 19, 2021

Memecahkan masalah identitas dan akses MSK Amazon

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan Amazon MSK dan IAM.

Topik

- Saya tidak berwenang untuk melakukan tindakan di Amazon MSK

Saya tidak berwenang untuk melakukan tindakan di Amazon MSK

Jika AWS Management Console memberitahu Anda bahwa Anda tidak berwenang untuk melakukan tindakan, maka Anda harus menghubungi administrator Anda untuk bantuan. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Contoh kesalahan berikut terjadi ketika pengguna `mateojackson` IAM mencoba menggunakan konsol untuk menghapus cluster tetapi tidak memiliki `kafka:DeleteCluster` izin.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 kafka:DeleteCluster on resource: purchaseQueriesCluster
```

Dalam hal ini, Mateo meminta administratornya untuk memperbarui kebijakannya untuk mengizinkan dia mengakses sumber daya `purchaseQueriesCluster` menggunakan tindakan `kafka:DeleteCluster`.

Otentikasi dan otorisasi untuk Apache Kafka API

Anda dapat menggunakan IAM untuk mengautentikasi klien dan mengizinkan atau menolak tindakan Apache Kafka. Atau, Anda dapat menggunakan TLS atau SASL/SCRAM untuk mengautentikasi klien, dan Apache Kafka ACL untuk mengizinkan atau menolak tindakan.

Untuk informasi tentang cara mengontrol siapa yang dapat melakukan operasi MSK Amazon di klaster Anda, lihatthe section called “Otentikasi dan otorisasi untuk Amazon MSK API”.

Topik

- Kontrol akses IAM
- Otentikasi TLS timbal balik
- Otentikasi kredensi masuk dengan Secrets Manager AWS

- Apache Kafka ACL

Kontrol akses IAM

Kontrol akses IAM untuk Amazon MSK memungkinkan Anda menangani otentikasi dan otorisasi untuk klaster MSK Anda. Ini menghilangkan kebutuhan untuk menggunakan satu mekanisme untuk otentikasi dan satu lagi untuk otorisasi. Misalnya, ketika klien mencoba menulis ke klaster Anda, Amazon MSK menggunakan IAM untuk memeriksa apakah klien tersebut adalah identitas yang diautentikasi dan juga apakah itu berwenang untuk diproduksi ke cluster Anda. Kontrol akses IAM berfungsi untuk klien Java dan non-Java, termasuk klien Kafka yang ditulis dengan Python, Go, dan .NET. JavaScript

Amazon MSK mencatat peristiwa akses sehingga Anda dapat mengaudit mereka. Untuk informasi selengkapnya, lihat the section called “CloudTrail acara”.

Untuk memungkinkan kontrol akses IAM, Amazon MSK membuat modifikasi kecil pada kode sumber Apache Kafka. Modifikasi ini tidak akan menyebabkan perbedaan nyata dalam pengalaman Apache Kafka Anda.

Important

Kontrol akses IAM tidak berlaku untuk node Apache ZooKeeper . Untuk informasi tentang bagaimana Anda dapat mengontrol akses ke node tersebut, lihatthe section called “Mengontrol akses ke Apache ZooKeeper”.

Important

Pengaturan `allow.everyone.if.no.acl.found` Apache Kafka tidak berpengaruh jika cluster Anda menggunakan kontrol akses IAM.

Important

Anda dapat memanggil Apache Kafka ACL API untuk klaster MSK yang menggunakan kontrol akses IAM. Namun, Apache Kafka ACL tidak berpengaruh pada otorisasi untuk peran IAM. Anda harus menggunakan kebijakan IAM untuk mengontrol akses untuk peran IAM.

Cara kerja kontrol akses IAM untuk Amazon MSK

Untuk menggunakan kontrol akses IAM untuk Amazon MSK, lakukan langkah-langkah berikut, yang dijelaskan secara rinci di bagian ini.

- the section called “Buat cluster yang menggunakan kontrol akses IAM”
- the section called “Konfigurasikan klien untuk kontrol akses IAM”
- the section called “Buat kebijakan otorisasi”
- the section called “Dapatkan broker bootstrap untuk kontrol akses IAM”

Buat cluster yang menggunakan kontrol akses IAM

Bagian ini menjelaskan bagaimana Anda dapat menggunakan AWS Management Console, API, atau AWS CLI untuk membuat klaster yang menggunakan kontrol akses IAM. Untuk informasi tentang cara mengaktifkan kontrol akses IAM untuk klaster yang ada, lihatthe section called “Memperbarui keamanan”.

Gunakan AWS Management Console untuk membuat cluster yang menggunakan kontrol akses IAM

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Pilih Buat klaster.
3. Pilih Buat cluster dengan pengaturan khusus.
4. Di bagian Otentikasi, pilih kontrol akses IAM.
5. Selesaikan sisa alur kerja untuk membuat cluster.

Menggunakan API atau AWS CLI untuk membuat klaster yang menggunakan kontrol akses IAM

- Untuk membuat cluster dengan kontrol akses IAM diaktifkan, gunakan CreateClusterAPI atau perintah CLI create-cluster, dan teruskan JSON berikut untuk parameter:. `ClientAuthentication` `"ClientAuthentication": { "Sasl": { "Iam": { "Enabled": true } }`

Konfigurasikan klien untuk kontrol akses IAM

Untuk memungkinkan klien berkomunikasi dengan kluster MSK yang menggunakan kontrol akses IAM, Anda dapat menggunakan salah satu dari mekanisme ini:

- Konfigurasi klien non-Java menggunakan mekanisme SASL_OAUTHBEARER
- Konfigurasi klien Java menggunakan mekanisme SASL_OAUTHBEARER atau mekanisme AWS_MSK_IAM

Menggunakan mekanisme SASL_OAUTHBEARER untuk mengonfigurasi IAM

1. Edit file konfigurasi client.properties Anda menggunakan sintaks yang disorot dalam contoh klien Python Kafka di bawah ini sebagai panduan. Perubahan konfigurasi serupa dalam bahasa lain.

```
#!/usr/bin/python3from kafka import KafkaProducer
from kafka.errors import KafkaError
import socket
import time
from aws_msk_iam_sasl_signer import MSKAuthTokenProvider

class MSKTokenProvider():
    def token(self):
        token, _ = MSKAuthTokenProvider.generate_auth_token('<my aws region>')
        return token

tp = MSKTokenProvider()

producer = KafkaProducer(
    bootstrap_servers='<my bootstrap string>',
    security_protocol='SASL_SSL',
    sasl_mechanism='OAUTHBEARER',
    sasl_oauth_token_provider=tp,
    client_id=socket.gethostname(),
)

topic = "<my-topic>"
while True:
    try:
        inp=input(">")
        producer.send(topic, inp.encode())
        producer.flush()
        print("Produced!")
    except Exception:
        print("Failed to send message:", e)

producer.close()
```

2. Unduh pustaka pembantu untuk bahasa konfigurasi yang Anda pilih dan ikuti petunjuk di bagian Memulai di beranda perpustakaan bahasa tersebut.
   - JavaScript: https://github.com/aws/aws-msk-iam-sasl-signer-js#getting-started
   - Python: https://github.com/aws/aws-msk-iam-sasl-signer-python#get-started
   - Pergi: https://github.com/aws/aws-msk-iam-sasl-signer-go#getting-started
   - .NET: https://github.com/aws/aws-msk-iam-sasl-signer-net#getting-started
   - JAVA: Dukungan SASL_OAUTHBEARER untuk Java tersedia melalui file jar `aws-msk-iam-auth`

Menggunakan mekanisme MSK custom AWS_MSK_IAM untuk mengkonfigurasi IAM

1. Tambahkan yang berikut ini ke `client.properties` file. Ganti <PATH_TO_TRUST_STORE_FILE>dengan jalur yang sepenuhnya memenuhi syarat ke file trust store pada klien.

   > **Note**
   >
   > Jika Anda tidak ingin menggunakan sertifikat tertentu, Anda dapat menghapus `ssl.truststore.location=<PATH_TO_TRUST_STORE_FILE>` dari `client.properties` file Anda. Bila Anda tidak menentukan nilai untuk`ssl.truststore.location`, proses Java menggunakan sertifikat default.

   ```
   ssl.truststore.location=<PATH_TO_TRUST_STORE_FILE>
   security.protocol=SASL_SSL
   sasl.mechanism=AWS_MSK_IAM
   sasl.jaas.config=software.amazon.msk.auth.iam.IAMLoginModule required;
   sasl.client.callback.handler.class=software.amazon.msk.auth.iam.IAMClientCallbackHandler
   ```

   Untuk menggunakan profil bernama yang Anda buat untuk AWS kredensyal, sertakan `awsProfileName="your profile name";` dalam file konfigurasi klien Anda. Untuk informasi tentang profil bernama, lihat Profil bernama dalam AWS CLI dokumentasi.

2. Unduh file JAR aws-msk-iam-auth stabil terbaru, dan letakkan di jalur kelas. Jika Anda menggunakan Maven, tambahkan dependensi berikut, sesuaikan nomor versi sesuai kebutuhan:

   ```
   <dependency>
   ```

```
    <groupId>software.amazon.msk</groupId>
    <artifactId>aws-msk-iam-auth</artifactId>
    <version>1.0.0</version>
</dependency>
```

Plugin klien MSK Amazon bersumber terbuka di bawah lisensi Apache 2.0.

Buat kebijakan otorisasi

Lampirkan kebijakan otorisasi ke peran IAM yang sesuai dengan klien. Dalam kebijakan otorisasi, Anda menentukan tindakan mana yang akan diizinkan atau ditolak untuk peran tersebut. Jika klien Anda menggunakan instans Amazon EC2, kaitkan kebijakan otorisasi dengan peran IAM untuk instans Amazon EC2 tersebut. Atau, Anda dapat mengonfigurasi klien Anda untuk menggunakan profil bernama, dan kemudian Anda mengaitkan kebijakan otorisasi dengan peran untuk profil bernama tersebut. the section called “Konfigurasikan klien untuk kontrol akses IAM”menjelaskan cara mengkonfigurasi klien untuk menggunakan profil bernama.

Untuk informasi tentang cara membuat kebijakan IAM, lihat Membuat kebijakan IAM.

Berikut ini adalah contoh kebijakan otorisasi untuk cluster bernama MyTestCluster. Untuk memahami semantik `Action` dan `Resource` elemen, lihat. the section called “Semantik tindakan dan sumber daya”

> **Important**
>
> Perubahan yang Anda buat pada kebijakan IAM tercermin dalam API IAM dan segera. AWS CLI Namun, perlu waktu yang nyata agar perubahan kebijakan berlaku. Dalam kebanyakan kasus, perubahan kebijakan berlaku dalam waktu kurang dari satu menit. Kondisi jaringan terkadang dapat meningkatkan penundaan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:Connect",
                "kafka-cluster:AlterCluster",
```

```
                "kafka-cluster:DescribeCluster"
            ],
            "Resource": [
                "arn:aws:kafka:us-east-1:0123456789012:cluster/MyTestCluster/
abcd1234-0123-abcd-5678-1234abcd-1"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:*Topic*",
                "kafka-cluster:WriteData",
                "kafka-cluster:ReadData"
            ],
            "Resource": [
                "arn:aws:kafka:us-east-1:0123456789012:topic/MyTestCluster/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:AlterGroup",
                "kafka-cluster:DescribeGroup"
            ],
            "Resource": [
                "arn:aws:kafka:us-east-1:0123456789012:group/MyTestCluster/*"
            ]
        }
    ]
}
```

Untuk mempelajari cara membuat kebijakan dengan elemen tindakan yang sesuai dengan kasus penggunaan Apache Kafka yang umum, seperti memproduksi dan mengkonsumsi data, lihat. the section called “Kasus penggunaan umum”

Untuk Kafka versi 2.8.0 dan di atasnya, izin WriteDataIdempotently tidak digunakan lagi (KIP-679). Secara default, `enable.idempotence = true` diatur. Oleh karena itu, untuk Kafka versi 2.8.0 ke atas, IAM tidak menawarkan fungsionalitas yang sama dengan Kafka ACL. Tidak mungkin `WriteDataIdempotently` untuk topik dengan hanya menyediakan `WriteData` akses ke topik itu. Ini tidak mempengaruhi kasus ketika `WriteData` disediakan untuk SEMUA topik. Dalam hal ini, `WriteDataIdempotently` diperbolehkan. Hal ini disebabkan perbedaan dalam implementasi logika IAM versus bagaimana ACL Kafka diimplementasikan.

Untuk mengatasi hal ini, sebaiknya gunakan kebijakan yang mirip dengan contoh di bawah ini:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:Connect",
                "kafka-cluster:AlterCluster",
                "kafka-cluster:DescribeCluster",
                "kafka-cluster:WriteDataIdempotently"
            ],
            "Resource": [
                "arn:aws:kafka:us-east-1:0123456789012:cluster/MyTestCluster/
abcd1234-0123-abcd-5678-1234abcd-1"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:*Topic*",
                "kafka-cluster:WriteData",
                "kafka-cluster:ReadData"
            ],
            "Resource": [
                "arn:aws:kafka:us-east-1:0123456789012:topic/MyTestCluster/
abcd1234-0123-abcd-5678-1234abcd-1/TestTopic"
            ]
        }
    ]
}
```

Dalam hal ini, `WriteData` memungkinkan menulis ke`TestTopic`, sementara `WriteDataIdempotently` memungkinkan penulisan idempoten ke cluster. Penting untuk dicatat bahwa itu `WriteDataIdempotently` adalah izin tingkat cluster. Itu tidak dapat digunakan di tingkat topik. Jika `WriteDataIdempotently` dibatasi pada tingkat topik, kebijakan ini tidak akan berfungsi.

Dapatkan broker bootstrap untuk kontrol akses IAM

Lihat the section called “Mendapatkan broker bootstrap”.

Semantik tindakan dan sumber daya

Bagian ini menjelaskan semantik elemen tindakan dan sumber daya yang dapat Anda gunakan dalam kebijakan otorisasi IAM. Untuk contoh kebijakan, lihat the section called “Buat kebijakan otorisasi”.

Tindakan

Tabel berikut mencantumkan tindakan yang dapat Anda sertakan dalam kebijakan otorisasi saat Anda menggunakan kontrol akses IAM untuk Amazon MSK. Bila Anda menyertakan dalam kebijakan otorisasi tindakan dari kolom Tindakan tabel, Anda juga harus menyertakan tindakan terkait dari kolom Tindakan yang diperlukan.

Tindakan	Deskripsi	Tindakan yang diperlukan	Sumber daya yang dibutuhkan	Berlaku untuk cluster tanpa server
`kafka-cluster:Connect`	Memberikan izin untuk menghubungkan dan mengautentikasi ke cluster.	Tidak ada	cluster	Ya
`kafka-cluster:DescribeCluster`	Memberikan izin untuk mendeskripsikan berbagai aspek cluster, setara dengan DESCRIPTION CLUSTER ACL Apache Kafka.	`kafka-cluster:Connect`	cluster	Ya

`kafka-cluster:AlterCluster`	Memberikan izin untuk mengubah berbagai aspek cluster, setara dengan ALTER	`kafka-cluster:Connect`
		
		`kafka-cluster:Desc`
			cluster	Tidak

Tindakan	Deskripsi	Tindakan yang diperlukan	Sumber daya yang dibutuhkan	Berlaku untuk cluster tanpa server
	CLUSTER ACL Apache Kafka.	`ribeClust er`		
`kafka-clu ster:Desc ribeClust erDynamic Configura tion`	Memberika n izin untuk mendeskripsikan konfigurasi dinamis cluster, setara dengan APache Kafka DESCRIBE_ CONFIGS CLUSTER ACL.	`kafka-clu ster:Conn ect`	cluster	Tidak

`kafka-clu ster:Alte rClusterD ynamicCon figuration`	Memberikan izin untuk mengubah konfigurasi dinamis cluster, setara dengan ALTER_CON FIGS CLUSTER ACL Apache Kafka.	`kafka-clu ster:Conn ect`
		
		`kafka-clu ster:Desc ribeClust erDynamic Configura tion`
			cluster	Tidak

`kafka-clu ster:Writ eDataIdem potently`	Memberikan izin untuk menulis data idempoten tly pada cluster, setara dengan Apache Kafka IDEMPOTEN T_WRITE CLUSTER ACL.	`kafka-clu ster:Conn ect`
		
		`kafka-clu ster:Writ eData`
			cluster	Ya

Tindakan	Deskripsi	Tindakan yang diperlukan	Sumber daya yang dibutuhkan	Berlaku untuk cluster tanpa server
`kafka-cluster:CreateTopic`	Memberikan izin untuk membuat topik di cluster, setara dengan CREATE CLUSTER/ TOPIC ACL Apache Kafka.	`kafka-cluster:Connect`	topik	Ya
`kafka-cluster:DescribeTopic`	Memberikan izin untuk mendeskripsikan topik di cluster, setara dengan APache Kafka's DESCRIPTE TOPIC ACL.	`kafka-cluster:Connect`	topik	Ya

`kafka-cluster:AlterTopic`	Memberikan izin untuk mengubah topik di klaster, setara dengan ALTER TOPIC ACL Apache Kafka.	`kafka-cluster:Connect`
		
		`kafka-cluster:DescribeTopic`
			topik	Ya

`kafka-cluster:DeleteTopic`	Memberikan izin untuk menghapus topik di cluster, setara dengan DELETE TOPIC ACL Apache Kafka.	`kafka-cluster:Connect`
		
		`kafka-cluster:DescribeTopic`
			topik	Ya

Tindakan	Deskripsi	Tindakan yang diperlukan	Sumber daya yang dibutuhkan	Berlaku untuk cluster tanpa server
`kafka-cluster:DescribeTopicDynamicConfiguration`	Memberikan izin untuk mendeskripsikan konfigurasi dinamis topik pada klaster, setara dengan DESCRIBE_CONFIGS TOPIC ACL Apache Kafka.	`kafka-cluster:Connect`	topik	Ya
`kafka-cluster:AlterTopicDynamicConfiguration`	Memberikan izin untuk mengubah konfigurasi dinamis topik pada klaster, setara dengan ALTER_CONFIGS TOPIC ACL Apache Kafka.	`kafka-cluster:Connect` `kafka-cluster:DescribeTopicDynamicConfiguration`	topik	Ya

`kafka-cluster:ReadData`	Memberikan izin untuk membaca data dari topik di cluster, setara dengan READ TOPIC ACL Apache Kafka.	`kafka-cluster:Connect`
		`kafka-cluster:DescribeTopic`
		`kafka-cluster:AlterGroup`
			topik	Ya

Tindakan	Deskripsi	Tindakan yang diperlukan	Sumber daya yang dibutuhkan	Berlaku untuk cluster tanpa server

`kafka-cluster:WriteData`	Memberikan izin untuk menulis data ke topik di cluster, setara dengan WRITE TOPIC ACL Apache Kafka	`kafka-cluster:Connect`
		
		`kafka-cluster:DescribeTopic`
			topik	Ya

`kafka-cluster:DescribeGroup`	Memberikan izin untuk mendeskripsikan grup pada sebuah cluster, setara dengan Apache Kafka's DESCRIPTE GROUP ACL.	`kafka-cluster:Connect`	grup	Ya

`kafka-cluster:AlterGroup`	Memberikan izin untuk bergabung dengan grup di cluster, setara dengan READ GROUP ACL Apache Kafka.	`kafka-cluster:Connect`
		
		`kafka-cluster:DescribeGroup`
			grup	Ya

`kafka-cluster:DeleteGroup`	Memberikan izin untuk menghapus grup di cluster, setara dengan DELETE GROUP ACL Apache Kafka.	`kafka-cluster:Connect`
		
		`kafka-cluster:DescribeGroup`
			grup	Ya

Tindakan	Deskripsi	Tindakan yang diperlukan	Sumber daya yang dibutuhkan	Berlaku untuk cluster tanpa server
`kafka-cluster:DescribeTransactionalId`	Memberikan izin untuk mendeskripsikan ID transaksional pada klaster, setara dengan DESCRIPT TRANSACTIONAL_ID ACL Apache Kafka.	`kafka-cluster:Connect`	transaksional-id	Ya

`kafka-cluster:AlterTransactionalId`	Memberikan izin untuk mengubah ID transaksional pada klaster, setara dengan WRITE TRANSACTIONAL_ID ACL Apache Kafka.	`kafka-cluster:Connect`
		
		`kafka-cluster:DescribeTransactionalId`
		
		`kafka-cluster:WriteData`
			transaksional-id	Ya

Anda dapat menggunakan wildcard asterisk (*) beberapa kali dalam tindakan setelah titik dua. Berikut ini adalah beberapa contohnya.

- `kafka-cluster:*Topic`singkatan dari`kafka-cluster:CreateTopic`,`kafka-cluster:DescribeTopic`,`kafka-cluster:AlterTopic`, dan`kafka-cluster:DeleteTopic`. Itu tidak termasuk `kafka-cluster:DescribeTopicDynamicConfiguration` atau`kafka-cluster:AlterTopicDynamicConfiguration`.
- `kafka-cluster:*`singkatan dari semua izin.

Sumber daya

Tabel berikut menunjukkan empat jenis sumber daya yang dapat Anda gunakan dalam kebijakan otorisasi saat Anda menggunakan kontrol akses IAM untuk Amazon MSK. Anda bisa mendapatkan klaster Amazon Resource Name (ARN) dari AWS Management Console atau dengan menggunakan DescribeClusterAPI atau perintah AWS CLI describe-cluster. Anda kemudian dapat menggunakan ARN cluster untuk membangun ARN topik, grup, dan ID transaksional. Untuk menentukan sumber daya dalam kebijakan otorisasi, gunakan ARN sumber daya tersebut.

Sumber Daya	Format ARN
Klaster	*arn:aws:kafka: wilayah: account-id:cluster/cluster-name/ cluster-uuid*
Topik	*arn:aws:kafka: wilayah: account-id:topik/cluster-name / cluster-uid/topic-name*
Grup	*arn:aws:kafka: wilayah: account-id:group/cluster-name / cluster-uuid/group-name*
ID Transaksional	*arn:aws:kafka: wilayah: account-id:transactional-id/ cluster-name /cluster-uuid/transactional-id*

Anda dapat menggunakan wildcard asterisk (*) beberapa kali di mana saja di bagian ARN yang muncul setelah`:cluster/`,,, `:topic/` dan. `:group/` `:transactional-id/` Berikut ini adalah beberapa contoh bagaimana Anda dapat menggunakan wildcard asterisk (*) untuk merujuk ke beberapa sumber daya:

- `arn:aws:kafka:us-east-1:0123456789012:topic/MyTestCluster/*`: semua topik di cluster mana pun bernama MyTestCluster, terlepas dari UUID cluster.
- `arn:aws:kafka:us-east-1:0123456789012:topic/MyTestCluster/abcd1234-0123-abcd-5678-1234abcd-1/*_test`: semua topik yang namanya diakhiri dengan “_test” di cluster yang namanya MyTestCluster dan UUIDnya abcd1234-0123-abcd-5678-1234abcd-1.
- `arn:aws:kafka:us-east-1:0123456789012:transactional-id/MyTestCluster/*/5555abcd-1111-abcd-1234-abcd1234-1`: semua transaksi yang ID transaksionalnya adalah 5555abcd-1111-abcd-1234-abcd1234-1, di semua inkarnasi klaster yang disebutkan di

akun Anda. MyTestCluster Ini berarti bahwa jika Anda membuat klaster bernama MyTestCluster, lalu menghapusnya, dan kemudian membuat cluster lain dengan nama yang sama, Anda dapat menggunakan ARN sumber daya ini untuk mewakili ID transaksi yang sama pada kedua cluster. Namun, cluster yang dihapus tidak dapat diakses.

Kasus penggunaan umum

Kolom pertama dalam tabel berikut menunjukkan beberapa kasus penggunaan umum. Untuk mengotorisasi klien untuk melaksanakan kasus penggunaan tertentu, sertakan tindakan yang diperlukan untuk kasus penggunaan tersebut dalam kebijakan otorisasi klien, dan atur `Effect` ke. `Allow`

Untuk informasi tentang semua tindakan yang merupakan bagian dari kontrol akses IAM untuk Amazon MSK, lihat. the section called “Semantik tindakan dan sumber daya”

> **Note**
>
> Tindakan ditolak secara default. Anda harus secara eksplisit mengizinkan setiap tindakan yang ingin Anda berikan otorisasi kepada klien untuk dilakukan.

Kasus penggunaan	Tindakan yang diperlukan
Admin	`kafka-cluster:*`
Buat topik	`kafka-cluster:Connect` `kafka-cluster:CreateTopic`

Menghasilkan data	`kafka-cluster:Connect`
	`kafka-cluster:DescribeTopic`
	`kafka-cluster:WriteData`

Konsumsi data	`kafka-cluster:Connect`
	`kafka-cluster:DescribeTopic`
	`kafka-cluster:DescribeGroup`

Kasus penggunaan	Tindakan yang diperlukan

	`kafka-cluster:AlterGroup`
	
	`kafka-cluster:ReadData`

Menghasilkan data secara idempotently	`kafka-cluster:Connect`
	
	`kafka-cluster:DescribeTopic`
	
	`kafka-cluster:WriteData`
	
	`kafka-cluster:WriteDataIdempotently`

Menghasilkan data secara transaksional	`kafka-cluster:Connect`
	
	`kafka-cluster:DescribeTopic`
	
	`kafka-cluster:WriteData`
	
	`kafka-cluster:DescribeTransactionalId`
	
	`kafka-cluster:AlterTransactionalId`

Jelaskan konfigurasi cluster	`kafka-cluster:Connect`
	
	`kafka-cluster:DescribeClusterDynamicConfiguration`

Perbarui konfigurasi cluster	`kafka-cluster:Connect`
	
	`kafka-cluster:DescribeClusterDynamicConfiguration`
	
	`kafka-cluster:AlterClusterDynamicConfiguration`

Kasus penggunaan	Tindakan yang diperlukan

Jelaskan konfigurasi suatu topik	`kafka-cluster:Connect`
	
	`kafka-cluster:DescribeTopic DynamicConfiguration`

Perbarui konfigurasi topik	`kafka-cluster:Connect`
	
	`kafka-cluster:DescribeTopic DynamicConfiguration`
	
	`kafka-cluster:AlterTopicDyn amicConfiguration`

Mengubah topik	`kafka-cluster:Connect`
	
	`kafka-cluster:DescribeTopic`
	
	`kafka-cluster:AlterTopic`

Otentikasi TLS timbal balik

Anda dapat mengaktifkan otentikasi klien dengan TLS untuk koneksi dari aplikasi Anda ke broker MSK Amazon Anda. Untuk menggunakan otentikasi klien, Anda memerlukan file AWS Private CA. AWS Private CA Bisa sama dengan cluster Anda Akun AWS , atau di akun yang berbeda. Untuk informasi tentang AWS Private CA s, lihat Membuat dan Mengelola a AWS Private CA.

> **Note**
>
> Otentikasi TLS saat ini tidak tersedia di Wilayah Beijing dan Ningxia.

Amazon MSK tidak mendukung daftar pencabutan sertifikat (CRL). Untuk mengontrol akses ke topik klaster Anda atau memblokir sertifikat yang disusupi, gunakan Apache Kafka ACL dan grup keamanan. AWS Untuk informasi tentang menggunakan Apache Kafka ACL, lihat. the section called “Apache Kafka ACL”

Topik ini berisi bagian-bagian berikut:

- Untuk membuat cluster yang mendukung otentikasi klien
- Untuk mengatur klien untuk menggunakan otentikasi
- Untuk menghasilkan dan menggunakan pesan menggunakan otentikasi

Untuk membuat cluster yang mendukung otentikasi klien

Prosedur ini menunjukkan kepada Anda cara mengaktifkan otentikasi klien menggunakan file. AWS Private CA

> **Note**
>
> Kami sangat merekomendasikan penggunaan independen AWS Private CA untuk setiap cluster MSK ketika Anda menggunakan TLS timbal balik untuk mengontrol akses. Melakukannya akan memastikan bahwa sertifikat TLS yang ditandatangani oleh PCA hanya mengautentikasi dengan satu cluster MSK.

1. Buat file bernama `clientauthinfo.json` dengan isi berikut ini. Ganti *`Private-CA-ARN dengan ARN PCA`* Anda.

```
{
   "Tls": {
       "CertificateAuthorityArnList": ["Private-CA-ARN"]
    }
}
```

2. Buat file bernama `brokernodegroupinfo.json` seperti yang dijelaskan dalamthe section called “Membuat cluster menggunakan AWS CLI”.
3. Otentikasi klien mengharuskan Anda juga mengaktifkan enkripsi dalam perjalanan antara klien dan broker. Buat file bernama `encryptioninfo.json` dengan isi berikut ini. Ganti *`KMS-Key-ARN dengan ARN`* kunci KMS Anda. Anda dapat mengatur `ClientBroker` ke `TLS` atau`TLS_PLAINTEXT`.

```
{
   "EncryptionAtRest": {
       "DataVolumeKMSKeyId": "KMS-Key-ARN"
    },
   "EncryptionInTransit": {
```

```
        "InCluster": true,
        "ClientBroker": "TLS"
    }
}
```

Untuk informasi selengkapnya tentang enkripsi, lihatthe section called “Enkripsi”.

4. Pada mesin tempat Anda AWS CLI menginstal, jalankan perintah berikut untuk membuat cluster dengan otentikasi dan enkripsi dalam transit diaktifkan. Simpan ARN cluster yang disediakan dalam tanggapan.

```
aws kafka create-cluster --cluster-name "AuthenticationTest" --broker-node-group-
info file://brokernodegroupinfo.json --encryption-info file://encryptioninfo.json
 --client-authentication file://clientauthinfo.json --kafka-version "{YOUR KAFKA
 VERSION}" --number-of-broker-nodes 3
```

Untuk mengatur klien untuk menggunakan otentikasi

1. Buat instans Amazon EC2 untuk digunakan sebagai mesin klien. Untuk mempermudah, buat instance ini di VPC yang sama yang Anda gunakan untuk cluster. Lihat the section called “Langkah 3: Buat mesin klien” contoh cara membuat mesin klien seperti itu.
2. Buat topik. Sebagai contoh, lihat instruksi di bawahthe section called “Langkah 4: Buat topik”.
3. Pada mesin tempat Anda AWS CLI menginstal, jalankan perintah berikut untuk mendapatkan broker bootstrap dari cluster. Ganti *Cluster-ARN dengan ARN* cluster Anda.

```
aws kafka get-bootstrap-brokers --cluster-arn Cluster-ARN
```

Simpan string yang terkait `BootstrapBrokerStringTls` dengan respons.

4. Pada mesin klien Anda, jalankan perintah berikut untuk menggunakan toko kepercayaan JVM untuk membuat toko kepercayaan klien Anda. Jika jalur JVM Anda berbeda, sesuaikan perintahnya.

```
cp /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.201.b09-0.amzn2.x86_64/jre/lib/security/
cacerts kafka.client.truststore.jks
```

5. Pada mesin klien Anda, jalankan perintah berikut untuk membuat kunci pribadi untuk klien Anda. *Ganti Dibedakan Nama, Contoh-Alias, Your-Store-Pass, dan Your-Key-Pass dengan string pilihan Anda.*

```
keytool -genkey -keystore kafka.client.keystore.jks -validity 300 -storepass Your-Store-Pass -keypass Your-Key-Pass -dname "CN=Distinguished-Name" -alias Example-Alias -storetype pkcs12
```

6. Pada mesin klien Anda, jalankan perintah berikut untuk membuat permintaan sertifikat dengan kunci pribadi yang Anda buat pada langkah sebelumnya.

```
keytool -keystore kafka.client.keystore.jks -certreq -file client-cert-sign-request
 -alias Example-Alias -storepass Your-Store-Pass -keypass Your-Key-Pass
```

7. Buka `client-cert-sign-request` file dan pastikan bahwa itu dimulai dengan `-----BEGIN CERTIFICATE REQUEST-----` dan diakhiri dengan`-----END CERTIFICATE REQUEST-----`. Jika dimulai dengan`-----BEGIN NEW CERTIFICATE REQUEST-----`, hapus kata `NEW` (dan spasi tunggal yang mengikutinya) dari awal dan akhir file.

8. Pada mesin tempat Anda AWS CLI menginstal, jalankan perintah berikut untuk menandatangani permintaan sertifikat Anda. Ganti *`Private-CA-ARN dengan ARN PCA`* Anda. Anda dapat mengubah nilai validitas jika Anda mau. Di sini kita menggunakan 300 sebagai contoh.

```
aws acm-pca issue-certificate --certificate-authority-arn Private-CA-ARN --csr
 fileb://client-cert-sign-request --signing-algorithm "SHA256WITHRSA" --validity
 Value=300,Type="DAYS"
```

   Simpan sertifikat ARN yang disediakan dalam tanggapan.

> **Note**
>
> Untuk mengambil sertifikat klien Anda, gunakan `acm-pca get-certificate` perintah dan tentukan ARN sertifikat Anda. Untuk informasi selengkapnya, lihat mendapatkan sertifikat di Referensi AWS CLI Perintah.

9. Jalankan perintah berikut untuk mendapatkan sertifikat yang AWS Private CA ditandatangani untuk Anda. Ganti *`Certificate-ARN dengan`* ARN yang Anda peroleh dari respons ke perintah sebelumnya.

```
aws acm-pca get-certificate --certificate-authority-arn Private-CA-ARN --
certificate-arn Certificate-ARN
```

10. Dari hasil JSON menjalankan perintah sebelumnya, salin string yang terkait dengan `Certificate` dan. `CertificateChain` Tempel kedua string ini dalam file baru bernama signed-certificate-from-acm. Tempel string yang terkait dengan `Certificate` pertama, diikuti oleh string yang terkait dengan`CertificateChain`. Ganti \n karakter dengan baris baru. Berikut ini adalah struktur file setelah Anda menempelkan sertifikat dan rantai sertifikat di dalamnya.

```
-----BEGIN CERTIFICATE-----
...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
...
-----END CERTIFICATE-----
```

11. Jalankan perintah berikut pada mesin klien untuk menambahkan sertifikat ini ke keystore Anda sehingga Anda dapat mempresentasikannya ketika Anda berbicara dengan broker MSK.

```
keytool -keystore kafka.client.keystore.jks -import -file signed-certificate-from-
acm -alias Example-Alias -storepass Your-Store-Pass -keypass Your-Key-Pass
```

12. Buat file bernama `client.properties` dengan isi berikut ini. Sesuaikan lokasi truststore dan keystore ke jalur tempat Anda menyimpan. `kafka.client.truststore.jks` Ganti versi klien Kafka Anda dengan *`placeholder {YOUR KAFKA VERSION}`*.

```
security.protocol=SSL
ssl.truststore.location=/tmp/kafka_2.12-{YOUR KAFKA VERSION}/
kafka.client.truststore.jks
ssl.keystore.location=/tmp/kafka_2.12-{YOUR KAFKA VERSION}/
kafka.client.keystore.jks
ssl.keystore.password=Your-Store-Pass
ssl.key.password=Your-Key-Pass
```

Untuk menghasilkan dan menggunakan pesan menggunakan otentikasi

1. Jalankan perintah berikut untuk membuat topik. File bernama `client.properties` adalah yang Anda buat di prosedur sebelumnya.

```
<path-to-your-kafka-installation>/bin/kafka-topics.sh --create --bootstrap-
server BootstrapBroker-String --replication-factor 3 --partitions 1 --topic
 ExampleTopic --command-config client.properties
```

2. Jalankan perintah berikut untuk memulai produsen konsol. File bernama `client.properties` adalah yang Anda buat di prosedur sebelumnya.

```
<path-to-your-kafka-installation>/bin/kafka-console-producer.sh --bootstrap-
server BootstrapBroker-String --topic ExampleTopic --producer.config
 client.properties
```

3. Di jendela perintah baru di mesin klien Anda, jalankan perintah berikut untuk memulai konsumen konsol.

```
<path-to-your-kafka-installation>/bin/kafka-console-consumer.sh --bootstrap-
server BootstrapBroker-String --topic ExampleTopic --consumer.config
 client.properties
```

4. Ketik pesan di jendela produser dan saksikan mereka muncul di jendela konsumen.

Otentikasi kredensi masuk dengan Secrets Manager AWS

Anda dapat mengontrol akses ke kluster MSK Amazon menggunakan kredensyal masuk yang disimpan dan diamankan menggunakan Secrets Manager. AWS Menyimpan kredensyal pengguna di Secrets Manager mengurangi overhead otentikasi klaster seperti mengaudit, memperbarui, dan memutar kredensyal. Secrets Manager juga memungkinkan Anda berbagi kredensyal pengguna di seluruh cluster.

Topik ini berisi bagian-bagian berikut:

- Cara kerjanya
- Menyiapkan otentikasi SASL/SCRAM untuk klaster MSK Amazon
- Bekerja dengan pengguna
- Batasan

Cara kerjanya

Otentikasi kredensyal masuk untuk Amazon MSK menggunakan otentikasi SASL/SCRAM (Otentikasi Sederhana dan Lapisan Keamanan/Mekanisme Respons Tantangan Asin). Untuk menyiapkan autentikasi kredensyal masuk untuk klaster, Anda membuat sumber daya Rahasia di Secrets Manager, dan mengaitkan kredensyal masuk dengan AWS rahasia tersebut.

SASL/SCRAM didefinisikan dalam RFC 5802. SCRAM menggunakan algoritma hashing aman, dan tidak mengirimkan kredensyal masuk plaintext antara klien dan server.

> **Note**
>
> Saat Anda mengatur otentikasi SASL/SCRAM untuk klaster Anda, Amazon MSK mengaktifkan enkripsi TLS untuk semua lalu lintas antara klien dan broker.

Menyiapkan otentikasi SASL/SCRAM untuk klaster MSK Amazon

Untuk mengatur AWS rahasia di Secrets Manager, ikuti tutorial Creating and Retrieving a Secret di Panduan Pengguna AWS Secrets Manager.

Perhatikan persyaratan berikut saat membuat rahasia untuk cluster MSK Amazon:

- Pilih jenis rahasia lainnya (misalnya kunci API) untuk tipe rahasia.
- Nama rahasia Anda harus dimulai dengan awalan AmazonMSK_.
- Anda harus menggunakan AWS KMS kunci kustom yang ada atau membuat AWS KMS kunci khusus baru untuk rahasia Anda. Secrets Manager menggunakan AWS KMS kunci default untuk rahasia secara default.

  > **Important**
  >
  > Rahasia yang dibuat dengan AWS KMS kunci default tidak dapat digunakan dengan kluster MSK Amazon.

- Data kredensi login Anda harus dalam format berikut untuk memasukkan pasangan nilai kunci menggunakan opsi Plaintext.

  ```
  {
    "username": "alice",
  ```

```
  "password": "alice-secret"
}
```

- Catat nilai ARN (Amazon Resource Name) untuk rahasia Anda.
- **Important**

  Anda tidak dapat mengaitkan rahasia Secrets Manager dengan klaster yang melebihi batas yang dijelaskan dalamthe section called “ Ukuran kluster Anda dengan benar: Jumlah partisi per broker”.
- Jika Anda menggunakan AWS CLI untuk membuat rahasia, tentukan ID kunci atau ARN untuk parameter. `kms-key-id` Jangan tentukan alias.
- Untuk mengaitkan rahasia dengan cluster Anda, gunakan konsol MSK Amazon, atau BatchAssociateScramSecretoperasinya.

  Important

  Saat Anda mengaitkan rahasia dengan klaster, Amazon MSK melampirkan kebijakan sumber daya ke rahasia yang memungkinkan klaster Anda mengakses dan membaca nilai rahasia yang Anda tetapkan. Anda tidak boleh mengubah kebijakan sumber daya ini. Melakukannya dapat mencegah cluster Anda mengakses rahasia Anda.

  Contoh input JSON berikut untuk `BatchAssociateScramSecret` operasi mengaitkan rahasia dengan cluster:

```
{
  "clusterArn" : "arn:aws:kafka:us-west-2:0123456789019:cluster/SalesCluster/
abcd1234-abcd-cafe-abab-9876543210ab-4",
  "secretArnList": [
    "arn:aws:secretsmanager:us-west-2:0123456789019:secret:AmazonMSK_MyClusterSecret"
  ]
}
```

Menghubungkan ke klaster Anda dengan kredensyal masuk

Setelah Anda membuat rahasia dan mengaitkannya dengan cluster Anda, Anda dapat menghubungkan klien Anda ke cluster. Contoh langkah berikut menunjukkan cara menghubungkan

klien ke cluster yang menggunakan otentikasi SASL/SCRAM, dan cara menghasilkan dan mengkonsumsi dari topik contoh.

1. *Jalankan perintah berikut pada mesin yang memiliki AWS CLI diinstal, menggantikan ClusterARN dengan ARN cluster Anda.*

   ```
   aws kafka get-bootstrap-brokers --cluster-arn clusterARN
   ```

2. Untuk membuat contoh topik, jalankan perintah berikut, ganti *BootstrapServerString* dengan salah satu titik akhir broker yang Anda peroleh pada langkah sebelumnya.

   ```
   <path-to-your-kafka-installation>/bin/kafka-topics.sh --create --bootstrap-
   server BootstrapServerString --replication-factor 3 --partitions 1 --topic
    ExampleTopicName
   ```

3. Di mesin klien Anda, buat file konfigurasi JAAS yang berisi kredensyal pengguna yang disimpan dalam rahasia Anda. Misalnya, untuk pengguna alice, buat file yang dipanggil `users_jaas.conf` dengan konten berikut.

   ```
   KafkaClient {
      org.apache.kafka.common.security.scram.ScramLoginModule required
      username="alice"
      password="alice-secret";
   };
   ```

4. Gunakan perintah berikut untuk mengekspor file konfigurasi JAAS Anda sebagai parameter `KAFKA_OPTS` lingkungan.

   ```
   export KAFKA_OPTS=-Djava.security.auth.login.config=<path-to-jaas-file>/
   users_jaas.conf
   ```

5. Buat file bernama `kafka.client.truststore.jks` dalam `./tmp` direktori.

6. Gunakan perintah berikut untuk menyalin file penyimpanan kunci JDK dari `cacerts` folder JVM Anda ke `kafka.client.truststore.jks` file yang Anda buat pada langkah sebelumnya. Ganti *JDKFolder* dengan nama folder JDK pada instance Anda. Misalnya, folder JDK Anda mungkin diberi nama. `java-1.8.0-openjdk-1.8.0.201.b09-0.amzn2.x86_64`

   ```
   cp /usr/lib/jvm/JDKFolder/jre/lib/security/cacerts /tmp/kafka.client.truststore.jks
   ```

7. Di `bin` direktori instalasi Apache Kafka Anda, buat file properti klien yang disebut `client_sasl.properties` dengan konten berikut. File ini mendefinisikan mekanisme dan protokol SASL.

```
security.protocol=SASL_SSL
sasl.mechanism=SCRAM-SHA-512
ssl.truststore.location=<path-to-keystore-file>/kafka.client.truststore.jks
```

8. Ambil string broker bootstrap Anda dengan perintah berikut. Ganti *`ClusterArn`*dengan Nama Sumber Daya Amazon (ARN) klaster Anda:

```
aws kafka get-bootstrap-brokers --cluster-arn ClusterArn
```

   Dari hasil JSON perintah, simpan nilai yang terkait dengan string bernama`BootstrapBrokerStringSaslScram`.

9. Untuk menghasilkan contoh topik yang Anda buat, jalankan perintah berikut di mesin klien Anda. Ganti *`BootstrapBrokerStringSaslScram`* dengan nilai yang Anda ambil pada langkah sebelumnya.

```
<path-to-your-kafka-installation>/bin/kafka-console-producer.sh --broker-
list BootstrapBrokerStringSaslScram --topic ExampleTopicName --producer.config
 client_sasl.properties
```

10. Untuk mengkonsumsi dari topik yang Anda buat, jalankan perintah berikut di mesin klien Anda. Ganti *`BootstrapBrokerStringSaslScram`* dengan nilai yang Anda peroleh sebelumnya.

```
<path-to-your-kafka-installation>/bin/kafka-console-consumer.sh --bootstrap-
server BootstrapBrokerStringSaslScram --topic ExampleTopicName --from-beginning --
consumer.config client_sasl.properties
```

Bekerja dengan pengguna

Membuat pengguna: Anda membuat pengguna dalam rahasia Anda sebagai pasangan nilai kunci. Saat Anda menggunakan opsi Plaintext di konsol Secrets Manager, Anda harus menentukan data kredensi masuk dalam format berikut.

```
{
  "username": "alice",
```

```
  "password": "alice-secret"
}
```

Mencabut akses pengguna: Untuk mencabut kredensyal pengguna untuk mengakses kluster, sebaiknya Anda menghapus atau menerapkan ACL di klaster terlebih dahulu, lalu memisahkan rahasianya. Ini karena hal-hal berikut:

- Menghapus pengguna tidak menutup koneksi yang ada.
- Perubahan pada rahasia Anda membutuhkan waktu hingga 10 menit untuk disebarkan.

Untuk informasi tentang menggunakan ACL dengan Amazon MSK, lihat. Apache Kafka ACL

Untuk cluster yang menggunakan ZooKeeper mode, kami menyarankan Anda membatasi akses ke ZooKeeper node Anda untuk mencegah pengguna memodifikasi ACL. Untuk informasi selengkapnya, lihat Mengontrol akses ke Apache ZooKeeper.

Batasan

Perhatikan batasan berikut saat menggunakan rahasia SCRAM:

- Amazon MSK hanya mendukung otentikasi SCRAM-SHA-512.
- Cluster MSK Amazon dapat memiliki hingga 1000 pengguna.
- Anda harus menggunakan sebuah AWS KMS key dengan Rahasia Anda. Anda tidak dapat menggunakan Rahasia yang menggunakan kunci enkripsi Secrets Manager default dengan Amazon MSK. Untuk informasi tentang membuat kunci KMS, lihat Membuat kunci KMS enkripsi simetris.
- Anda tidak dapat menggunakan kunci KMS asimetris dengan Secrets Manager.
- Anda dapat mengaitkan hingga 10 rahasia dengan cluster sekaligus menggunakan BatchAssociateScramSecretoperasi.
- Nama rahasia yang terkait dengan cluster MSK Amazon harus memiliki awalan AmazonMSK_.
- Rahasia yang terkait dengan kluster MSK Amazon harus berada di akun dan AWS wilayah Amazon Web Services yang sama dengan cluster.

Apache Kafka ACL

Apache Kafka memiliki otorisasi yang dapat dicolokkan dan dikirimkan dengan implementasi otorisasi. out-of-box Amazon MSK memungkinkan otorisasi ini dalam `server.properties` file di broker.

Apache Kafka ACL memiliki format “Principal P adalah [Diizinkan/Ditolak] Operasi O Dari Host H pada Resource R apa pun yang cocok dengan RP”. ResourcePattern Jika RP tidak cocok dengan R sumber daya tertentu, maka R tidak memiliki ACL terkait, dan oleh karena itu tidak ada orang lain selain pengguna super yang diizinkan mengakses R. Untuk mengubah perilaku Apache Kafka ini, Anda menetapkan properti `allow.everyone.if.no.acl.found` ke true. Amazon MSK menyetelnya ke true secara default. Ini berarti bahwa dengan kluster MSK Amazon, jika Anda tidak secara eksplisit menyetel ACL pada sumber daya, semua prinsipal dapat mengakses sumber daya ini. Jika Anda mengaktifkan ACL pada sumber daya, hanya kepala sekolah yang berwenang yang dapat mengaksesnya. Jika Anda ingin membatasi akses ke topik dan mengotorisasi klien menggunakan otentikasi timbal balik TLS, tambahkan ACL menggunakan Apache Kafka Authorizer CLI. Untuk informasi selengkapnya tentang menambahkan, menghapus, dan mencantumkan ACL, lihat Antarmuka Baris Perintah Otorisasi Kafka.

Selain klien, Anda juga perlu memberikan semua broker Anda akses ke topik Anda sehingga broker dapat mereplikasi pesan dari partisi utama. Jika broker tidak memiliki akses ke topik, replikasi untuk topik gagal.

Untuk menambah atau menghapus akses baca dan tulis ke topik

1. Tambahkan broker Anda ke tabel ACL untuk memungkinkan mereka membaca dari semua topik yang memiliki ACL di tempat. Untuk memberi broker Anda membaca akses ke topik, jalankan perintah berikut pada mesin klien yang dapat berkomunikasi dengan cluster MSK.

   Ganti *`Dibedakan Nama`* dengan DNS dari salah satu broker bootstrap cluster Anda, lalu ganti string sebelum periode pertama dalam nama yang dibedakan ini dengan tanda bintang (). * Misalnya, jika salah satu broker bootstrap cluster Anda memiliki DNS`b-6.mytestcluster.67281x.c4.kafka.us-east-1.amazonaws.com`, ganti *`Differentiished-Name`* dalam perintah berikut dengan. `*.mytestcluster.67281x.c4.kafka.us-east-1.amazonaws.com` Untuk informasi tentang cara mendapatkan broker bootstrap, lihatthe section called “Mendapatkan broker bootstrap”.

```
<path-to-your-kafka-installation>/bin/kafka-acls.sh --authorizer-properties
 --bootstrap-server BootstrapServerString --add --allow-principal
 "User:CN=Distinguished-Name" --operation Read --group=* --topic Topic-Name
```

2. Untuk memberikan akses baca ke topik, jalankan perintah berikut di mesin klien Anda. Jika Anda menggunakan otentikasi TLS timbal balik, gunakan *Nama Dibedakan* yang sama yang Anda gunakan saat Anda membuat kunci pribadi.

   ```
   <path-to-your-kafka-installation>/bin/kafka-acls.sh --authorizer-properties
    --bootstrap-server BootstrapServerString --add --allow-principal
    "User:CN=Distinguished-Name" --operation Read --group=* --topic Topic-Name
   ```

   Untuk menghapus akses baca, Anda dapat menjalankan perintah yang sama, menggantinya `--add` dengan`--remove`.

3. Untuk memberikan akses tulis ke topik, jalankan perintah berikut di mesin klien Anda. Jika Anda menggunakan otentikasi TLS timbal balik, gunakan *Nama Dibedakan* yang sama yang Anda gunakan saat Anda membuat kunci pribadi.

   ```
   <path-to-your-kafka-installation>/bin/kafka-acls.sh --authorizer-properties
    --bootstrap-server BootstrapServerString --add --allow-principal
    "User:CN=Distinguished-Name" --operation Write --topic Topic-Name
   ```

   Untuk menghapus akses tulis, Anda dapat menjalankan perintah yang sama, menggantinya `--add` dengan`--remove`.

Mengubah grup keamanan klaster MSK Amazon

Halaman ini menjelaskan cara mengubah grup keamanan klaster MSK yang ada. Anda mungkin perlu mengubah grup keamanan klaster untuk menyediakan akses ke kumpulan pengguna tertentu atau membatasi akses ke klaster. Untuk informasi tentang grup keamanan, lihat Grup keamanan untuk VPC Anda di panduan pengguna Amazon VPC.

1. Gunakan ListNodesAPI atau perintah daftar-node di dalam AWS CLI untuk mendapatkan daftar broker di cluster Anda. Hasil operasi ini termasuk ID antarmuka jaringan elastis (ENI) yang terkait dengan broker.
2. Masuk ke AWS Management Console dan buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.

3. Menggunakan daftar dropdown di dekat sudut kanan atas layar, pilih Wilayah tempat cluster digunakan.
4. Di panel kiri, di bawah Jaringan & Keamanan, pilih Antarmuka Jaringan.
5. Pilih ENI pertama yang Anda peroleh pada langkah pertama. Pilih menu Tindakan di bagian atas layar, lalu pilih Ubah Grup Keamanan. Tetapkan grup keamanan baru ke ENI ini. Ulangi langkah ini untuk setiap ENI yang Anda peroleh pada langkah pertama.

> **Note**
>
> Perubahan yang Anda buat pada grup keamanan klaster menggunakan konsol Amazon EC2 tidak tercermin di konsol MSK di bawah Pengaturan jaringan.

6. Konfigurasikan aturan grup keamanan baru untuk memastikan bahwa klien Anda memiliki akses ke broker. Untuk informasi tentang menyetel aturan grup keamanan, lihat Menambahkan, Menghapus, dan Memperbarui Aturan di panduan pengguna Amazon VPC.

> **Important**
>
> Jika Anda mengubah grup keamanan yang terkait dengan broker cluster, dan kemudian menambahkan broker baru ke cluster itu, Amazon MSK mengaitkan broker baru dengan grup keamanan asli yang dikaitkan dengan cluster ketika cluster dibuat. Namun, agar cluster berfungsi dengan benar, semua brokernya harus dikaitkan dengan grup keamanan yang sama. Oleh karena itu, jika Anda menambahkan broker baru setelah mengubah grup keamanan, Anda harus mengikuti prosedur sebelumnya lagi dan memperbarui ENI dari broker baru.

Mengontrol akses ke Apache ZooKeeper

Untuk alasan keamanan, Anda dapat membatasi akses ke ZooKeeper node Apache yang merupakan bagian dari kluster MSK Amazon Anda. Untuk membatasi akses ke node, Anda dapat menetapkan grup keamanan terpisah untuk mereka. Anda kemudian dapat memutuskan siapa yang mendapat akses ke grup keamanan itu.

> **Important**
>
> Bagian ini tidak berlaku untuk cluster yang berjalan dalam mode kRAFT. Lihat the section called “modus kRAFT ”.

Topik ini berisi bagian-bagian berikut:

- Untuk menempatkan ZooKeeper node Apache Anda dalam grup keamanan terpisah
- Menggunakan keamanan TLS dengan Apache ZooKeeper

Untuk menempatkan ZooKeeper node Apache Anda dalam grup keamanan terpisah

1. Dapatkan string ZooKeeper koneksi Apache untuk cluster Anda. Untuk mempelajari caranya, lihat the section called “ZooKeeper modus”. String koneksi berisi nama DNS dari node Apache ZooKeeper Anda.
2. Gunakan alat seperti `host` atau `ping` untuk mengonversi nama DNS yang Anda peroleh pada langkah sebelumnya ke alamat IP. Simpan alamat IP ini karena Anda membutuhkannya nanti dalam prosedur ini.
3. Masuk ke AWS Management Console dan buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
4. Di panel kiri, di bawah NETWORK & SECURITY, pilih Network Interfaces.
5. Di bidang pencarian di atas tabel antarmuka jaringan, ketikkan nama cluster Anda, lalu ketik return. Ini membatasi jumlah antarmuka jaringan yang muncul dalam tabel ke antarmuka yang terkait dengan cluster Anda.
6. Pilih kotak centang di awal baris yang sesuai dengan antarmuka jaringan pertama dalam daftar.
7. Di panel detail di bagian bawah halaman, cari IP IPv4 pribadi Primer. Jika alamat IP ini cocok dengan salah satu alamat IP yang Anda peroleh pada langkah pertama prosedur ini, ini berarti bahwa antarmuka jaringan ini ditetapkan ke ZooKeeper node Apache yang merupakan bagian dari cluster Anda. Jika tidak, batalkan centang kotak di sebelah antarmuka jaringan ini, dan pilih antarmuka jaringan berikutnya dalam daftar. Urutan di mana Anda memilih antarmuka jaringan tidak masalah. Pada langkah selanjutnya, Anda akan melakukan operasi yang sama pada semua antarmuka jaringan yang ditugaskan ke ZooKeeper node Apache, satu per satu.

8. Saat Anda memilih antarmuka jaringan yang sesuai dengan ZooKeeper node Apache, pilih menu Tindakan di bagian atas halaman, lalu pilih Ubah Grup Keamanan. Tetapkan grup keamanan baru ke antarmuka jaringan ini. Untuk informasi tentang membuat grup keamanan, lihat Membuat Grup Keamanan di dokumentasi Amazon VPC.
9. Ulangi langkah sebelumnya untuk menetapkan grup keamanan baru yang sama ke semua antarmuka jaringan yang terkait dengan ZooKeeper node Apache cluster Anda.
10. Anda sekarang dapat memilih siapa yang memiliki akses ke grup keamanan baru ini. Untuk informasi tentang menyetel aturan grup keamanan, lihat Menambahkan, Menghapus, dan Memperbarui Aturan di dokumentasi Amazon VPC.

Menggunakan keamanan TLS dengan Apache ZooKeeper

Anda dapat menggunakan keamanan TLS untuk enkripsi dalam perjalanan antara klien Anda dan node Apache ZooKeeper Anda. Untuk menerapkan keamanan TLS dengan ZooKeeper node Apache Anda, lakukan hal berikut:

- Cluster harus menggunakan Apache Kafka versi 2.5.1 atau yang lebih baru untuk menggunakan keamanan TLS dengan Apache. ZooKeeper
- Aktifkan keamanan TLS saat Anda membuat atau mengonfigurasi klaster Anda. Cluster yang dibuat dengan Apache Kafka versi 2.5.1 atau yang lebih baru dengan TLS diaktifkan secara otomatis menggunakan keamanan TLS dengan titik akhir Apache. ZooKeeper Untuk informasi tentang pengaturan keamanan TLS, lihatBagaimana cara memulai dengan enkripsi?.
- Ambil ZooKeeper titik akhir TLS Apache menggunakan operasi. DescribeCluster
- Buat file ZooKeeper konfigurasi Apache untuk digunakan dengan `kafka-configs.sh` dan `kafka-acls.sh`alat, atau dengan ZooKeeper shell. Dengan setiap alat, Anda menggunakan `--zk-tls-config-file` parameter untuk menentukan ZooKeeper konfigurasi Apache Anda.

  Contoh berikut menunjukkan file ZooKeeper konfigurasi Apache khas:

```
zookeeper.ssl.client.enable=true
zookeeper.clientCnxnSocket=org.apache.zookeeper.ClientCnxnSocketNetty
zookeeper.ssl.keystore.location=kafka.jks
zookeeper.ssl.keystore.password=test1234
zookeeper.ssl.truststore.location=truststore.jks
zookeeper.ssl.truststore.password=test1234
```

- Untuk perintah lain (sepertikafka-topics), Anda harus menggunakan variabel KAFKA_OPTS lingkungan untuk mengkonfigurasi ZooKeeper parameter Apache. Contoh berikut menunjukkan cara mengkonfigurasi variabel KAFKA_OPTS lingkungan untuk meneruskan ZooKeeper parameter Apache ke perintah lain:

```
export KAFKA_OPTS="
-Dzookeeper.clientCnxnSocket=org.apache.zookeeper.ClientCnxnSocketNetty
-Dzookeeper.client.secure=true
-Dzookeeper.ssl.trustStore.location=/home/ec2-user/kafka.client.truststore.jks
-Dzookeeper.ssl.trustStore.password=changeit"
```

  Setelah Anda mengkonfigurasi variabel KAFKA_OPTS lingkungan, Anda dapat menggunakan perintah CLI secara normal. Contoh berikut membuat topik Apache Kafka menggunakan ZooKeeper konfigurasi Apache dari variabel lingkungan: KAFKA_OPTS

```
<path-to-your-kafka-installation>/bin/kafka-topics.sh --create --
zookeeper ZooKeeperTLSConnectString --replication-factor 3 --partitions 1 --topic
 AWSKafkaTutorialTopic
```

> **Note**
>
> Nama-nama parameter yang Anda gunakan dalam file ZooKeeper konfigurasi Apache Anda dan yang Anda gunakan dalam variabel KAFKA_OPTS lingkungan Anda tidak konsisten. Perhatikan nama mana yang Anda gunakan dengan parameter mana dalam file konfigurasi dan variabel KAFKA_OPTS lingkungan Anda.

Untuk informasi selengkapnya tentang mengakses ZooKeeper node Apache Anda dengan TLS, lihat KIP-515: Aktifkan klien ZK untuk menggunakan otentikasi baru yang didukung TLS.

Pencatatan log

Anda dapat mengirimkan log broker Apache Kafka ke satu atau lebih jenis tujuan berikut: Amazon CloudWatch Log, Amazon S3, Amazon Data Firehose. Anda juga dapat mencatat panggilan API MSK Amazon dengan AWS CloudTrail.

Log broker

Log broker memungkinkan Anda untuk memecahkan masalah aplikasi Apache Kafka Anda dan menganalisis komunikasi mereka dengan cluster MSK Anda. Anda dapat mengonfigurasi klaster MSK baru atau yang sudah ada untuk mengirimkan log broker tingkat Info ke satu atau beberapa jenis sumber daya tujuan berikut: grup CloudWatch log, bucket S3, aliran pengiriman Firehose. Melalui Firehose Anda kemudian dapat mengirimkan data log dari aliran pengiriman Anda ke OpenSearch Layanan. Anda harus membuat sumber daya tujuan sebelum mengonfigurasi klaster Anda untuk mengirimkan log broker ke sana. Amazon MSK tidak membuat sumber daya tujuan ini untuk Anda jika belum ada. Untuk informasi tentang ketiga jenis sumber daya tujuan ini dan cara membuatnya, lihat dokumentasi berikut:

- CloudWatch Log Amazon
- Amazon S3
- Amazon Data Firehose

Izin yang diperlukan

Untuk mengonfigurasi tujuan log broker MSK Amazon, identitas IAM yang Anda gunakan untuk tindakan MSK Amazon harus memiliki izin yang dijelaskan dalam kebijakan. AWS kebijakan terkelola: AmazonMSK FullAccess

Untuk melakukan streaming log broker ke bucket S3, Anda juga memerlukan `s3:PutBucketPolicy` izin. Untuk informasi tentang kebijakan bucket S3, lihat Bagaimana Cara Menambahkan Kebijakan Bucket S3? di Panduan Pengguna Amazon S3. Untuk informasi tentang kebijakan IAM secara umum, lihat Manajemen Akses di Panduan Pengguna IAM.

Kebijakan kunci KMS yang diperlukan untuk digunakan dengan bucket SSE-KMS

Jika Anda mengaktifkan enkripsi sisi server untuk bucket S3 menggunakan kunci AWS KMS-managed (SSE-KMS) dengan kunci yang dikelola pelanggan, tambahkan berikut ini ke kebijakan kunci untuk kunci KMS Anda sehingga Amazon MSK dapat menulis file broker ke bucket.

```
{
  "Sid": "Allow Amazon MSK to use the key.",
  "Effect": "Allow",
  "Principal": {
    "Service": [
      "delivery.logs.amazonaws.com"
```

```
    ]
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
  ],
  "Resource": "*"
}
```

Mengkonfigurasi log broker menggunakan AWS Management Console

Jika Anda membuat cluster baru, cari judul pengiriman log Broker di bagian Monitoring. Anda dapat menentukan tujuan yang Anda inginkan Amazon MSK untuk mengirimkan log broker Anda.

Untuk cluster yang ada, pilih cluster dari daftar cluster Anda, lalu pilih tab Properties. Gulir ke bawah ke bagian pengiriman Log dan kemudian pilih tombol Edit. Anda dapat menentukan tujuan yang Anda inginkan Amazon MSK untuk mengirimkan log broker Anda.

Mengkonfigurasi log broker menggunakan AWS CLI

Bila Anda menggunakan `create-cluster` atau `update-monitoring` perintah, Anda dapat secara opsional menentukan `logging-info` parameter dan meneruskannya struktur JSON seperti contoh berikut. Dalam JSON ini, ketiga tipe tujuan adalah opsional.

```
{
  "BrokerLogs": {
    "S3": {
      "Bucket": "ExampleBucketName",
      "Prefix": "ExamplePrefix",
      "Enabled": true
    },
    "Firehose": {
      "DeliveryStream": "ExampleDeliveryStreamName",
      "Enabled": true
    },
    "CloudWatchLogs": {
      "Enabled": true,
      "LogGroup": "ExampleLogGroupName"
    }
  }
```

```
}
```

Mengkonfigurasi log broker menggunakan API

Anda dapat menentukan `loggingInfo` struktur opsional di JSON yang Anda berikan ke UpdateMonitoringoperasi CreateClusteratau.

> **Note**
>
> Secara default, saat pencatatan broker diaktifkan, Amazon MSK mencatat log `INFO` level ke tujuan yang ditentukan. Namun, pengguna Apache Kafka 2.4.X dan yang lebih baru dapat secara dinamis mengatur level log broker ke salah satu level log log4j. Untuk informasi tentang pengaturan level log broker secara dinamis, lihat KIP-412: Memperluas Admin API untuk mendukung level log aplikasi dinamis. Jika Anda menyetel level log secara dinamis ke `DEBUG` atau`TRACE`, sebaiknya gunakan Amazon S3 atau Firehose sebagai tujuan log. Jika Anda menggunakan CloudWatch Log sebagai tujuan log dan Anda mengaktifkan `DEBUG` atau menaikkan `TRACE` level secara dinamis, MSK Amazon dapat terus mengirimkan sampel log. Ini dapat secara signifikan mempengaruhi kinerja broker dan hanya boleh digunakan ketika level `INFO` log tidak cukup bertele-tele untuk menentukan akar penyebab suatu masalah.

Logging panggilan API dengan AWS CloudTrail

> **Note**
>
> AWS CloudTrail log tersedia untuk Amazon MSK hanya ketika Anda menggunakanKontrol akses IAM.

Amazon MSK terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di Amazon MSK. CloudTrail menangkap panggilan API untuk sebagai acara. Panggilan yang diambil termasuk panggilan dari konsol MSK Amazon dan panggilan kode ke operasi Amazon MSK API. Ini juga menangkap tindakan Apache Kafka seperti membuat dan mengubah topik dan grup.

Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara secara berkelanjutan ke bucket Amazon S3, termasuk acara untuk Amazon MSK. Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa terbaru di CloudTrail konsol dalam Riwayat

acara. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat untuk Amazon MSK atau tindakan Apache Kafka, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan.

Untuk mempelajari selengkapnya CloudTrail, termasuk cara mengonfigurasi dan mengaktifkannya, lihat Panduan AWS CloudTrail Pengguna.

Informasi MSK Amazon di CloudTrail

CloudTrail diaktifkan di akun Amazon Web Services Anda saat Anda membuat akun. Ketika aktivitas acara yang didukung terjadi di klaster MSK, aktivitas tersebut direkam dalam suatu CloudTrail peristiwa bersama dengan peristiwa AWS layanan lainnya dalam riwayat Acara. Anda dapat melihat, mencari, dan mengunduh kejadian terbaru di akun Amazon Web Services Anda. Untuk informasi selengkapnya, lihat Melihat Acara dengan Riwayat CloudTrail Acara.

Untuk catatan peristiwa yang sedang berlangsung di akun Amazon Web Services Anda, termasuk acara untuk Amazon MSK, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Secara default, ketika Anda membuat jejak di konsol, jejak diterapkan ke semua Region. Jejak mencatat peristiwa dari semua Wilayah di partisi AWS dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi layanan Amazon lainnya untuk menganalisis lebih lanjut dan menindaklanjuti data peristiwa yang dikumpulkan dalam CloudTrail log. Untuk informasi selengkapnya, lihat berikut:

- Gambaran Umum untuk Membuat Jejak
- CloudTrail Layanan dan Integrasi yang Didukung
- Mengonfigurasi Notifikasi Amazon SNS untuk CloudTrail
- Menerima File CloudTrail Log dari Beberapa Wilayah dan Menerima File CloudTrail Log dari Beberapa Akun

Amazon MSK mencatat semua operasi MSK Amazon sebagai peristiwa dalam file CloudTrail log. Selain itu, ia mencatat tindakan Apache Kafka berikut.

- kafka-cluster: DescribeClusterDynamicConfiguration
- kafka-cluster: AlterClusterDynamicConfiguration
- kafka-cluster: CreateTopic
- kafka-cluster: DescribeTopicDynamicConfiguration
- kafka-cluster: AlterTopic

- kafka-cluster: AlterTopicDynamicConfiguration
- kafka-cluster: DeleteTopic

Setiap entri peristiwa atau log berisi informasi tentang siapa yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan berikut ini:

- Apakah permintaan dibuat dengan pengguna root atau kredensyal pengguna AWS Identity and Access Management (IAM).
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna terfederasi.
- Apakah permintaan itu dibuat oleh AWS layanan lain.

Untuk informasi selengkapnya, lihat Elemen CloudTrail UserIdentity.

Contoh: Entri file log Amazon MSK

Trail adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai file log ke bucket Amazon S3 yang Anda tentukan. CloudTrail file log berisi satu atau lebih entri log. Peristiwa merepresentasikan satu permintaan dari sumber apa pun dan menyertakan informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik dan tindakan Apache Kafka, sehingga file tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan tindakan MSK `DescribeCluster` dan `DeleteCluster` Amazon.


```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "ABCDEF0123456789ABCDE",
        "arn": "arn:aws:iam::012345678901:user/Joe",
        "accountId": "012345678901",
        "accessKeyId": "AIDACKCEVSQ6C2EXAMPLE",
        "userName": "Joe"
      },
      "eventTime": "2018-12-12T02:29:24Z",
```

```
      "eventSource": "kafka.amazonaws.com",
      "eventName": "DescribeCluster",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "aws-cli/1.14.67 Python/3.6.0 Windows/10 botocore/1.9.20",
      "requestParameters": {
        "clusterArn": "arn%3Aaws%3Akafka%3Aus-east-1%3A012345678901%3Acluster
%2Fexamplecluster%2F01234567-abcd-0123-abcd-abcd0123efa-2"
      },
      "responseElements": null,
      "requestID": "bd83f636-fdb5-abcd-0123-157e2fbf2bde",
      "eventID": "60052aba-0123-4511-bcde-3e18dbd42aa4",
      "readOnly": true,
      "eventType": "AwsApiCall",
      "recipientAccountId": "012345678901"
    },
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "ABCDEF0123456789ABCDE",
        "arn": "arn:aws:iam::012345678901:user/Joe",
        "accountId": "012345678901",
        "accessKeyId": "AIDACKCEVSQ6C2EXAMPLE",
        "userName": "Joe"
      },
      "eventTime": "2018-12-12T02:29:40Z",
      "eventSource": "kafka.amazonaws.com",
      "eventName": "DeleteCluster",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "aws-cli/1.14.67 Python/3.6.0 Windows/10 botocore/1.9.20",
      "requestParameters": {
        "clusterArn": "arn%3Aaws%3Akafka%3Aus-east-1%3A012345678901%3Acluster
%2Fexamplecluster%2F01234567-abcd-0123-abcd-abcd0123efa-2"
      },
      "responseElements": {
        "clusterArn": "arn:aws:kafka:us-east-1:012345678901:cluster/
examplecluster/01234567-abcd-0123-abcd-abcd0123efa-2",
        "state": "DELETING"
      },
      "requestID": "c6bfb3f7-abcd-0123-afa5-293519897703",
      "eventID": "8a7f1fcf-0123-abcd-9bdb-1ebf0663a75c",
      "readOnly": false,
```

```
      "eventType": "AwsApiCall",
      "recipientAccountId": "012345678901"
    }
  ]
}
```

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan `kafka-cluster:CreateTopic` tindakan.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "ABCDEFGH1IJKLMN2P34Q5",
    "arn": "arn:aws:iam::111122223333:user/Admin",
    "accountId": "111122223333",
    "accessKeyId": "CDEFAB1C2UUUUU3AB4TT",
    "userName": "Admin"
  },
  "eventTime": "2021-03-01T12:51:19Z",
  "eventSource": "kafka-cluster.amazonaws.com",
  "eventName": "CreateTopic",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "198.51.100.0/24",
  "userAgent": "aws-msk-iam-auth/unknown-version/aws-internal/3 aws-sdk-java/1.11.970
 Linux/4.14.214-160.339.amzn2.x86_64 OpenJDK_64-Bit_Server_VM/25.272-b10 java/1.8.0_272
 scala/2.12.8 vendor/Red_Hat,_Inc.",
  "requestParameters": {
    "kafkaAPI": "CreateTopics",
    "resourceARN": "arn:aws:kafka:us-east-1:111122223333:topic/IamAuthCluster/3ebafd8e-
dae9-440d-85db-4ef52679674d-1/Topic9"
  },
  "responseElements": null,
  "requestID": "e7c5e49f-6aac-4c9a-a1d1-c2c46599f5e4",
  "eventID": "be1f93fd-4f14-4634-ab02-b5a79cb833d2",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "111122223333"
}
```

Validasi kepatuhan untuk Amazon Managed Streaming for Apache Kafka

Auditor pihak ketiga menilai keamanan dan kepatuhan Amazon Managed Streaming for Apache Kafka sebagai bagian dari program kepatuhan. AWS Ini termasuk PCI dan HIPAA BAA.

Untuk daftar AWS layanan dalam lingkup program kepatuhan tertentu, lihat Layanan Amazon dalam Lingkup menurut Program Kepatuhan . 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 Amazon MSK ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, serta undang-undang dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- Panduan Quick Start Keamanan dan Kepatuhan – Panduan deployment ini membahas pertimbangan arsitektur dan menyediakan langkah–langkah untuk melakukan deployment terhadap lingkungan dasar di AWS yang menjadi fokus keamanan dan kepatuhan.
- Arsitektur untuk Whitepaper Keamanan dan Kepatuhan HIPAA — Whitepaper ini menjelaskan bagaimana perusahaan dapat menggunakan untuk membuat aplikasi yang sesuai dengan HIPAA. AWS
- AWS Sumber Daya AWS — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- Mengevaluasi Sumber Daya dengan Aturan dalam Panduan AWS Config Pengembang — AWS Config Layanan menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- AWS Security Hub— AWS Layanan ini memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS yang membantu Anda memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik.

Ketahanan di Amazon Managed Streaming untuk Apache Kafka Kafka

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 Zona Ketersediaan, Anda dapat merancang serta mengoperasikan aplikasi dan basis data yang secara otomatis melakukan fail over 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 di Amazon Managed Streaming for Apache Kafka

Sebagai layanan terkelola, Amazon Managed Streaming for Apache Kafka dilindungi oleh prosedur keamanan jaringan global AWS yang dijelaskan dalam whitepaper Amazon Web Services: Overview of Security Processes.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses Amazon MSK melalui jaringan. Klien harus mendukung Keamanan Lapisan Pengangkutan (TLS) 1.0 atau versi yang lebih baru. Kami merekomendasikan TLS 1.2 atau versi yang lebih baru. Klien juga harus mendukung suite cipher dengan perfect forward secrecy (PFS) seperti Ephemeral Diffie-Hellman (DHE) atau Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Sebagian besar sistem modern seperti Java 7 dan sistem yang lebih baru mendukung mode ini.

Selain itu, permintaan harus ditandatangani menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan prinsipal IAM. Atau Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk menghasilkan kredensial keamanan sementara untuk menandatangani permintaan.

Menghubungkan ke cluster MSK Amazon

Secara default, klien dapat mengakses kluster MSK hanya jika mereka berada di VPC yang sama dengan cluster. Semua komunikasi antara klien Kafka Anda dan kluster MSK Anda bersifat pribadi secara default dan data streaming Anda tidak pernah melintasi internet. Untuk terhubung ke klaster MSK Anda dari klien yang berada di VPC yang sama dengan cluster, pastikan grup keamanan klaster memiliki aturan masuk yang menerima lalu lintas dari grup keamanan klien. Untuk informasi tentang mengatur aturan ini, lihat Aturan Grup Keamanan. Untuk contoh cara mengakses cluster dari instans Amazon EC2 yang berada di VPC yang sama dengan cluster, lihat. Memulai

Untuk terhubung ke klaster MSK Anda dari klien yang berada di luar VPC cluster, lihat Akses dari AWS dalam tetapi di luar VPC cluster.

Topik

- Akses publik
- Akses dari dalam AWS tetapi di luar VPC cluster

Akses publik

Amazon MSK memberi Anda opsi untuk mengaktifkan akses publik ke broker cluster MSK yang menjalankan Apache Kafka 2.6.0 atau versi yang lebih baru. Untuk alasan keamanan, Anda tidak dapat mengaktifkan akses publik saat membuat klaster MSK. Namun, Anda dapat memperbarui cluster yang ada agar dapat diakses publik. Anda juga dapat membuat cluster baru dan kemudian memperbaruinya agar dapat diakses publik.

Anda dapat mengaktifkan akses publik ke kluster MSK tanpa biaya tambahan, tetapi biaya transfer AWS data standar berlaku untuk transfer data masuk dan keluar dari cluster. Untuk informasi tentang harga, lihat Harga Sesuai Permintaan Amazon EC2.

Untuk mengaktifkan akses publik ke klaster, pertama-tama pastikan bahwa klaster memenuhi semua kondisi berikut:

- Subnet yang terkait dengan cluster harus bersifat publik. Ini berarti bahwa subnet harus memiliki tabel rute terkait dengan gateway internet terpasang. Untuk informasi tentang cara membuat dan melampirkan gateway internet, lihat gateway Internet di panduan pengguna Amazon VPC.

- Kontrol akses yang tidak diautentikasi harus dimatikan dan setidaknya satu dari metode kontrol akses berikut harus aktif: SASL/IAM, SASL/SCRAM, mTLS. Untuk informasi tentang cara memperbarui metode kontrol akses klaster, lihat. the section called “Memperbarui keamanan”
- Enkripsi dalam cluster harus dihidupkan. Pengaturan on adalah default saat membuat cluster. Tidak mungkin mengaktifkan enkripsi di dalam cluster untuk cluster yang dibuat dengan dimatikan. Oleh karena itu tidak mungkin untuk mengaktifkan akses publik untuk cluster yang dibuat dengan enkripsi dalam cluster dimatikan.
- Lalu lintas teks biasa antara broker dan klien harus dimatikan. Untuk informasi tentang cara mematikannya jika aktif, lihatthe section called “Memperbarui keamanan”.
- Jika Anda menggunakan metode kontrol akses SASL/SCRAM atau mTLS, Anda harus mengatur Apache Kafka ACL untuk cluster Anda. Setelah Anda menyetel ACL Apache Kafka untuk klaster Anda, perbarui konfigurasi cluster agar properti `allow.everyone.if.no.acl.found` menjadi false untuk cluster. Untuk informasi tentang cara memperbarui konfigurasi klaster, lihatthe section called “Operasi konfigurasi”. Jika Anda menggunakan kontrol akses IAM dan ingin menerapkan kebijakan otorisasi atau memperbarui kebijakan otorisasi Anda, lihat. the section called “Kontrol akses IAM” Untuk informasi tentang Apache Kafka ACL, lihat. the section called “Apache Kafka ACL”

Setelah Anda memastikan bahwa klaster MSK memenuhi ketentuan yang tercantum di atas, Anda dapat menggunakan AWS Management Console, API MSK Amazon AWS CLI, atau Amazon untuk mengaktifkan akses publik. Setelah Anda mengaktifkan akses publik ke cluster, Anda bisa mendapatkan string bootstrap-broker publik untuk itu. Untuk informasi tentang mendapatkan broker bootstrap untuk sebuah cluster, lihatthe section called “Mendapatkan broker bootstrap”.

> **Important**
>
> Selain mengaktifkan akses publik, pastikan bahwa grup keamanan klaster memiliki aturan TCP masuk yang memungkinkan akses publik dari alamat IP Anda. Kami menyarankan Anda membuat aturan ini seketat mungkin. Untuk informasi tentang grup keamanan dan aturan masuk, lihat Grup keamanan untuk VPC Anda di Panduan Pengguna Amazon VPC. Untuk nomor port, lihatthe section called “Informasi pelabuhan”. Untuk petunjuk tentang cara mengubah grup keamanan klaster, lihatthe section called “Mengubah grup keamanan”.

> **Note**
>
> Jika Anda menggunakan petunjuk berikut untuk mengaktifkan akses publik dan kemudian masih tidak dapat mengakses cluster, lihatthe section called “Tidak dapat mengakses klaster yang mengaktifkan akses publik”.

Mengaktifkan akses publik menggunakan konsol

1. Masuk ke AWS Management Console, dan buka konsol MSK Amazon di https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Dalam daftar cluster, pilih cluster yang ingin Anda aktifkan akses publik.
3. Pilih tab Properties, lalu temukan bagian Pengaturan jaringan.
4. Pilih Edit akses publik.

Mengaktifkan akses publik menggunakan AWS CLI

1. Jalankan AWS CLI perintah berikut, ganti *ClusterArn*dan *Current-Cluster-Version dengan ARN dan versi* cluster saat ini. Untuk menemukan versi cluster saat ini, gunakan DescribeClusteroperasi atau perintah AWS CLI deskripsi-cluster. Contoh versi adalahKTVPDKIKX0DER.

   ```
   aws kafka update-connectivity --cluster-arn ClusterArn --current-
   version Current-Cluster-Version --connectivity-info '{"PublicAccess": {"Type":
    "SERVICE_PROVIDED_EIPS"}}'
   ```

   Output dari `update-connectivity` perintah ini terlihat seperti contoh JSON berikut.

   ```
   {
       "ClusterArn": "arn:aws:kafka:us-east-1:012345678012:cluster/exampleClusterName/
   abcdefab-1234-abcd-5678-cdef0123ab01-2",
       "ClusterOperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-
   operation/exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
   abcd-4f7f-1234-9876543210ef"
   }
   ```

> **Note**
>
> Untuk mematikan akses publik, gunakan AWS CLI perintah serupa, tetapi dengan info konektivitas berikut sebagai gantinya:
>
> ```
> '{"PublicAccess": {"Type": "DISABLED"}}'
> ```

2. Untuk mendapatkan hasil `update-connectivity` operasi, jalankan perintah berikut, ganti *`ClusterOperationArn`* dengan ARN yang Anda peroleh dalam output perintah. `update-connectivity`

```
aws kafka describe-cluster-operation --cluster-operation-arn ClusterOperationArn
```

Output dari `describe-cluster-operation` perintah ini terlihat seperti contoh JSON berikut.

```
{
    "ClusterOperationInfo": {
        "ClientRequestId": "982168a3-939f-11e9-8a62-538df00285db",
        "ClusterArn": "arn:aws:kafka:us-east-1:012345678012:cluster/
exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2",
        "CreationTime": "2019-06-20T21:08:57.735Z",
        "OperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-
operation/exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
abcd-4f7f-1234-9876543210ef",
        "OperationState": "UPDATE_COMPLETE",
        "OperationType": "UPDATE_CONNECTIVITY",
        "SourceClusterInfo": {
            "ConnectivityInfo": {
                "PublicAccess": {
                    "Type": "DISABLED"
                }
            }
        },
        "TargetClusterInfo": {
            "ConnectivityInfo": {
                "PublicAccess": {
                    "Type": "SERVICE_PROVIDED_EIPS"
                }
            }
        }
```

```
    }
}
```

Jika `OperationState` memiliki nilai`UPDATE_IN_PROGRESS`, tunggu sebentar, lalu jalankan `describe-cluster-operation` perintah lagi.

Mengaktifkan akses publik menggunakan Amazon MSK API

- Untuk menggunakan API untuk mengaktifkan atau menonaktifkan akses publik ke klaster, lihat UpdateConnectivity.

> **Note**
>
> Untuk alasan keamanan, Amazon MSK tidak mengizinkan akses publik ke node pengontrol Apache ZooKeeper atau kRAFT.

Akses dari dalam AWS tetapi di luar VPC cluster

Untuk terhubung ke cluster MSK dari dalam AWS tetapi di luar VPC Amazon cluster, ada opsi berikut.

Pengintip VPC Amazon

Untuk terhubung ke cluster MSK Anda dari VPC yang berbeda dari VPC cluster, Anda dapat membuat koneksi peering antara dua VPC. Untuk informasi tentang peering VPC, lihat Panduan Peering VPC Amazon.

AWS Direct Connect

AWS Direct Connect menghubungkan jaringan on-premise Anda ke AWS lebih dari kabel serat optik Ethernet 1 gigabit atau 10 gigabit standar. Salah satu ujung kabel terhubung ke router Anda, yang lain ke AWS Direct Connect router. Dengan koneksi ini, Anda dapat membuat antarmuka virtual langsung ke AWS cloud dan Amazon VPC, melewati penyedia layanan Internet di jalur jaringan Anda. Untuk informasi selengkapnya, lihat AWS Direct Connect.

AWS Transit Gateway

AWS Transit Gateway adalah layanan yang memungkinkan Anda menghubungkan VPC dan jaringan lokal Anda ke satu gateway. Untuk informasi tentang cara menggunakan AWS Transit Gateway, lihat AWS Transit Gateway.

Koneksi VPN

Anda dapat menghubungkan VPC klaster MSK Anda ke jaringan jarak jauh dan pengguna menggunakan opsi konektivitas VPN yang dijelaskan dalam topik berikut: Koneksi VPN.

Proksi REST

Anda dapat menginstal proxy REST pada instance yang berjalan di dalam Amazon VPC klaster Anda. Proksi REST memungkinkan awproducer dan konsumen Anda untuk berkomunikasi dengan cluster melalui permintaan HTTP API.

Konektivitas Multi-VPC Beberapa Wilayah

Dokumen berikut menjelaskan opsi konektivitas untuk beberapa VPC yang berada di Wilayah yang berbeda: Konektivitas Multi-VPC Multiple Region.

Konektivitas pribadi multi-VPC Wilayah Tunggal

Konektivitas pribadi multi-VPC (didukung oleh AWS PrivateLink) untuk Amazon Managed Streaming for Apache Kafka (Amazon MSK) cluster adalah fitur yang memungkinkan Anda untuk lebih cepat menghubungkan klien Kafka yang dihosting di berbagai Virtual Private Clouds (VPC AWS) dan akun ke kluster MSK Amazon.

Lihat konektivitas multi-VPC Wilayah Tunggal untuk klien lintas akun.

Jaringan EC2-Classic sudah pensiun

Amazon MSK tidak lagi mendukung instans Amazon EC2 yang berjalan dengan jaringan Amazon EC2-Classic.

Lihat EC2-Classic Networking Pensiun — Inilah Cara Mempersiapkan.

Amazon MSK Multi-VPC konektivitas pribadi dalam satu Wilayah

Konektivitas pribadi multi-VPC (didukung oleh AWS PrivateLink) untuk Amazon Managed Streaming for Apache Kafka (Amazon MSK) cluster adalah fitur yang memungkinkan Anda untuk lebih cepat menghubungkan klien Kafka yang dihosting di berbagai Virtual Private Clouds (VPC AWS) dan akun ke kluster MSK Amazon.

Konektivitas pribadi multi-VPC adalah solusi terkelola yang menyederhanakan infrastruktur jaringan untuk konektivitas multi-VPC dan lintas akun. Klien dapat terhubung ke cluster MSK Amazon PrivateLink sambil menjaga semua lalu lintas dalam AWS jaringan. Konektivitas pribadi multi-VPC untuk kluster MSK Amazon tersedia di semua Wilayah di AWS mana Amazon MSK tersedia.

Topik

Apa itu konektivitas pribadi multi-VPC?

Konektivitas pribadi multi-VPC untuk Amazon MSK adalah opsi konektivitas yang memungkinkan Anda menghubungkan klien Apache Kafka yang di-host di berbagai Virtual Private Clouds (VPC) dan AWS akun ke cluster MSK.

Amazon MSK menyederhanakan akses lintas akun dengan kebijakan klaster. Kebijakan ini memungkinkan pemilik klaster untuk memberikan izin bagi AWS akun lain untuk membangun konektivitas pribadi ke klaster MSK.

Manfaat konektivitas pribadi multi-VPC

Konektivitas pribadi multi-VPC memiliki beberapa keunggulan dibandingkan solusi konektivitas lainnya:

- Ini mengotomatiskan manajemen operasional solusi AWS PrivateLink konektivitas.
- Ini memungkinkan IP yang tumpang tindih di seluruh VPC yang menghubungkan, menghilangkan kebutuhan untuk mempertahankan IP yang tidak tumpang tindih, peering kompleks, dan tabel perutean yang terkait dengan solusi konektivitas VPC lainnya.

Anda menggunakan kebijakan klaster untuk klaster MSK Anda untuk menentukan AWS akun mana yang memiliki izin untuk menyiapkan konektivitas pribadi lintas akun ke kluster MSK Anda. Admin lintas akun dapat mendelegasikan izin ke peran atau pengguna yang sesuai. Saat digunakan dengan otentikasi klien IAM, Anda juga dapat menggunakan kebijakan klaster untuk menentukan izin bidang data Kafka secara terperinci untuk klien yang menghubungkan.

Persyaratan dan batasan untuk konektivitas pribadi multi-VPC

Perhatikan persyaratan klaster MSK ini untuk menjalankan konektivitas pribadi multi-VPC:

- Konektivitas pribadi multi-VPC hanya didukung pada Apache Kafka 2.7.1 atau lebih tinggi. Pastikan bahwa setiap klien yang Anda gunakan dengan kluster MSK menjalankan versi Apache Kafka yang kompatibel dengan cluster.
- Konektivitas pribadi multi-VPC mendukung jenis autentikasi IAM, TLS dan SASL/SCRAM. Cluster yang tidak diautentikasi tidak dapat menggunakan konektivitas pribadi multi-VPC.
- Jika Anda menggunakan metode kontrol akses SASL/SCRAM atau mTLS, Anda harus mengatur Apache Kafka ACL untuk cluster Anda. Pertama, atur Apache Kafka ACL untuk cluster Anda. Kemudian, perbarui konfigurasi cluster agar properti `allow.everyone.if.no.acl.found` disetel ke false untuk cluster. Untuk informasi tentang cara memperbarui konfigurasi klaster, lihatthe section called “Operasi konfigurasi”. Jika Anda menggunakan kontrol akses IAM dan ingin menerapkan kebijakan otorisasi atau memperbarui kebijakan otorisasi Anda, lihat. the section called “Kontrol akses IAM” Untuk informasi tentang Apache Kafka ACL, lihat. the section called “Apache Kafka ACL”
- Konektivitas pribadi multi-vPC tidak mendukung tipe instans t3.small.
- Konektivitas pribadi multi-VPC tidak didukung di seluruh AWS Wilayah, hanya pada AWS akun dalam Wilayah yang sama.
- Amazon MSK tidak mendukung konektivitas pribadi multi-VPC ke node Zookeeper.

Memulai menggunakan konektivitas pribadi multi-VPC

Topik

- Langkah 1: Pada kluster MSK di Akun A, aktifkan konektivitas multi-VPC untuk skema autentikasi IAM di cluster
- Langkah 2: Lampirkan kebijakan klaster ke klaster MSK
- Langkah 3: Tindakan pengguna lintas akun untuk mengonfigurasi koneksi VPC yang dikelola klien

Tutorial ini menggunakan kasus penggunaan umum sebagai contoh bagaimana Anda dapat menggunakan konektivitas multi-VPC untuk menghubungkan klien Apache Kafka secara pribadi ke cluster MSK dari dalam, AWS tetapi di luar VPC cluster. Proses ini mengharuskan pengguna lintas akun untuk membuat koneksi dan konfigurasi VPC yang dikelola MSK untuk setiap klien, termasuk izin klien yang diperlukan. Proses ini juga mengharuskan pemilik klaster MSK untuk mengaktifkan PrivateLink konektivitas pada cluster MSK dan memilih skema otentikasi untuk mengontrol akses ke cluster.

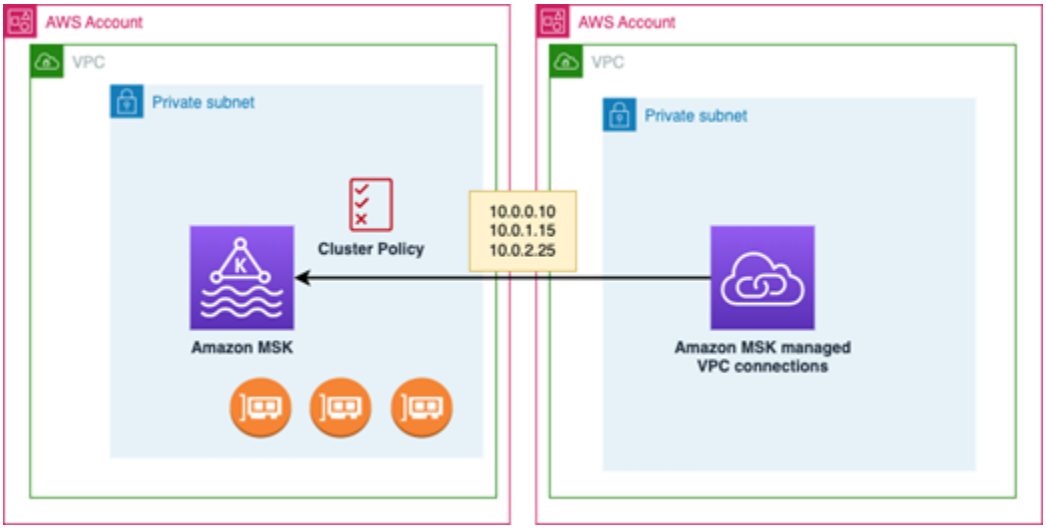
Di berbagai bagian tutorial ini, kami memilih opsi yang berlaku untuk contoh ini. Ini tidak berarti bahwa mereka adalah satu-satunya opsi yang berfungsi untuk menyiapkan cluster MSK atau instance klien.

Konfigurasi jaringan untuk kasus penggunaan ini adalah sebagai berikut:

- Pengguna lintas akun (klien Kafka) dan cluster MSK berada di AWS jaringan/wilayah yang sama, tetapi di akun yang berbeda:
  - Kluster MSK di Akun A
  - Klien Kafka di Akun B
- Pengguna lintas akun akan terhubung secara pribadi ke kluster MSK menggunakan skema autentikasi IAM.

Tutorial ini mengasumsikan bahwa ada cluster MSK yang disediakan dibuat dengan Apache Kafka versi 2.7.1 atau lebih tinggi. Cluster MSK harus dalam keadaan AKTIF sebelum memulai proses konfigurasi. Untuk menghindari potensi kehilangan data atau downtime, klien yang akan menggunakan koneksi pribadi multi-VPC untuk terhubung ke cluster harus menggunakan versi Apache Kafka yang kompatibel dengan cluster.

Diagram berikut menggambarkan arsitektur konektivitas Amazon MSK multi-VPC yang terhubung ke klien di akun yang berbeda. AWS



Langkah 1: Pada kluster MSK di Akun A, aktifkan konektivitas multi-VPC untuk skema autentikasi IAM di cluster

Pemilik cluster MSK perlu membuat pengaturan konfigurasi pada cluster MSK setelah cluster dibuat dan dalam keadaan AKTIF.

Pemilik cluster mengaktifkan konektivitas pribadi multi-VPC di cluster ACTIVE untuk skema autentikasi apa pun yang akan aktif di cluster. Ini dapat dilakukan dengan menggunakan konsol [UpdateSecurity API](#) atau MSK. Skema autentikasi IAM, SASL/SCRAM, dan TLS mendukung konektivitas pribadi multi-VPC. Konektivitas pribadi multi-VPC tidak dapat diaktifkan untuk cluster yang tidak diautentikasi.

Untuk kasus penggunaan ini, Anda akan mengonfigurasi cluster untuk menggunakan skema autentikasi IAM.

> **Note**
>
> Jika Anda mengonfigurasi cluster MSK Anda untuk menggunakan skema autentikasi SASL/SCRAM, properti Apache Kafka ACL "" adalah wajib. `allow.everyone.if.no.acl.found=false` Lihat [Apache Kafka ACL](#).

Saat Anda memperbarui pengaturan konektivitas pribadi multi-VPC, Amazon MSK memulai reboot bergulir node broker yang memperbarui konfigurasi broker. Ini bisa memakan waktu hingga 30 menit atau lebih untuk menyelesaikannya. Anda tidak dapat membuat pembaruan lain ke cluster saat konektivitas sedang diperbarui.

Aktifkan multi-VPC untuk skema autentikasi yang dipilih pada cluster di Akun A menggunakan konsol

1. Buka konsol MSK Amazon di https://console.aws.amazon.com/msk/ untuk akun tempat cluster berada.
2. Di panel navigasi, di bawah MSK Clusters, pilih Cluster untuk menampilkan daftar cluster di akun.
3. Pilih cluster yang akan dikonfigurasi untuk konektivitas pribadi multi-VPC. Cluster harus dalam keadaan AKTIF.
4. Pilih tab Cluster Properties, dan kemudian pergi ke Pengaturan jaringan.
5. Pilih menu tarik-turun Edit dan pilih Aktifkan konektivitas multi-VPC.
6. Pilih satu atau beberapa jenis otentikasi yang ingin Anda aktifkan untuk klaster ini. Untuk kasus penggunaan ini, pilih autentikasi berbasis peran IAM.
7. Pilih Simpan perubahan.

Example - UpdateConnectivity API yang mengaktifkan skema autentikasi konektivitas pribadi multi-VPC pada sebuah cluster

Sebagai alternatif dari konsol MSK, Anda dapat menggunakan UpdateConnectivity API untuk mengaktifkan konektivitas pribadi multi-VPC dan mengonfigurasi skema autentikasi pada cluster AKTIF. Contoh berikut menunjukkan skema autentikasi IAM diaktifkan untuk cluster.

```
{
  "currentVersion": "K3T4TT2Z381HKD",
  "connectivityInfo": {
    "vpcConnectivity": {
      "clientAuthentication": {
        "sasl": {
          "iam": {
            "enabled": TRUE
            }
        }
      }
    }
  }
}
```

Amazon MSK menciptakan infrastruktur jaringan yang diperlukan untuk konektivitas pribadi. Amazon MSK juga membuat satu set titik akhir broker bootstrap baru untuk setiap jenis autentikasi yang

memerlukan konektivitas pribadi. Perhatikan bahwa skema autentikasi plaintext tidak mendukung konektivitas pribadi multi-VPC.

Langkah 2: Lampirkan kebijakan klaster ke klaster MSK

Pemilik klaster dapat melampirkan kebijakan klaster (juga dikenal sebagai kebijakan berbasis sumber daya) ke klaster MSK di mana Anda akan mengaktifkan konektivitas pribadi multi-VPC. Kebijakan klaster memberikan izin kepada klien untuk mengakses klaster dari akun lain. Sebelum Anda dapat mengedit kebijakan klaster, Anda memerlukan ID akun untuk akun yang harus memiliki izin untuk mengakses kluster MSK. Lihat Bagaimana Amazon MSK bekerja dengan IAM.

Pemilik klaster harus melampirkan kebijakan klaster ke klaster MSK yang memberi wewenang kepada pengguna lintas akun di Akun B untuk mendapatkan broker bootstrap untuk klaster dan untuk mengotorisasi tindakan berikut pada klaster MSK di Akun A:

- CreateVpcKoneksi
- GetBootstrapPialang
- DescribeCluster
- DescribeClusterV2

Example

Sebagai referensi, berikut ini adalah contoh JSON untuk kebijakan klaster dasar, mirip dengan kebijakan default yang ditampilkan di editor kebijakan IAM konsol MSK.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "123456789012"
                ]
            },
            "Action": [
                "kafka:CreateVpcConnection",
                "kafka:GetBootstrapBrokers",
                "kafka:DescribeCluster",
```

```
                "kafka:DescribeClusterV2"
            ],
            "Resource": "arn:aws:kafka:us-east-1:123456789012:cluster/testing/
de8982fa-8222-4e87-8b20-9bf3cdfa1521-2"
        }
    ]
}
```

Lampirkan kebijakan klaster ke klaster MSK

1. Di konsol MSK Amazon, di bawah MSK Clusters, pilih Cluster.
2. Gulir ke bawah ke Pengaturan keamanan dan pilih Edit kebijakan klaster.
3. Di konsol, di layar Edit Kebijakan Cluster, pilih Kebijakan dasar untuk konektivitas multi-VPC.
4. Di bidang ID Akun, masukkan ID akun untuk setiap akun yang seharusnya memiliki izin untuk mengakses klaster ini. Saat Anda mengetik ID, ID secara otomatis disalin ke dalam sintaks JSON kebijakan yang ditampilkan. Dalam contoh kebijakan klaster kami, ID Akun adalah`123456789012`.
5. Pilih Simpan perubahan.

Untuk informasi tentang API kebijakan klaster, lihat kebijakan berbasis sumber daya MSK Amazon.

Langkah 3: Tindakan pengguna lintas akun untuk mengonfigurasi koneksi VPC yang dikelola klien

Untuk mengatur konektivitas pribadi multi-VPC antara klien di akun yang berbeda dari kluster MSK, pengguna lintas akun membuat koneksi VPC terkelola untuk klien. Beberapa klien dapat dihubungkan ke cluster MSK dengan mengulangi prosedur ini. Untuk keperluan kasus penggunaan ini, Anda akan mengkonfigurasi hanya satu klien.

Klien dapat menggunakan skema autentikasi yang didukung IAM, SASL/SCRAM, atau TLS. Setiap koneksi VPC yang dikelola hanya dapat memiliki satu skema autentikasi yang terkait dengannya. Skema autentikasi klien harus dikonfigurasi pada cluster MSK tempat klien akan terhubung.

Untuk kasus penggunaan ini, konfigurasikan skema autentikasi klien sehingga klien di Akun B menggunakan skema autentikasi IAM.

Prasyarat

Proses ini membutuhkan item berikut:

- Kebijakan klaster yang dibuat sebelumnya yang memberikan klien di Akun B izin untuk melakukan tindakan pada klaster MSK di Akun A.
- Kebijakan identitas yang dilampirkan pada klien di Akun B yang memberikan izin untuk`kafka:CreateVpcConnection`,`ec2:CreateTags`, `ec2:CreateVPCEndpoint` dan `ec2:DescribeVpcAttribute` tindakan.

Example

Sebagai referensi, berikut ini adalah contoh JSON untuk kebijakan identitas klien dasar.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kafka:CreateVpcConnection",
        "ec2:CreateTags",
        "ec2:CreateVPCEndpoint",
        "ec2:DescribeVpcAttribute"
      ],
      "Resource": "*"
    }
  ]
}
```

Untuk membuat koneksi VPC terkelola untuk klien di Akun B

1. Dari administrator cluster, dapatkan ARN Cluster dari klaster MSK di Akun A yang Anda inginkan klien di Akun B untuk terhubung. Catat ARN cluster yang akan digunakan nanti.
2. Di konsol MSK untuk klien Akun B, pilih Koneksi VPC Terkelola, lalu pilih Buat koneksi.
3. Di panel Pengaturan koneksi, tempel ARN cluster ke bidang teks ARN cluster, lalu pilih Verifikasi.
4. Pilih jenis otentikasi untuk klien di Akun B. Untuk kasus penggunaan ini, pilih IAM saat membuat koneksi VPC klien.
5. Pilih VPC untuk klien.
6. Pilih setidaknya dua Availability Zone dan Subnet terkait. Anda bisa mendapatkan ID zona ketersediaan dari detail klaster AWS Management Console atau dengan menggunakan

DescribeClusterAPI atau perintah AWS CLI klaster deskripsikan. ID zona yang Anda tentukan untuk subnet klien harus cocok dengan subnet cluster. Jika nilai untuk subnet hilang, pertama-tama buat subnet dengan ID zona yang sama dengan cluster MSK Anda.

7. Pilih grup Keamanan untuk koneksi VPC ini. Anda dapat menggunakan grup keamanan default. Untuk informasi selengkapnya tentang mengonfigurasi grup keamanan, lihat Mengontrol lalu lintas ke sumber daya menggunakan grup keamanan.
8. Pilih Buat koneksi.
9. Untuk mendapatkan daftar string broker bootstrap baru dari konsol MSK pengguna lintas akun (Detail klaster > Koneksi VPC Terkelola), lihat string broker bootstrap yang ditampilkan di bawah “String koneksi cluster.” Dari Akun B klien, daftar broker bootstrap dapat dilihat dengan memanggil GetBootstrapBroker API atau dengan melihat daftar broker bootstrap di detail cluster konsol.
10. Perbarui grup keamanan yang terkait dengan koneksi VPC sebagai berikut:

    a. Tetapkan aturan masuk untuk PrivateLink VPC untuk mengizinkan semua lalu lintas untuk rentang IP dari jaringan Akun B.

    b. [Opsional] Tetapkan konektivitas aturan Outbound ke cluster MSK. Pilih Grup Keamanan di konsol VPC, Edit Aturan Keluar, dan tambahkan aturan untuk Lalu Lintas TCP Kustom untuk rentang port 14001-14100. Penyeimbang beban jaringan multi-VPC mendengarkan pada rentang port 14001-14100. Lihat Network Load Balancers.
11. Konfigurasikan klien di Akun B untuk menggunakan broker bootstrap baru untuk konektivitas pribadi multi-VPC untuk terhubung ke cluster MSK di Akun A. Lihat Menghasilkan dan mengkonsumsi data.

Setelah otorisasi selesai, Amazon MSK membuat koneksi VPC terkelola untuk setiap skema VPC dan autentikasi yang ditentukan. Grup keamanan yang dipilih dikaitkan dengan setiap koneksi. Koneksi VPC terkelola ini dikonfigurasi oleh Amazon MSK untuk terhubung secara pribadi ke broker. Anda dapat menggunakan kumpulan broker bootstrap baru untuk terhubung secara pribadi ke cluster MSK Amazon.

Perbarui skema otorisasi pada cluster

Konektivitas pribadi multi-VPC mendukung beberapa skema otorisasi: SASL/SCRAM, IAM, dan TLS. Pemilik cluster dapat mengaktifkan/menonaktifkan konektivitas pribadi untuk satu atau beberapa skema autentikasi. Cluster harus dalam keadaan AKTIF untuk melakukan tindakan ini.

Untuk mengaktifkan skema autentikasi menggunakan konsol MSK Amazon

1. Buka konsol MSK Amazon di AWS Management Consoleuntuk cluster yang ingin Anda edit.
2. Di panel navigasi, di bawah MSK Clusters, pilih Cluster untuk menampilkan daftar cluster di akun.
3. Pilih cluster yang ingin Anda edit. Cluster harus dalam keadaan AKTIF.
4. Pilih tab Cluster Properties, dan kemudian pergi ke Pengaturan jaringan.
5. Pilih menu tarik-turun Edit dan pilih Aktifkan konektivitas multi-VPC untuk mengaktifkan skema autentikasi baru.
6. Pilih satu atau beberapa jenis otentikasi yang ingin Anda aktifkan untuk klaster ini.
7. Pilih Aktifkan pilihan.

Saat Anda mengaktifkan skema autentikasi baru, Anda juga harus membuat koneksi VPC terkelola baru untuk skema autentikasi baru dan memperbarui klien Anda untuk menggunakan broker bootstrap khusus untuk skema autentikasi baru.

Untuk mematikan skema autentikasi menggunakan konsol MSK Amazon

> **Note**
>
> Saat Anda mematikan konektivitas pribadi multi-VPC untuk skema autentikasi, semua infrastruktur terkait konektivitas, termasuk koneksi VPC terkelola, akan dihapus.

Saat Anda mematikan konektivitas pribadi multi-VPC untuk skema autentikasi, koneksi VPC yang ada di sisi klien berubah menjadi INACTIVE, dan infrastruktur Privatelink di sisi cluster, termasuk koneksi VPC terkelola, di sisi cluster dihapus. Pengguna lintas akun hanya dapat menghapus koneksi VPC yang tidak aktif. Jika konektivitas pribadi diaktifkan lagi di cluster, pengguna lintas akun perlu membuat koneksi baru ke cluster.

1. Buka konsol MSK Amazon di AWS Management Console.
2. Di panel navigasi, di bawah MSK Clusters, pilih Cluster untuk menampilkan daftar cluster di akun.
3. Pilih cluster yang ingin Anda edit. Cluster harus dalam keadaan AKTIF.
4. Pilih tab Cluster Properties, lalu pergi ke Pengaturan jaringan.

5. Pilih menu tarik-turun Edit dan pilih Matikan konektivitas multi-VPC (untuk mematikan skema autentikasi).
6. Pilih satu atau beberapa jenis otentikasi yang ingin dimatikan untuk klaster ini.
7. Pilih Matikan pilihan.

Example Untuk mengaktifkan/menonaktifkan skema autentikasi dengan API

Sebagai alternatif dari konsol MSK, Anda dapat menggunakan UpdateConnectivity API untuk mengaktifkan konektivitas pribadi multi-VPC dan mengonfigurasi skema autentikasi pada cluster AKTIF. Contoh berikut menunjukkan skema autentikasi SASL/SCRAM dan IAM diaktifkan untuk cluster.

Saat Anda mengaktifkan skema autentikasi baru, Anda juga harus membuat koneksi VPC terkelola baru untuk skema autentikasi baru dan memperbarui klien Anda untuk menggunakan broker bootstrap khusus untuk skema autentikasi baru.

Saat Anda mematikan konektivitas pribadi multi-VPC untuk skema autentikasi, koneksi VPC yang ada di sisi klien berubah menjadi INACTIVE, dan infrastruktur Privatelink di sisi cluster, termasuk koneksi VPC terkelola, akan dihapus. Pengguna lintas akun hanya dapat menghapus koneksi VPC yang tidak aktif. Jika konektivitas pribadi diaktifkan lagi di cluster, pengguna lintas akun perlu membuat koneksi baru ke cluster.

```
Request:
{
  "currentVersion": "string",
  "connnectivityInfo": {
    "publicAccess": {
      "type": "string"
    },
    "vpcConnectivity": {
      "clientAuthentication": {
        "sasl": {
          "scram": {
            "enabled": TRUE
          },
          "iam": {
            "enabled": TRUE
          }
        },
        "tls": {
```

```
          "enabled": FALSE
        }
      }
    }
  }
}

Response:
{
  "clusterArn": "string",
  "clusterOperationArn": "string"
}
```

Tolak koneksi VPC terkelola ke kluster MSK Amazon

Dari konsol MSK Amazon di akun admin cluster, Anda dapat menolak koneksi VPC klien. Koneksi VPC klien harus dalam status TERSEDIA untuk ditolak. Anda mungkin ingin menolak koneksi VPC terkelola dari klien yang tidak lagi diizinkan untuk terhubung ke klaster Anda. Untuk mencegah koneksi VPC terkelola baru dari koneksi ke klien, tolak akses ke klien dalam kebijakan klaster. Koneksi yang ditolak masih menimbulkan biaya hingga dihapus oleh pemilik koneksi. Lihat Menghapus koneksi VPC terkelola ke kluster MSK Amazon.

Untuk menolak koneksi VPC klien menggunakan konsol MSK

1. Buka konsol MSK Amazon di AWS Management Console.
2. Di panel navigasi, pilih Cluster dan gulir ke Pengaturan jaringan > Daftar koneksi VPC klien.
3. Pilih koneksi yang ingin Anda tolak dan pilih Tolak koneksi VPC klien.
4. Konfirmasikan bahwa Anda ingin menolak koneksi VPC klien yang dipilih.

Untuk menolak koneksi VPC terkelola menggunakan API, gunakan `RejectClientVpcConnection` API.

Hapus koneksi VPC terkelola ke kluster MSK Amazon

Pengguna lintas akun dapat menghapus koneksi VPC terkelola untuk klaster MSK dari konsol akun klien. Karena pengguna pemilik klaster tidak memiliki koneksi VPC terkelola, koneksi tidak dapat dihapus dari akun admin cluster. Setelah koneksi VPC dihapus, itu tidak lagi menimbulkan biaya.

Untuk menghapus koneksi VPC yang dikelola menggunakan konsol MSK

1. Dari akun klien, buka konsol MSK Amazon di AWS Management Console.
2. Di panel navigasi, pilih Koneksi VPC terkelola.
3. Dari daftar koneksi, pilih koneksi yang ingin Anda hapus.
4. Konfirmasikan bahwa Anda ingin menghapus koneksi VPC.

Untuk menghapus koneksi VPC terkelola menggunakan API, gunakan API. `DeleteVpcConnection`

Izin untuk konektivitas pribadi multi-VPC

Bagian ini merangkum izin yang diperlukan untuk klien dan cluster menggunakan fitur konektivitas pribadi multi-VPC. Konektivitas pribadi multi-VPC mengharuskan admin klien untuk membuat izin pada setiap klien yang akan memiliki koneksi VPC terkelola ke cluster MSK. Ini juga membutuhkan admin cluster MSK untuk mengaktifkan PrivateLink konektivitas pada cluster MSK dan memilih skema otentikasi untuk mengontrol akses ke cluster.

Jenis autentikasi cluster dan izin akses topik

Aktifkan fitur konektivitas pribadi multi-VPC untuk skema autentikasi yang diaktifkan untuk kluster MSK Anda. Lihat Persyaratan dan batasan untuk konektivitas pribadi multi-VPC. Jika Anda mengonfigurasi cluster MSK Anda untuk menggunakan skema autentikasi SASL/SCRAM, properti Apache Kafka ACL adalah wajib. `allow.everyone.if.no.acl.found=false` Setelah Anda menyetel Apache Kafka ACL untuk klaster Anda, perbarui konfigurasi cluster agar properti `allow.everyone.if.no.acl.found` disetel ke false untuk cluster. Untuk informasi tentang cara memperbarui konfigurasi klaster, lihatOperasi konfigurasi MSK Amazon.

Izin kebijakan klaster lintas akun

Jika klien Kafka berada di AWS akun yang berbeda dari klaster MSK, lampirkan kebijakan berbasis cluster ke kluster MSK yang mengotorisasi pengguna root klien untuk konektivitas lintas akun. Anda dapat mengedit kebijakan klaster multi-VPC menggunakan editor kebijakan IAM di konsol MSK (Pengaturan keamanan klaster > Edit kebijakan klaster), atau menggunakan API berikut untuk mengelola kebijakan klaster:

PutClusterKebijakan

Melampirkan kebijakan klaster ke cluster. Anda dapat menggunakan API ini untuk membuat atau memperbarui kebijakan klaster MSK yang ditentukan. Jika Anda memperbarui kebijakan, bidang currentVersion diperlukan dalam payload permintaan.

GetClusterKebijakan

Mengambil teks JSON dari dokumen kebijakan cluster yang dilampirkan ke cluster.

DeleteClusterKebijakan

Menghapus kebijakan klaster.

Berikut ini adalah contoh JSON untuk kebijakan cluster dasar, mirip dengan yang ditampilkan di editor kebijakan IAM konsol MSK.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "123456789012"
                ]
            },
            "Action": [
                "kafka:CreateVpcConnection",
                "kafka:GetBootstrapBrokers",
                "kafka:DescribeCluster",
                "kafka:DescribeClusterV2"
            ],
            "Resource": "arn:aws:kafka:us-east-1:123456789012:cluster/testing/
de8982fa-8222-4e87-8b20-9bf3cdfa1521-2"
        }
    ]
}
```

Izin klien untuk konektivitas pribadi multi-VPC ke kluster MSK

Untuk mengatur konektivitas pribadi multi-VPC antara klien Kafka dan kluster MSK, klien memerlukan kebijakan identitas terlampir yang memberikan izin untuk`kafka:CreateVpcConnection`,

`ec2:CreateTags` dan tindakan pada klien. `ec2:CreateVPCEndpoint` Sebagai referensi, berikut ini adalah contoh JSON untuk kebijakan identitas klien dasar.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kafka:CreateVpcConnection",
        "ec2:CreateTags",
        "ec2:CreateVPCEndpoint"
      ],
      "Resource": "*"
    }
  ]
}
```

Informasi pelabuhan

Gunakan nomor port berikut sehingga Amazon MSK dapat berkomunikasi dengan mesin klien:

- Untuk berkomunikasi dengan broker dalam plaintext, gunakan port 9092.
- Untuk berkomunikasi dengan broker dengan enkripsi TLS, gunakan port 9094 untuk akses dari dalam AWS dan port 9194 untuk akses publik.
- Untuk berkomunikasi dengan broker dengan SASL/SCRAM, gunakan port 9096 untuk akses dari dalam AWS dan port 9196 untuk akses publik.
- Untuk berkomunikasi dengan broker dalam cluster yang diatur untuk digunakanthe section called “Kontrol akses IAM”, gunakan port 9098 untuk akses dari dalam AWS dan port 9198 untuk akses publik.
- Untuk berkomunikasi dengan Apache ZooKeeper dengan menggunakan enkripsi TLS, gunakan port 2182. Apache ZooKeeper node menggunakan port 2181 secara default.

Migrasi ke Cluster MSK Amazon

Amazon MSK Replicator dapat digunakan untuk migrasi cluster MSK. Lihat Apa itu Amazon MSK Replicator?. Atau, Anda dapat menggunakan Apache MirrorMaker 2.0 untuk bermigrasi dari klaster non-MSK ke kluster MSK Amazon. Untuk contoh cara melakukannya, lihat Memigrasi kluster Apache Kafka lokal ke Amazon MSK menggunakan. MirrorMaker Untuk informasi tentang cara menggunakan MirrorMaker, lihat Mencerminkan data antar cluster dalam dokumentasi Apache Kafka. Kami merekomendasikan pengaturan MirrorMaker dalam konfigurasi yang sangat tersedia.

Garis besar langkah-langkah yang harus diikuti saat menggunakan migrasi MirrorMaker ke kluster MSK

1. Buat klaster MSK tujuan
2. Mulai MirrorMaker dari instans Amazon EC2 dalam VPC Amazon yang sama dengan cluster tujuan.
3. Periksa MirrorMaker lag.
4. Setelah MirrorMaker mengejar ketinggalan, arahkan produsen dan konsumen ke cluster baru menggunakan broker bootstrap cluster MSK.
5. Matikan MirrorMaker.

Migrasi cluster Apache Kafka Anda ke Amazon MSK

Misalkan Anda memiliki cluster Apache Kafka bernama. `CLUSTER_ONPREM` Cluster itu diisi dengan topik dan data. Jika Anda ingin memigrasikan klaster tersebut ke kluster MSK Amazon yang baru dibuat bernama`CLUSTER_AWSMSK`, prosedur ini memberikan tampilan tingkat tinggi dari langkah-langkah yang perlu Anda ikuti.

Untuk memigrasikan cluster Apache Kafka yang ada ke Amazon MSK

1. Di`CLUSTER_AWSMSK`, buat semua topik yang ingin Anda migrasikan.

   Anda tidak dapat menggunakan MirrorMaker untuk langkah ini karena tidak secara otomatis membuat ulang topik yang ingin Anda migrasi dengan tingkat replikasi yang tepat. Anda dapat membuat topik di Amazon MSK dengan faktor replikasi yang sama dan jumlah partisi yang mereka miliki. `CLUSTER_ONPREM` Anda juga dapat membuat topik dengan berbagai faktor replikasi dan jumlah partisi.

2. Mulai MirrorMaker dari instance yang memiliki akses baca `CLUSTER_ONPREM` dan akses tulis ke`CLUSTER_AWSMSK`.
3. Jalankan perintah berikut untuk mencerminkan semua topik:

```
<path-to-your-kafka-installation>/bin/kafka-mirror-maker.sh --consumer.config
 config/mirrormaker-consumer.properties --producer.config config/mirrormaker-
producer.properties --whitelist '.*'
```

   Dalam perintah ini, `config/mirrormaker-consumer.properties` arahkan ke broker bootstrap di`CLUSTER_ONPREM`; misalnya,`bootstrap.servers=localhost:9092`.
   Dan `config/mirrormaker-producer.properties` menunjuk ke broker bootstrap di CLUSTER_AWSMSK; misalnya,.
   `bootstrap.servers=10.0.0.237:9092,10.0.2.196:9092,10.0.1.233:9092`
4. Terus MirrorMaker berjalan di latar belakang, dan terus gunakan`CLUSTER_ONPREM`. MirrorMaker mencerminkan semua data baru.
5. Periksa kemajuan pencerminan dengan memeriksa jeda antara offset terakhir untuk setiap topik dan offset saat ini yang dikonsumsi. MirrorMaker

   Ingat MirrorMaker itu hanya menggunakan konsumen dan produsen. Jadi, Anda dapat memeriksa lag menggunakan `kafka-consumer-groups.sh` alat ini. Untuk menemukan nama grup konsumen, lihat di dalam `mirrormaker-consumer.properties` file untuk`group.id`, dan gunakan nilainya. Jika tidak ada kunci seperti itu dalam file, Anda dapat membuatnya. Misalnya, atur`group.id=mirrormaker-consumer-group`.
6. Setelah MirrorMaker selesai mencerminkan semua topik, hentikan semua produsen dan konsumen, lalu berhenti. MirrorMaker Kemudian arahkan produsen dan konsumen ke `CLUSTER_AWSMSK` cluster dengan mengubah nilai produsen dan broker bootstrap konsumen mereka. Mulai ulang semua produsen dan konsumen`CLUSTER_AWSMSK`.

Migrasi dari satu kluster MSK Amazon ke cluster MSK lainnya

Anda dapat menggunakan Apache MirrorMaker 2.0 untuk bermigrasi dari cluster non-MSK ke kluster MSK. Misalnya, Anda dapat bermigrasi dari satu versi Apache Kafka ke versi lain. Untuk contoh cara melakukannya, lihat Memigrasi kluster Apache Kafka lokal ke Amazon MSK menggunakan. MirrorMaker Atau, Amazon MSK Replicator dapat digunakan untuk migrasi cluster MSK. Untuk informasi selengkapnya tentang Amazon MSK Replicator, lihat. Replikator MSK

MirrorMaker 1.0 praktik terbaik

Daftar praktik terbaik ini berlaku untuk MirrorMaker 1.0.

- Jalankan MirrorMaker di cluster tujuan. Dengan cara ini, jika masalah jaringan terjadi, pesan masih tersedia di cluster sumber. Jika Anda menjalankan MirrorMaker di cluster sumber dan peristiwa di-buffer di produser dan ada masalah jaringan, peristiwa mungkin hilang.
- Jika enkripsi diperlukan dalam perjalanan, jalankan di cluster sumber.
- Untuk konsumen, setel auto.commit.enabled=false
- Untuk produsen, tetapkan
  - max.in.flight.requests.per.connection=1
  - retries=int.max_value
  - acks=semua
  - max.block.ms = Long.Max_Value
- Untuk throughput produsen yang tinggi:
  - Menyangga pesan dan mengisi kumpulan pesan - tune buffer.memory, batch.size, linger.ms
  - Tune buffer soket - menerima.buffer.bytes, send.buffer.bytes
- Untuk menghindari kehilangan data, matikan komit otomatis di sumbernya, sehingga MirrorMaker dapat mengontrol komit, yang biasanya dilakukan setelah menerima ack dari cluster tujuan. Jika produsen memiliki acks=all dan cluster tujuan memiliki min.insync.replicas yang disetel ke lebih dari 1, pesan disimpan di lebih dari satu broker di tujuan sebelum konsumen melakukan offset di sumbernya. MirrorMaker
- Jika pesanan penting, Anda dapat mengatur percobaan ulang ke 0. Atau, untuk lingkungan produksi, setel koneksi dalam pesawat maksimal ke 1 untuk memastikan bahwa batch yang dikirim tidak dilakukan rusak jika batch gagal di tengah. Dengan cara ini, setiap batch yang dikirim akan dicoba lagi sampai batch berikutnya dikirim. Jika max.block.ms tidak disetel ke nilai maksimum, dan jika buffer produsen penuh, mungkin ada kehilangan data (tergantung pada beberapa pengaturan lainnya). Ini dapat memblokir dan menekan kembali konsumen.
- Untuk throughput tinggi
  - Meningkatkan buffer.memory.
  - Tingkatkan ukuran batch.
  - Tune linger.ms untuk memungkinkan batch terisi. Ini juga memungkinkan kompresi yang lebih baik, penggunaan bandwidth jaringan yang lebih sedikit, dan lebih sedikit penyimpanan di cluster. Ini menghasilkan peningkatan retensi.

    - Pantau penggunaan CPU dan memori.
- Untuk throughput konsumen yang tinggi
    - Tingkatkan jumlah utas/konsumen per MirrorMaker proses - num.streams.
    - Tingkatkan jumlah MirrorMaker proses di seluruh mesin terlebih dahulu sebelum meningkatkan thread untuk memungkinkan ketersediaan tinggi.
    - Tingkatkan jumlah MirrorMaker proses pertama pada mesin yang sama dan kemudian pada mesin yang berbeda (dengan ID grup yang sama).
    - Isolasi topik yang memiliki throughput sangat tinggi dan gunakan instance terpisah MirrorMaker .
- Untuk manajemen dan konfigurasi
    - Gunakan AWS CloudFormation dan alat manajemen konfigurasi seperti Chef dan Ansible.
    - Gunakan mount Amazon EFS agar semua file konfigurasi dapat diakses dari semua instans Amazon EC2.
    - Gunakan wadah untuk memudahkan penskalaan dan pengelolaan MirrorMaker instance.
- Biasanya, dibutuhkan lebih dari satu konsumen untuk memenuhi produsen. MirrorMaker Jadi, siapkan banyak konsumen. Pertama, atur pada mesin yang berbeda untuk memberikan ketersediaan tinggi. Kemudian, skala mesin individu hingga memiliki konsumen untuk setiap partisi, dengan konsumen didistribusikan secara merata di seluruh mesin.
- Untuk konsumsi dan pengiriman throughput yang tinggi, setel buffer terima dan kirim karena defaultnya mungkin terlalu rendah. Untuk performa maksimal, pastikan jumlah total stream (num.streams) cocok dengan semua partisi topik yang MirrorMaker mencoba menyalin ke cluster tujuan.

MirrorMaker 2.* keuntungan

- Memanfaatkan kerangka kerja dan ekosistem Apache Kafka Connect.
- Mendeteksi topik dan partisi baru.
- Secara otomatis menyinkronkan konfigurasi topik antar cluster.
- Mendukung pasangan cluster “aktif/aktif”, serta sejumlah cluster aktif.
- Menyediakan metrik baru termasuk latensi end-to-end replikasi di beberapa pusat data dan cluster.
- Memancarkan offset yang diperlukan untuk memigrasikan konsumen antar cluster dan menyediakan perkakas untuk terjemahan offset.

- Mendukung file konfigurasi tingkat tinggi untuk menentukan beberapa cluster dan aliran replikasi di satu tempat, dibandingkan dengan properti produsen/konsumen tingkat rendah untuk setiap proses 1.*. MirrorMaker

Memantau kluster MSK Amazon

Ada beberapa cara Amazon MSK membantu Anda memantau status cluster MSK Amazon Anda.

- Amazon MSK membantu Anda memantau kapasitas penyimpanan disk Anda dengan secara otomatis mengirimkan peringatan kapasitas penyimpanan saat klaster akan mencapai batas kapasitas penyimpanannya. Peringatan juga memberikan rekomendasi tentang langkah-langkah terbaik yang harus diambil untuk mengatasi masalah yang terdeteksi. Ini membantu Anda mengidentifikasi dan menyelesaikan masalah kapasitas disk dengan cepat sebelum menjadi kritis. Amazon MSK secara otomatis mengirimkan peringatan ini ke konsol MSK Amazon, AWS Health Dashboard Amazon EventBridge, dan kontak email untuk akun Anda. AWS Untuk informasi tentang peringatan kapasitas penyimpanan, lihatPeringatan kapasitas penyimpanan MSK Amazon.
- Amazon MSK mengumpulkan metrik Apache Kafka dan mengirimkannya ke Amazon CloudWatch di mana Anda dapat melihatnya. Untuk informasi selengkapnya tentang metrik Apache Kafka, termasuk metrik yang ditampilkan MSK Amazon, lihat Pemantauan dalam dokumentasi Apache Kafka.
- Anda juga dapat memantau kluster MSK Anda dengan Prometheus, aplikasi pemantauan sumber terbuka. Untuk informasi tentang Prometheus, lihat Ikhtisar di dokumentasi Prometheus. Untuk mempelajari cara memantau cluster Anda dengan Prometheus, lihat. the section called “Pemantauan terbuka dengan Prometheus”

Topik

- Metrik MSK Amazon untuk pemantauan dengan CloudWatch
- Melihat metrik MSK Amazon menggunakan CloudWatch
- Pemantauan kelambatan konsumen
- Pemantauan terbuka dengan Prometheus
- Peringatan kapasitas penyimpanan MSK Amazon

Metrik MSK Amazon untuk pemantauan dengan CloudWatch

Amazon MSK terintegrasi dengan Amazon CloudWatch sehingga Anda dapat mengumpulkan, melihat, dan menganalisis CloudWatch metrik untuk kluster MSK Amazon Anda. Metrik yang Anda konfigurasikan untuk klaster MSK Anda secara otomatis dikumpulkan dan didorong ke. CloudWatch Anda dapat mengatur tingkat pemantauan untuk klaster MSK ke salah satu dari berikut ini:`DEFAULT`,,

PER_BROKERPER_TOPIC_PER_BROKER, atauPER_TOPIC_PER_PARTITION. Tabel di bagian berikut menunjukkan semua metrik yang tersedia mulai dari setiap tingkat pemantauan.

> **Note**
>
> Nama-nama beberapa metrik MSK Amazon untuk CloudWatch pemantauan telah berubah di versi 3.6.0 dan lebih tinggi. Gunakan nama baru untuk memantau metrik ini. Untuk metrik dengan nama yang diubah, tabel di bawah ini menunjukkan nama yang digunakan dalam versi 3.6.0 dan yang lebih tinggi, diikuti dengan nama di versi 2.8.2.tiered.

DEFAULTMetrik -level gratis. Harga untuk metrik lainnya dijelaskan di halaman CloudWatchharga Amazon.

DEFAULTPemantauan tingkat

Metrik yang dijelaskan dalam tabel berikut tersedia di tingkat DEFAULT pemantauan. Mereka bebas.

Metrik tersedia di tingkat **DEFAULT** pemantauan

Nama	Saat terlihat	Dimensi	Deskripsi
ActiveController Count	Setelah cluster sampai ke status ACTIVE.	Nama Klaster	Hanya satu pengontrol per cluster yang harus aktif pada waktu tertentu.
BurstBalance	Setelah cluster sampai ke status ACTIVE.	Nama Cluster, ID Pialang	Saldo sisa kredit burst input-out put untuk volume EBS di cluster. Gunakan untuk menyelidiki latensi atau penurunan throughput. BurstBalance tidak dilaporkan untuk volume EBS ketika kinerja dasar volume lebih tinggi dari kinerja burst maksimum. Untuk informasi selengkapnya, lihat Kredit I/O dan kinerja burst.
BytesInPerSec	Setelah Anda membuat topik.	Nama Cluster, ID	Jumlah byte per detik yang diterima dari klien. Metrik ini tersedia per broker dan juga per topik.

Nama	Saat terlihat	Dimensi	Deskripsi
		Pialang, Topik	
BytesOutPerSec	Setelah Anda membuat topik.	Nama Cluster, ID Pialang, Topik	Jumlah byte per detik dikirim ke klien. Metrik ini tersedia per broker dan juga per topik.
ClientConnectionCount	Setelah cluster sampai ke status ACTIVE.	Nama Cluster, ID Broker, Otentikasi Klien	Jumlah koneksi klien yang diautentikasi aktif.
ConnectionCount	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Jumlah koneksi aktif yang diautentikasi, tidak diautentikasi, dan antar-broker.
CPUCreditBalance	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Jumlah kredit CPU yang diperoleh yang diperoleh broker sejak diluncurkan. Kredit diakumulasi ke saldo kredit setelah diperoleh, dan dihapus dari saldo kredit saat digunakan. Jika Anda kehabisan saldo kredit CPU, itu dapat berdampak negatif pada kinerja cluster Anda. Anda dapat mengambil langkah-langkah untuk mengurangi beban CPU. Misalnya, Anda dapat mengurangi jumlah permintaan klien atau memperbarui jenis broker ke jenis broker M5.

Nama	Saat terlihat	Dimensi	Deskripsi
`CpuIdle`	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Persentase waktu idle CPU.
`CpuIoWait`	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Persentase waktu idle CPU selama operasi disk yang tertunda.
`CpuSystem`	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Persentase CPU di ruang kernel.
`CpuUser`	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Persentase CPU di ruang pengguna.
`GlobalPartitionCount`	Setelah cluster sampai ke status ACTIVE.	Nama Klaster	Jumlah partisi di semua topik di cluster, tidak termasuk replika. Karena `GlobalPartitionCount` tidak termasuk replika, jumlah `PartitionCount` nilai bisa lebih tinggi daripada GlobalPartitionCount jika faktor replikasi untuk suatu topik lebih besar dari 1.
`GlobalTopicCount`	Setelah cluster sampai ke status ACTIVE.	Nama Klaster	Jumlah total topik di semua broker di cluster.
`EstimatedMaxTimeLag`	Setelah kelompok konsumen mengkonsumsi dari suatu topik.	Grup Konsum Topik	Perkiraan waktu (dalam detik) untuk mengurasMaxOffsetLag .

Nama	Saat terlihat	Dimensi	Deskripsi
KafkaAppLogsDiskUsed	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Persentase ruang disk yang digunakan untuk log aplikasi.
KafkaDataLogsDiskUsed (Cluster Name, Broker IDdimensi)	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Persentase ruang disk yang digunakan untuk log data.
KafkaDataLogsDiskUsed (Cluster Namedimensi)	Setelah cluster sampai ke status ACTIVE.	Nama Klaster	Persentase ruang disk yang digunakan untuk log data.
LeaderCount	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Jumlah total pemimpin partisi per broker, tidak termasuk replika.
MaxOffsetLag	Setelah kelompok konsumen mengkonsumsi dari suatu topik.	Grup Konsum Topik	Keterlambatan offset maksimum di semua partisi dalam suatu topik.
MemoryBuffered	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Ukuran dalam byte memori buffer untuk broker.
MemoryCached	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Ukuran dalam byte memori cache untuk broker.

Nama	Saat terlihat	Dimensi	Deskripsi
MemoryFree	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Ukuran dalam byte memori yang gratis dan tersedia untuk broker.
HeapMemoryAfterGC	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Persentase total memori heap yang digunakan setelah pengumpulan sampah.
MemoryUsed	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Ukuran dalam byte memori yang digunakan untuk broker.
MessagesInPerSec	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Jumlah pesan masuk per detik untuk broker.
NetworkRxDropped	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Jumlah paket penerima yang dijatuhkan.
NetworkRxErrors	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Jumlah jaringan menerima kesalahan untuk broker.
NetworkRxPackets	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Jumlah paket yang diterima oleh broker.

Nama	Saat terlihat	Dimensi	Deskripsi
NetworkTx Dropped	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Jumlah paket pengiriman yang dijatuhkan.
NetworkTx Errors	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Jumlah kesalahan transmisi jaringan untuk broker.
NetworkTx Packets	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Jumlah paket yang dikirimkan oleh broker.
OfflinePa rtitionsCount	Setelah cluster sampai ke status ACTIVE.	Nama Klaster	Jumlah total partisi yang offline di cluster.
PartitionCount	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Jumlah total partisi topik per broker, termasuk replika.
ProduceTo talTimeMsMean	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Rata-rata menghasilkan waktu dalam milidetik.
RequestBy tesMean	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Jumlah rata-rata byte permintaan untuk broker.

Nama	Saat terlihat	Dimensi	Deskripsi
RequestTime	Setelah permintaan throttling diterapkan.	Nama Klaster, ID Broker	Rata-rata waktu dalam milidetik yang dihabiskan di jaringan broker dan utas I/O untuk memproses permintaan.
RootDiskUsed	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Persentase disk root yang digunakan oleh broker.
SumOffsetLag	Setelah kelompok konsumen mengkonsumsi dari suatu topik.	Grup Konsume Topik	Kelambatan offset agregat untuk semua partisi dalam suatu topik.
SwapFree	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Ukuran dalam byte memori swap yang tersedia untuk broker.
SwapUsed	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Ukuran dalam byte memori swap yang digunakan untuk broker.
TrafficShaping	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Metrik tingkat tinggi menunjukkan jumlah paket yang dibentuk (jatuh atau antri) karena melebihi alokasi jaringan. Detail yang lebih halus tersedia dengan metrik PER_BROKER.
UnderMinIsrPartitionCount	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Jumlah partisi miniSR di bawah untuk broker.

Nama	Saat terlihat	Dimensi	Deskripsi
`UnderReplicatedPartitions`	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Jumlah partisi yang kurang direplikasi untuk broker.
`ZooKeeperRequestLatencyMsMean`	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Untuk cluster ZooKeeper berbasis. Latensi rata-rata dalam milidetik untuk ZooKeeper permintaan Apache dari broker.
`ZooKeeperSessionState`	Setelah cluster sampai ke status ACTIVE.	Nama Klaster, ID Broker	Untuk cluster ZooKeeper berbasis. Status koneksi ZooKeeper sesi broker yang mungkin salah satu dari yang berikut: NOT_CONNECTED: '0.0', ASSOCIATING: '0.1', CONNECTING: '0.5', CONNECTEDREADONLY: '0.8', CONNECTED: '1.0', CLOSED: '5.0', AUTH_FAILED: '10.0'.

PER_BROKER Pemantauan tingkat

Saat Anda mengatur level pemantauan`PER_BROKER`, Anda mendapatkan metrik yang dijelaskan dalam tabel berikut selain semua metrik `DEFAULT` level. Anda membayar metrik dalam tabel berikut, sedangkan metrik `DEFAULT` level tetap gratis. Metrik dalam tabel ini memiliki dimensi sebagai berikut: Nama Cluster, ID Broker.

Metrik tambahan yang tersedia mulai dari tingkat **`PER_BROKER`** pemantauan

Nama	Saat terlihat	Deskripsi
`BwInAllowanceExceeded`	Setelah cluster sampai ke status ACTIVE.	Jumlah paket yang dibentuk karena bandwidth agregat inbound melebihi maksimum untuk broker.

Nama	Saat terlihat	Deskripsi
BwOutAllowanceExceeded	Setelah cluster sampai ke status ACTIVE.	Jumlah paket yang dibentuk karena bandwidth agregat outbound melebihi maksimum untuk broker.
ConnTrackAllowanceExceeded	Setelah cluster sampai ke status ACTIVE.	Jumlah paket yang dibentuk karena koneksi tracking melebihi maksimum untuk broker. Pelacakan koneksi terkait dengan grup keamanan yang melacak setiap koneksi yang dibuat untuk memastikan bahwa paket pengembalian dikirim seperti yang diharapkan.
ConnectionCloseRate	Setelah cluster sampai ke status ACTIVE.	Jumlah koneksi ditutup per detik per pendengar. Jumlah ini dikumpulkan per pendengar dan disaring untuk pendengar klien.
ConnectionCreationRate	Setelah cluster sampai ke status ACTIVE.	Jumlah koneksi baru yang dibuat per detik per pendengar. Jumlah ini dikumpulkan per pendengar dan disaring untuk pendengar klien.
CpuCreditUsage	Setelah cluster sampai ke status ACTIVE.	Jumlah kredit CPU yang dihabiskan oleh broker. Jika Anda kehabisan saldo kredit CPU, itu dapat berdampak negatif pada kinerja cluter Anda. Anda dapat mengambil langkah-langkah untuk mengurangi beban CPU. Misalnya, Anda dapat mengurangi jumlah permintaan klien atau memperbarui jenis broker ke jenis broker M5.

Nama	Saat terlihat	Deskripsi
FetchConsumerLocalTimeMsMean	Setelah ada produsen/konsumen.	Waktu rata-rata dalam milidetik permintaan konsumen diproses di pemimpin.
FetchConsumerRequestQueueTimeMsMean	Setelah ada produsen/konsumen.	Waktu rata-rata dalam milidetik permintaan konsumen menunggu dalam antrian permintaan.
FetchConsumerResponseQueueTimeMsMean	Setelah ada produsen/konsumen.	Waktu rata-rata dalam milidetik permintaan konsumen menunggu dalam antrian respons.
FetchConsumerResponseSendTimeMsMean	Setelah ada produsen/konsumen.	Waktu rata-rata dalam milidetik bagi konsumen untuk mengirim respons.
FetchConsumerTotalTimeMsMean	Setelah ada produsen/konsumen.	Total waktu rata-rata dalam milidetik yang dihabiskan konsumen untuk mengambil data dari broker.
FetchFollowerLocalTimeMsMean	Setelah ada produsen/konsumen.	Waktu rata-rata dalam milidetik permintaan pengikut diproses di pemimpin.
FetchFollowerRequestQueueTimeMsMean	Setelah ada produsen/konsumen.	Waktu rata-rata dalam milidetik permintaan pengikut menunggu dalam antrian permintaan.
FetchFollowerResponseQueueTimeMsMean	Setelah ada produsen/konsumen.	Waktu rata-rata dalam milidetik permintaan pengikut menunggu dalam antrian respons.
FetchFollowerResponseSendTimeMsMean	Setelah ada produsen/konsumen.	Waktu rata-rata dalam milidetik bagi pengikut untuk mengirim respons.
FetchFollowerTotalTimeMsMean	Setelah ada produsen/konsumen.	Total waktu rata-rata dalam milidetik yang dihabiskan pengikut untuk mengambil data dari broker.

Nama	Saat terlihat	Deskripsi
FetchMessageConversionsPerSec	Setelah Anda membuat topik.	Jumlah konversi pesan ambil per detik untuk broker.
FetchThrottleByteRate	Setelah bandwidth throttling diterapkan.	Jumlah byte yang dibatasi per detik.
FetchThrottleQueueSize	Setelah bandwidth throttling diterapkan.	Jumlah pesan dalam antrian throttle.
FetchThrottleTime	Setelah bandwidth throttling diterapkan.	Rata-rata waktu fetch throttle dalam milidetik.
IAMNumberOfConnectionRequests	Setelah cluster sampai ke status ACTIVE.	Jumlah permintaan otentikasi IAM per detik.
IAMTooManyConnections	Setelah cluster sampai ke status ACTIVE.	Jumlah koneksi yang dicoba melebihi 100. 0 berarti jumlah koneksi berada dalam batas. Jika >0, batas throttle terlampaui dan Anda perlu mengurangi jumlah koneksi.
NetworkProcessorAvgIdlePercent	Setelah cluster sampai ke status ACTIVE.	Persentase rata-rata waktu prosesor jaringan menganggur.
PpsAllowanceExceeded	Setelah cluster sampai ke status ACTIVE.	Jumlah paket yang dibentuk karena PPS dua arah melebihi maksimum untuk broker.
ProduceLocalTimeMsMean	Setelah cluster sampai ke status ACTIVE.	Waktu rata-rata dalam milidetik bahwa permintaan diproses di pemimpin.
ProduceMessageConversionsPerSec	Setelah Anda membuat topik.	Jumlah konversi pesan produksi per detik untuk broker.

Nama	Saat terlihat	Deskripsi
ProduceMessageConversionsTimeMsMean	Setelah cluster sampai ke status ACTIVE.	Waktu rata-rata dalam milidetik dihabiskan untuk konversi format pesan.
ProduceRequestQueueTimeMsMean	Setelah cluster sampai ke status ACTIVE.	Waktu rata-rata dalam milidetik yang digunakan pesan permintaan dalam antrian.
ProduceResponseQueueTimeMsMean	Setelah cluster sampai ke status ACTIVE.	Waktu rata-rata dalam milidetik yang dihabiskan pesan respons dalam antrian.
ProduceResponseSendTimeMsMean	Setelah cluster sampai ke status ACTIVE.	Waktu rata-rata dalam milidetik dihabiskan untuk mengirim pesan respons.
ProduceThrottleByteRate	Setelah bandwidth throttling diterapkan.	Jumlah byte yang dibatasi per detik.
ProduceThrottleQueueSize	Setelah bandwidth throttling diterapkan.	Jumlah pesan dalam antrian throttle.
ProduceThrottleTime	Setelah bandwidth throttling diterapkan.	Rata-rata menghasilkan waktu throttle dalam milidetik.
ProduceTotalTimeMsMean	Setelah cluster sampai ke status ACTIVE.	Rata-rata menghasilkan waktu dalam milidetik.

Nama	Saat terlihat	Deskripsi
`RemoteFetchBytesPerSec (RemoteBytesInPerSec in v2.8.2.tiered)`	Setelah ada produsen/konsumen.	Jumlah total byte yang ditransfer dari penyimpanan berjenjang sebagai respons terhadap pengambilan konsumen. Metrik ini mencakup semua partisi topik yang berkontribusi pada lalu lintas transfer data hilir. Kategori: Lalu lintas dan tingkat kesalahan. Ini adalah metrik KIP-405.
`RemoteCopyBytesPerSec (RemoteBytesOutPerSec in v2.8.2.tiered)`	Setelah ada produsen/konsumen.	Jumlah total byte yang ditransfer ke penyimpanan berjenjang, termasuk data dari segmen log, indeks, dan file tambahan lainnya. Metrik ini mencakup semua partisi topik yang berkontribusi pada lalu lintas transfer data hulu. Kategori: Lalu lintas dan tingkat kesalahan. Ini adalah metrik KIP-405.
`RemoteLogManagerTasksAvgIdlePercent`	Setelah cluster sampai ke status ACTIVE.	Persentase rata-rata waktu yang dihabiskan manajer log jarak jauh untuk menganggur. Manajer log jarak jauh mentransfer data dari broker ke penyimpanan berjenjang. Kategori: Aktivitas internal. Ini adalah metrik KIP-405.
`RemoteLogReaderAvgIdlePercent`	Setelah cluster sampai ke status ACTIVE.	Persentase rata-rata waktu yang dihabiskan pembaca log jarak jauh untuk menganggur. Pembaca log jarak jauh mentransfer data dari penyimpanan jarak jauh ke broker sebagai respons terhadap pengambilan konsumen. Kategori: Aktivitas internal. Ini adalah metrik KIP-405.

Nama	Saat terlihat	Deskripsi
`RemoteLogReaderTaskQueueSize`	Setelah cluster sampai ke status ACTIVE.	Jumlah tugas yang bertanggung jawab untuk membaca dari penyimpanan berjenjang yang menunggu untuk dijadwalkan. Kategori: Aktivitas internal. Ini adalah metrik KIP-405.
`RemoteFetchErrorsPerSec (RemoteReadErrorPerSec in v2.8.2.tiered)`	Setelah cluster sampai ke status ACTIVE.	Tingkat total kesalahan dalam menanggapi permintaan baca yang dikirim broker tertentu ke penyimpanan berjenjang untuk mengambil data sebagai respons terhadap pengambilan konsumen. Metrik ini mencakup semua partisi topik yang berkontribusi pada lalu lintas transfer data hilir. Kategori: lalu lintas dan tingkat kesalahan. Ini adalah metrik KIP-405.
`RemoteFetchRequestsPerSec (RemoteReadRequestsPerSec in v2.8.2.tiered)`	Setelah cluster sampai ke status ACTIVE.	Jumlah total permintaan baca yang dikirimkan oleh broker ke penyimpanan berjenjang untuk mengambil data sebagai tanggapan terhadap pengambilan konsumen. Metrik ini mencakup semua partisi topik yang berkontribusi pada lalu lintas transfer data hilir. Kategori: lalu lintas dan tingkat kesalahan. Ini adalah metrik KIP-405.

Nama	Saat terlihat	Deskripsi
`RemoteCopyErrorsPerSec (RemoteWriteErrorPerSec in v2.8.2.tiered)`	Setelah cluster sampai ke status ACTIVE.	Tingkat total kesalahan dalam menanggapi permintaan penulisan yang dikirim oleh broker tertentu ke penyimpanan berjenjang untuk mentransfer data ke hulu. Metrik ini mencakup semua partisi topik yang berkontribusi pada lalu lintas transfer data hulu. Kategori: lalu lintas dan tingkat kesalahan. Ini adalah metrik KIP-405.
`ReplicationBytesInPerSec`	Setelah Anda membuat topik.	Jumlah byte per detik yang diterima dari broker lain.
`ReplicationBytesOutPerSec`	Setelah Anda membuat topik.	Jumlah byte per detik dikirim ke broker lain.
`RequestExemptFromThrottleTime`	Setelah permintaan throttling diterapkan.	Rata-rata waktu dalam milidetik yang dihabiskan di jaringan broker dan utas I/O untuk memproses permintaan yang dibebaskan dari pembatasan.
`RequestHandlerAvgIdlePercent`	Setelah cluster sampai ke status ACTIVE.	Persentase rata-rata waktu thread handler permintaan tidak digunakan.
`RequestThrottleQueueSize`	Setelah permintaan throttling diterapkan.	Jumlah pesan dalam antrian throttle.
`RequestThrottleTime`	Setelah permintaan throttling diterapkan.	Rata-rata waktu permintaan throttle dalam milidetik.
`TcpConnections`	Setelah cluster sampai ke status ACTIVE.	Menampilkan jumlah segmen TCP masuk dan keluar dengan set bendera SYN.

Nama	Saat terlihat	Deskripsi
`RemoteCopyLagBytes (TotalTierBytesLag in v2.8.2.tiered)`	Setelah Anda membuat topik.	Jumlah total byte data yang memenuhi syarat untuk tiering pada broker tetapi belum ditransfer ke penyimpanan berjenjang. Metrik ini menunjukkan efisiensi transfer data hulu. Ketika lag meningkat, jumlah data yang tidak bertahan dalam penyimpanan berjenjang meningkat. Kategori: Arsip lag. Ini bukan metrik KIP-405.
`TrafficBytes`	Setelah cluster sampai ke status ACTIVE.	Menunjukkan lalu lintas jaringan dalam byte keseluruhan antara klien (produsen dan konsumen) dan broker. Lalu lintas antar broker tidak dilaporkan.
`VolumeQueueLength`	Setelah cluster sampai ke status ACTIVE.	Jumlah permintaan operasi baca dan tulis yang menunggu untuk diselesaikan dalam jangka waktu tertentu.
`VolumeReadBytes`	Setelah cluster sampai ke status ACTIVE.	Jumlah byte yang dibaca dalam periode waktu tertentu.
`VolumeReadOps`	Setelah cluster sampai ke status ACTIVE.	Jumlah operasi baca dalam periode waktu tertentu.
`VolumeTotalReadTime`	Setelah cluster sampai ke status ACTIVE.	Jumlah total detik yang dihabiskan oleh semua operasi baca yang diselesaikan dalam periode waktu tertentu.

Nama	Saat terlihat	Deskripsi
`VolumeTotalWriteTime`	Setelah cluster sampai ke status ACTIVE.	Jumlah total detik yang dihabiskan oleh semua operasi penulisan yang diselesaikan dalam periode waktu tertentu.
`VolumeWriteBytes`	Setelah cluster sampai ke status ACTIVE.	Jumlah byte yang ditulis dalam periode waktu tertentu.
`VolumeWriteOps`	Setelah cluster sampai ke status ACTIVE.	Jumlah operasi tulis dalam periode waktu tertentu.

`PER_TOPIC_PER_BROKER`Pemantauan tingkat

Saat Anda mengatur tingkat pemantauan`PER_TOPIC_PER_BROKER`, Anda mendapatkan metrik yang dijelaskan dalam tabel berikut, selain semua metrik dari level `PER_BROKER` dan DEFAULT. Hanya metrik `DEFAULT` level yang gratis. Metrik dalam tabel ini memiliki dimensi sebagai berikut: Nama Cluster, ID Broker, Topik.

> **Important**
>
> Untuk kluster MSK Amazon yang menggunakan Apache Kafka 2.4.1 atau versi yang lebih baru, metrik dalam tabel berikut hanya muncul setelah nilainya menjadi bukan nol untuk pertama kalinya. Misalnya, untuk melihat`BytesInPerSec`, satu atau lebih produsen harus terlebih dahulu mengirim data ke cluster.

Metrik tambahan yang tersedia mulai dari tingkat **`PER_TOPIC_PER_BROKER`** pemantauan

Nama	Saat terlihat	Deskripsi
`FetchMessageConversionsPerSec`	Setelah Anda membuat topik.	Jumlah pesan yang diambil dikonversi per detik.

Nama	Saat terlihat	Deskripsi
`MessagesInPerSec`	Setelah Anda membuat topik.	Jumlah pesan yang diterima per detik.
`ProduceMessageConversionsPerSec`	Setelah Anda membuat topik.	Jumlah konversi per detik untuk pesan yang dihasilkan.
`RemoteFetchBytesPerSec (RemoteBytesInPerSec in v2.8.2.tiered)`	Setelah Anda membuat topik dan topik tersebut menghasilkan/memakan.	Jumlah byte yang ditransfer dari penyimpanan berjenjang sebagai respons terhadap pengambilan konsumen untuk topik dan broker yang ditentukan. Metrik ini mencakup semua partisi dari topik yang berkontribusi pada lalu lintas transfer data hilir pada broker yang ditentukan. Kategori: lalu lintas dan tingkat kesalahan. Ini adalah metrik KIP-405.
`RemoteCopyBytesPerSec (RemoteBytesOutPerSec in v2.8.2.tiered)`	Setelah Anda membuat topik dan topik tersebut menghasilkan/memakan.	Jumlah byte yang ditransfer ke penyimpanan berjenjang, untuk topik dan broker yang ditentukan. Metrik ini mencakup semua partisi dari topik yang berkontribusi pada lalu lintas transfer data hulu pada broker yang ditentukan. Kategori: lalu lintas dan tingkat kesalahan. Ini adalah metrik KIP-405.
`RemoteFetchErrorsPerSec (RemoteReadErrorPerSec in v2.8.2.tiered)`	Setelah Anda membuat topik dan topik tersebut menghasilkan/memakan.	Tingkat kesalahan dalam menanggapi permintaan baca yang dikirim broker tertentu ke penyimpanan berjenjang untuk mengambil data sebagai tanggapan terhadap pengambilan konsumen pada topik yang ditentukan. Metrik ini mencakup semua partisi dari topik yang berkontribusi pada lalu lintas transfer data hilir pada broker yang ditentukan. Kategori: lalu lintas dan tingkat kesalahan. Ini adalah metrik KIP-405.

Nama	Saat terlihat	Deskripsi
`RemoteFetchRequestsPerSec (RemoteReadRequestsPerSec in v2.8.2.tiered)`	Setelah Anda membuat topik dan topik tersebut menghasilkan/memakan.	Jumlah permintaan baca yang dikirimkan oleh broker tertentu ke penyimpanan berjenjang untuk mengambil data sebagai tanggapan terhadap pengambilan konsumen pada topik yang ditentukan. Metrik ini mencakup semua partisi dari topik yang berkontribusi pada lalu lintas transfer data hilir pada broker yang ditentukan. Kategori: lalu lintas dan tingkat kesalahan. Ini adalah metrik KIP-405.
`RemoteCopyErrorsPerSec (RemoteWriteErrorPerSec in v2.8.2.tiered)`	Setelah Anda membuat topik dan topik tersebut menghasilkan/memakan.	Tingkat kesalahan dalam menanggapi permintaan penulisan yang dikirim broker tertentu ke penyimpanan berjenjang untuk mentransfer data ke hulu. Metrik ini mencakup semua partisi dari topik yang berkontribusi pada lalu lintas transfer data hulu pada broker yang ditentukan. Kategori: lalu lintas dan tingkat kesalahan. Ini adalah metrik KIP-405.

`PER_TOPIC_PER_PARTITION`Pemantauan tingkat

Saat Anda mengatur tingkat pemantauan`PER_TOPIC_PER_PARTITION`, Anda mendapatkan metrik yang dijelaskan dalam tabel berikut, selain semua metrik dari level`PER_TOPIC_PER_BROKER`,`PER_BROKER`, dan DEFAULT. Hanya metrik `DEFAULT` level yang gratis. Metrik dalam tabel ini memiliki dimensi sebagai berikut: Grup Konsumen, Topik, Partisi.

Metrik tambahan yang tersedia mulai dari tingkat **`PER_TOPIC_PER_PARTITION`** pemantauan

Nama	Saat terlihat	Deskripsi
`EstimatedTimeLag`	Setelah kelompok konsumen mengkonsumsi dari suatu topik.	Perkiraan waktu (dalam detik) untuk menguras lag offset partisi.

Nama	Saat terlihat	Deskripsi
`OffsetLag`	Setelah kelompok konsumen mengkonsumsi dari suatu topik.	Kelambatan konsumen tingkat partisi dalam jumlah offset.

Melihat metrik MSK Amazon menggunakan CloudWatch

Anda dapat memantau metrik untuk Amazon MSK menggunakan CloudWatch konsol, baris perintah, atau API. CloudWatch Prosedur berikut menunjukkan cara mengakses metrik menggunakan berbagai metode ini.

Untuk mengakses metrik menggunakan konsol CloudWatch

Masuk ke AWS Management Console dan buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.

1. Pada panel navigasi, silakan pilih Metrik.
2. Pilih tab Semua metrik, lalu pilih AWS/Kafka.
3. Untuk melihat metrik tingkat topik, pilih Topik, ID Broker, Nama Cluster; untuk metrik tingkat broker, pilih ID Broker, Nama Cluster; dan untuk metrik tingkat cluster, pilih Nama Kluster.
4. (Opsional) Di panel grafik, pilih statistik dan periode waktu, lalu buat CloudWatch alarm menggunakan pengaturan ini.

Untuk mengakses metrik menggunakan AWS CLI

Gunakan perintah list-metrics dan get-metric-statistics.

Untuk mengakses metrik menggunakan CLI CloudWatch

Gunakan perintah mon-list-metrics dan mon-get-stats.

Untuk mengakses metrik menggunakan API CloudWatch

Gunakan operasi ListMetricsdan GetMetricStatistik.

Pemantauan kelambatan konsumen

Memantau kelambatan konsumen memungkinkan Anda mengidentifikasi konsumen yang lambat atau macet yang tidak mengikuti data terbaru yang tersedia dalam suatu topik. Bila perlu, Anda kemudian dapat mengambil tindakan perbaikan, seperti menskalakan atau me-reboot konsumen tersebut. Untuk memantau kelambatan konsumen, Anda dapat menggunakan Amazon CloudWatch atau pemantauan terbuka dengan Prometheus.

Metrik lag konsumen mengukur perbedaan antara data terbaru yang ditulis ke topik Anda dan data yang dibaca oleh aplikasi Anda. Amazon MSK menyediakan metrik kelambatan konsumen berikut, yang dapat Anda dapatkan melalui Amazon CloudWatch atau melalui pemantauan terbuka dengan Prometheus:,,,, dan. `EstimatedMaxTimeLag` `EstimatedTimeLag` `MaxOffsetLag` `OffsetLag` `SumOffsetLag` Untuk informasi selengkapnya tentang metrik ini, lihat the section called “Metrik MSK Amazon untuk pemantauan dengan CloudWatch”.

> **Note**
>
> Metrik kelambatan konsumen hanya terlihat untuk grup konsumen dalam keadaan STABIL. Grup konsumen STABIL setelah berhasil menyelesaikan penyeimbangan ulang, memastikan bahwa partisi didistribusikan secara merata di antara konsumen.

Amazon MSK mendukung metrik lag konsumen untuk cluster dengan Apache Kafka 2.2.1 atau versi yang lebih baru.

Pemantauan terbuka dengan Prometheus

Anda dapat memantau kluster MSK Anda dengan Prometheus, sistem pemantauan sumber terbuka untuk data metrik deret waktu. Anda dapat mempublikasikan data ini ke Amazon Managed Service untuk Prometheus menggunakan fitur tulis jarak jauh Prometheus. Anda juga dapat menggunakan alat yang kompatibel dengan metrik atau alat berformat Prometheus yang terintegrasi dengan Amazon MSK Open Monitoring, seperti Datadog, Lensa, Relik Baru, dan logika Sumo. Pemantauan terbuka tersedia secara gratis tetapi biaya berlaku untuk transfer data di seluruh Availability Zone. Untuk informasi tentang Prometheus, lihat dokumentasi Prometheus.

Membuat kluster MSK Amazon dengan pemantauan terbuka diaktifkan

Menggunakan AWS Management Console

1. Masuk ke AWS Management Console, dan buka konsol MSK Amazon di https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Di bagian Pemantauan, pilih kotak centang di sebelah Aktifkan pemantauan terbuka dengan Prometheus.
3. Berikan informasi yang diperlukan di semua bagian halaman, dan tinjau semua opsi yang tersedia.
4. Pilih Buat klaster.

Menggunakan AWS CLI

- Panggil perintah create-cluster dan tentukan opsinya. `open-monitoring` Aktifkan`JmxExporter`, yang`NodeExporter`, atau keduanya. Jika Anda menentukan`open-monitoring`, kedua eksportir tidak dapat dinonaktifkan secara bersamaan.

Menggunakan API

- Memanggil CreateClusteroperasi dan menentukan`OpenMonitoring`. Aktifkan`jmxExporter`, yang`nodeExporter`, atau keduanya. Jika Anda menentukan`OpenMonitoring`, kedua eksportir tidak dapat dinonaktifkan secara bersamaan.

Mengaktifkan pemantauan terbuka untuk kluster MSK Amazon yang ada

Untuk mengaktifkan pemantauan terbuka, pastikan klaster dalam `ACTIVE` status.

Menggunakan AWS Management Console

1. Masuk ke AWS Management Console, dan buka konsol MSK Amazon di https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Pilih nama cluster yang ingin Anda perbarui. Ini membawa Anda ke halaman yang berisi detail untuk cluster.
3. Pada tab Properties, gulir ke bawah untuk menemukan bagian Monitoring.
4. Pilih Edit.

5. Pilih kotak centang di sebelah Aktifkan pemantauan terbuka dengan Prometheus.
6. Pilih Simpan perubahan.

Menggunakan AWS CLI

- Panggil perintah pembaruan-pemantauan dan tentukan opsinya. `open-monitoring` Aktifkan`JmxExporter`, yang`NodeExporter`, atau keduanya. Jika Anda menentukan`open-monitoring`, kedua eksportir tidak dapat dinonaktifkan secara bersamaan.

Menggunakan API

- Memanggil UpdateMonitoringoperasi dan menentukan`OpenMonitoring`. Aktifkan`jmxExporter`, yang`nodeExporter`, atau keduanya. Jika Anda menentukan`OpenMonitoring`, kedua eksportir tidak dapat dinonaktifkan secara bersamaan.

Menyiapkan host Prometheus di instans Amazon EC2

1. Unduh server Prometheus dari ke instans Amazon EC2 https://prometheus.io/download/#prometheus Anda.
2. Ekstrak file yang diunduh ke direktori dan pergi ke direktori itu.
3. Buat file dengan konten berikut dan beri nama`prometheus.yml`.

```
# file: prometheus.yml
# my global config
global:
  scrape_interval:     60s

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped
 from this config.
  - job_name: 'prometheus'
    static_configs:
    # 9090 is the prometheus server port
    - targets: ['localhost:9090']
  - job_name: 'broker'
    file_sd_configs:
```

```
  - files:
    - 'targets.json'
```

4. Gunakan ListNodesoperasi untuk mendapatkan daftar broker cluster Anda.
5. Buat file bernama `targets.json` dengan JSON berikut. Ganti *broker_dns_1*, *broker_dns_2*, dan sisa nama DNS broker dengan nama DNS yang Anda peroleh untuk broker Anda pada langkah sebelumnya. Sertakan semua broker yang Anda peroleh pada langkah sebelumnya. Amazon MSK menggunakan port 11001 untuk JMX Exporter dan port 11002 untuk Node Exporter.

   ZooKeeper mode targets.json

```
[
  {
    "labels": {
      "job": "jmx"
    },
    "targets": [
      "broker_dns_1:11001",
      "broker_dns_2:11001",
      .
      .
      .
      "broker_dns_N:11001"
    ]
  },
  {
    "labels": {
      "job": "node"
    },
    "targets": [
      "broker_dns_1:11002",
      "broker_dns_2:11002",
      .
      .
      .
      "broker_dns_N:11002"
    ]
  }
]
```

KRaft mode targets.json

```
[
  {
    "labels": {
      "job": "jmx"
    },
    "targets": [
      "broker_dns_1:11001",
      "broker_dns_2:11001",
      .
      .
      .
      "broker_dns_N:11001",
      "controller_dns_1:11001",
      "controller_dns_2:11001",
      "controller_dns_3:11001"
    ]
  },
  {
    "labels": {
      "job": "node"
    },
    "targets": [
      "broker_dns_1:11002",
      "broker_dns_2:11002",
      .
      .
      .
      "broker_dns_N:11002"
    ]
  }
]
```

> **Note**
>
> Untuk mengikis metrik JMX dari pengontrol kRAFT, tambahkan nama DNS pengontrol sebagai target dalam file JSON. Misalnya:`controller_dns_1:11001`, mengganti `controller_dns_1` dengan nama DNS controller yang sebenarnya.

6. Untuk memulai server Prometheus di instans Amazon EC2 Anda, jalankan perintah berikut di direktori tempat Anda mengekstrak file Prometheus dan disimpan dan. `prometheus.yml` `targets.json`

```
./prometheus
```

7. Temukan alamat IP publik IPv4 dari instans Amazon EC2 tempat Anda menjalankan Prometheus pada langkah sebelumnya. Anda memerlukan alamat IP publik ini pada langkah berikut.
8. Untuk mengakses UI web Prometheus, buka browser yang dapat mengakses instans Amazon EC2 Anda, dan buka, di *mana* Prometheus-Instance-Public-IP adalah *`Prometheus-Instance-Public-IP`*`:9090` alamat IP publik yang Anda dapatkan di langkah sebelumnya.

Metrik Prometheus

Semua metrik yang dipancarkan oleh Apache Kafka ke JMX dapat diakses menggunakan pemantauan terbuka dengan Prometheus. Untuk informasi tentang metrik Apache Kafka, lihat Pemantauan dalam dokumentasi Apache Kafka. Seiring dengan metrik Apache Kafka, metrik lag konsumen juga tersedia di port 11001 dengan nama JMX mBean. `kafka.consumer.group:type=ConsumerLagMetrics` Anda juga dapat menggunakan Prometheus Node Exporter untuk mendapatkan metrik CPU dan disk untuk broker Anda di port 11002.

Menyimpan metrik Prometheus di Amazon Managed Service untuk Prometheus

Amazon Managed Service for Prometheus adalah layanan pemantauan dan peringatan yang kompatibel dengan Prometheus yang dapat Anda gunakan untuk memantau kluster MSK Amazon. Ini adalah layanan yang dikelola sepenuhnya yang secara otomatis menskalakan konsumsi, penyimpanan, kueri, dan peringatan metrik Anda. Ini juga terintegrasi dengan layanan AWS keamanan untuk memberi Anda akses cepat dan aman ke data Anda. Anda dapat menggunakan bahasa kueri PromQL sumber terbuka untuk menanyakan metrik dan memperingatkannya.

Untuk informasi selengkapnya, lihat Memulai Layanan Terkelola Amazon untuk Prometheus.

Peringatan kapasitas penyimpanan MSK Amazon

Di kluster yang disediakan MSK Amazon, Anda memilih kapasitas penyimpanan utama klaster. Jika Anda menghabiskan kapasitas penyimpanan pada broker di cluster yang disediakan, itu dapat memengaruhi kemampuannya untuk memproduksi dan mengkonsumsi data, yang menyebabkan waktu henti yang mahal. Amazon MSK menawarkan CloudWatch metrik untuk membantu Anda memantau kapasitas penyimpanan klaster Anda. Namun, untuk memudahkan Anda mendeteksi dan menyelesaikan masalah kapasitas penyimpanan, Amazon MSK secara otomatis mengirimkan peringatan kapasitas penyimpanan klaster dinamis kepada Anda. Peringatan kapasitas penyimpanan mencakup rekomendasi untuk langkah-langkah jangka pendek dan jangka panjang untuk mengelola kapasitas penyimpanan klaster Anda. Dari konsol MSK Amazon, Anda dapat menggunakan tautan cepat di dalam peringatan untuk segera mengambil tindakan yang disarankan.

Ada dua jenis peringatan kapasitas penyimpanan MSK: proaktif dan remedial.

- Peringatan kapasitas penyimpanan proaktif (“Diperlukan tindakan”) memperingatkan Anda tentang potensi masalah penyimpanan dengan klaster Anda. Ketika broker di kluster MSK telah menggunakan lebih dari 60% atau 80% dari kapasitas penyimpanan disknya, Anda akan menerima peringatan proaktif untuk broker yang terpengaruh.
- Peringatan kapasitas penyimpanan remedial (“Tindakan kritis diperlukan”) mengharuskan Anda untuk mengambil tindakan perbaikan untuk memperbaiki masalah klaster kritis ketika salah satu broker di klaster MSK Anda kehabisan kapasitas penyimpanan disk.

Amazon MSK secara otomatis mengirimkan peringatan ini ke konsol MSK Amazon, Dasbor AWS Kesehatan, Amazon EventBridge, dan kontak email untuk akun Anda. AWS Anda juga dapat mengonfigurasi Amazon EventBridge untuk mengirimkan peringatan ini ke Slack atau ke alat seperti New Relic, dan Datadog.

Peringatan kapasitas penyimpanan diaktifkan secara default untuk semua kluster yang disediakan MSK dan tidak dapat dimatikan. Fitur ini didukung di semua wilayah di mana MSK tersedia.

Memantau peringatan kapasitas penyimpanan MSK Amazon

Anda dapat memeriksa peringatan kapasitas penyimpanan dengan beberapa cara:

- Buka konsol MSK Amazon. Peringatan kapasitas penyimpanan ditampilkan di panel peringatan klaster selama 90 hari. Peringatan berisi rekomendasi dan tindakan tautan klik tunggal untuk mengatasi masalah kapasitas penyimpanan disk.

- Gunakan API ListClusters, ListClustersV2 DescribeCluster, atau DescribeClusterV2 untuk melihat `CustomerActionStatus` dan semua peringatan untuk klaster.
- Buka Dasbor AWS Kesehatan untuk melihat peringatan dari MSK dan layanan lainnya AWS .
- Siapkan AWS Health API dan Amazon EventBridge untuk merutekan pemberitahuan peringatan ke platform pihak ketiga seperti Datadog, NewRelic, dan Slack.

Menggunakan Cruise LinkedIn Control untuk Apache Kafka dengan Amazon MSK

Anda dapat menggunakan Cruise LinkedIn Control untuk menyeimbangkan kembali kluster MSK Amazon Anda, mendeteksi dan memperbaiki anomali, serta memantau keadaan dan kesehatan cluster.

Untuk mengunduh dan membangun Cruise Control

1. Buat instans Amazon EC2 di VPC Amazon yang sama dengan kluster MSK Amazon.
2. Instal Prometheus di instans Amazon EC2 yang Anda buat pada langkah sebelumnya. Perhatikan IP pribadi dan port. Nomor port default adalah 9090. Untuk informasi tentang cara mengonfigurasi Prometheus untuk menggabungkan metrik untuk klaster Anda, lihat. the section called “Pemantauan terbuka dengan Prometheus”
3. Unduh Cruise Control di instans Amazon EC2. (Atau, Anda dapat menggunakan instans Amazon EC2 terpisah untuk Cruise Control jika Anda mau.) Untuk cluster yang memiliki Apache Kafka versi 2.4.*, gunakan rilis Cruise Control 2.4.* terbaru. Jika cluster Anda memiliki versi Apache Kafka yang lebih tua dari 2.4.*, gunakan rilis Cruise Control 2.0.* terbaru.
4. Dekompresi file Cruise Control, lalu buka folder yang didekompresi.
5. Jalankan perintah berikut untuk menginstal git.

```
sudo yum -y install git
```

6. Jalankan perintah berikut untuk menginisialisasi repo lokal. Ganti *Your-Cruise-Control-Folder dengan nama folder* Anda saat ini (folder yang Anda peroleh saat Anda mendekompresi unduhan Cruise Control).

```
git init && git add . && git commit -m "Init local repo." && git tag -a Your-Cruise-Control-Folder -m "Init local version."
```

7. Jalankan perintah berikut untuk membangun kode sumber.

```
./gradlew jar copyDependantLibs
```

Untuk mengkonfigurasi dan menjalankan Cruise Control

1. Buat pembaruan berikut ke `config/cruisecontrol.properties` file. Ganti contoh server bootstrap dan string bootstrap-brokers dengan nilai untuk cluster Anda. Untuk mendapatkan string ini untuk cluster Anda, Anda dapat melihat detail cluster di konsol. Atau, Anda dapat menggunakan operasi GetBootstrapBrokersdan DescribeClusterAPI atau setara CLI mereka.

```
# If using TLS encryption, use 9094; use 9092 if using plaintext
bootstrap.servers=b-1.test-cluster.2skv42.c1.kafka.us-
east-1.amazonaws.com:9094,b-2.test-cluster.2skv42.c1.kafka.us-
east-1.amazonaws.com:9094,b-3.test-cluster.2skv42.c1.kafka.us-
east-1.amazonaws.com:9094

# SSL properties, needed if cluster is using TLS encryption
security.protocol=SSL
ssl.truststore.location=/home/ec2-user/kafka.client.truststore.jks

# Use the Prometheus Metric Sampler
metric.sampler.class=com.linkedin.kafka.cruisecontrol.monitor.sampling.prometheus.Prometheu

# Prometheus Metric Sampler specific configuration
prometheus.server.endpoint=1.2.3.4:9090 # Replace with your Prometheus IP and port

# Change the capacity config file and specify its path; details below
capacity.config.file=config/capacityCores.json
```

2. Edit `config/capacityCores.json` file untuk menentukan ukuran disk yang tepat dan inti CPU dan batas masuk/keluar jaringan. Anda dapat menggunakan operasi DescribeClusterAPI (atau setara dengan CLI) untuk mendapatkan ukuran disk. Untuk core CPU dan batas masuk/keluar jaringan, lihat Jenis Instans Amazon EC2.

```
{
  "brokerCapacities": [
    {
      "brokerId": "-1",
      "capacity": {
        "DISK": "10000",
        "CPU": {
          "num.cores": "2"
        },
        "NW_IN": "5000000",
        "NW_OUT": "5000000"
```

```
      },
      "doc": "This is the default capacity. Capacity unit used for disk is in MB,
 cpu is in number of cores, network throughput is in KB."
    }
  ]
}
```

3. Anda dapat menginstal UI Cruise Control secara opsional. Untuk mengunduhnya, buka Pengaturan Cruise Control Frontend.
4. Jalankan perintah berikut untuk memulai Cruise Control. Pertimbangkan untuk menggunakan alat seperti `screen` atau `tmux` untuk menjaga sesi yang berjalan lama tetap terbuka.

```
<path-to-your-kafka-installation>/bin/kafka-cruise-control-start.sh config/
cruisecontrol.properties 9091
```

5. Gunakan Cruise Control API atau UI untuk memastikan Cruise Control memiliki data pemuatan cluster dan membuat saran penyeimbangan kembali. Mungkin perlu beberapa menit untuk mendapatkan jendela metrik yang valid.

Template penyebaran otomatis Cruise Control untuk Amazon MSK

Anda juga dapat menggunakan CloudFormation template ini, untuk dengan mudah menyebarkan Cruise Control dan Prometheus untuk mendapatkan wawasan yang lebih dalam tentang kinerja cluster MSK Amazon Anda dan mengoptimalkan pemanfaatan sumber daya.

Fitur kunci:

- Penyediaan otomatis instans Amazon EC2 dengan Cruise Control dan Prometheus yang telah dikonfigurasi sebelumnya.
- Dukungan untuk kluster yang disediakan MSK Amazon.
- Otentikasi fleksibel dengan PlainText dan IAM.
- Tidak ada ketergantungan Zookeeper untuk Cruise Control.
- Sesuaikan target Prometheus, pengaturan kapasitas Cruise Control, dan konfigurasi lainnya dengan menyediakan file konfigurasi Anda sendiri yang disimpan dalam bucket Amazon S3.

Kuota MSK Amazon

AWS Akun Anda memiliki kuota default untuk Amazon MSK. Kecuali dinyatakan lain, setiap kuota per akun bersifat spesifik wilayah dalam akun Anda. AWS

Kuota MSK Amazon

- Hingga 90 broker per akun. 30 broker per cluster ZooKeeper mode. 60 broker per kluster mode Kraft. Untuk meminta kuota yang lebih tinggi, buka Pusat Dukungan AWS konsol dan buat kasus dukungan.
- Minimal 1 GiB penyimpanan per broker.
- Maksimal 16384 GiB penyimpanan per broker.
- Sebuah cluster yang menggunakan the section called “Kontrol akses IAM” dapat memiliki hingga 3000 koneksi TCP per broker pada waktu tertentu. Untuk meningkatkan batas ini, Anda dapat menyesuaikan `listener.name.client_iam.max.connections` atau properti `listener.name.client_iam_public.max.connections` konfigurasi menggunakan Kafka AlterConfig API atau `kafka-configs.sh` alat. Penting untuk dicatat bahwa meningkatkan salah satu properti ke nilai tinggi dapat mengakibatkan tidak tersedianya.
- Batas koneksi TCP. Dengan ledakan laju koneksi diaktifkan, MSK memungkinkan 100 koneksi per detik. Pengecualiannya adalah jenis instance kafka.t3.small, yang diizinkan 4 koneksi per detik dengan semburan laju koneksi diaktifkan. Cluster lama yang tidak mengaktifkan semburan laju koneksi akan mengaktifkan fitur secara otomatis saat cluster ditambal.

  Untuk menangani percobaan ulang pada koneksi yang gagal, Anda dapat mengatur parameter `reconnect.backoff.ms` konfigurasi di sisi klien. Misalnya, jika Anda ingin klien mencoba lagi koneksi setelah 1 detik, atur `reconnect.backoff.ms` ke 1000. Untuk informasi selengkapnya, lihat reconnect.backoff.ms di dokumentasi Apache Kafka.
- Hingga 100 konfigurasi per akun. Untuk meminta penyesuaian kuota, buka Pusat Dukungan AWS konsol dan buat kasus dukungan.
- Maksimal 50 revisi per konfigurasi.
- Untuk memperbarui konfigurasi atau versi Apache Kafka dari cluster MSK, pertama-tama pastikan jumlah partisi per broker berada di bawah batas yang dijelaskan dalam. the section called “ Ukuran kluster Anda dengan benar: Jumlah partisi per broker”

Kuota Replikator MSK

- Maksimal 15 Replikator MSK per akun.
- MSK Replicator hanya mereplikasi hingga 750 topik dalam urutan yang diurutkan. Jika Anda perlu mereplikasi lebih banyak topik, kami sarankan Anda membuat Replicator terpisah. Buka Pusat Dukungan AWS konsol dan buat kasus dukungan jika Anda memerlukan dukungan untuk lebih dari 750 topik per Replicator. Anda dapat memantau jumlah topik yang direplikasi menggunakan metrik TopicCount "".
- Throughput ingress maksimum 1GB per detik per MSK Replicator. Untuk meminta kuota yang lebih tinggi, buka Pusat Dukungan AWS konsol dan buat kasus dukungan.
- MSK Replicator Record Size - Maksimal ukuran rekaman 10MB (message.max.bytes). Untuk meminta kuota yang lebih tinggi, buka Pusat Dukungan AWS konsol dan buat kasus dukungan.

MSK Kuota Tanpa Server

> **Note**
>
> Jika Anda mengalami masalah dengan batas kuota, hubungi AWS Support dengan membuat kasus dukungan.

Batas adalah per cluster, kecuali dinyatakan lain.

Dimensi	Kuota	Hasil pelanggaran kuota
Throughput masuknya maksimum	200 MBps	Perlambatan dengan durasi throttle sebagai respons
Throughput keluar maksimum	400 MBps	Perlambatan dengan durasi throttle sebagai respons
Durasi retensi maksimum	Tidak terbatas.	N/A
Jumlah maksimum koneksi klien	3000	Koneksi dekat
Upaya koneksi maksimum	100 per detik	Koneksi dekat

Dimensi	Kuota	Hasil pelanggaran kuota
Ukuran pesan maksimal	8 MB	Permintaan gagal dengan ErrorCode: INVALID_REQUEST
Tingkat permintaan maksimum	15.000 per detik	Perlambatan dengan durasi throttle sebagai respons
Tingkat maksimum tingkat permintaan API manajemen topik	2 per detik	Perlambatan dengan durasi throttle sebagai respons
Byte pengambilan maksimum per permintaan	55 MB	Permintaan gagal dengan ErrorCode: INVALID_REQUEST
Jumlah maksimum kelompok konsumen	500	JoinGroup permintaan gagal
Jumlah maksimum partisi (pemimpin)	2400 untuk topik yang tidak dipadatkan. 120 untuk topik yang dipadatkan. Untuk meminta penyesuaian kuota, buka Pusat Dukungan AWS konsol dan buat kasus dukungan.	Permintaan gagal dengan ErrorCode: INVALID_REQUEST
Tingkat maksimum pembuatan dan penghapusan partisi	250 dalam 5 menit	Permintaan gagal dengan ErrorCode: THROUGHPUT_QUOTA_EXCEEDED
Throughput masuknya maksimum per partisi	5 MBps	Perlambatan dengan durasi throttle sebagai respons
Throughput keluar maksimum per partisi	10 MBps	Perlambatan dengan durasi throttle sebagai respons

Dimensi	Kuota	Hasil pelanggaran kuota
Ukuran partisi maksimum (untuk topik yang dipadatkan)	250 GB	Permintaan gagal dengan ErrorCode: THROUGHPUT_QUOTA_EXCEEDED
Jumlah maksimum VPC klien per cluster tanpa server	5	
Jumlah maksimum cluster tanpa server per akun	10. Untuk meminta penyesuaian kuota, buka Pusat Dukungan AWS konsol dan buat kasus dukungan.	

Kuota MSK Connect

- Hingga 100 plugin kustom.
- Hingga 100 konfigurasi pekerja.
- Hingga 60 pekerja yang terhubung. Jika konektor diatur untuk memiliki kapasitas auto scaled, maka jumlah maksimum pekerja yang telah diatur konektor adalah nomor yang digunakan MSK Connect untuk menghitung kuota akun tersebut.
- Hingga 10 pekerja per konektor.

Untuk meminta kuota MSK Connect yang lebih tinggi, buka Pusat AWS Dukungan konsol dan buat kasus dukungan.

Sumber daya MSK Amazon

Istilah sumber daya memiliki dua arti di Amazon MSK, tergantung pada konteksnya. Dalam konteks API, sumber daya adalah struktur tempat Anda dapat menjalankan operasi. Untuk daftar sumber daya ini dan operasi yang dapat Anda panggil padanya, lihat Sumber Daya di Referensi API MSK Amazon. Dalam konteksthe section called “Kontrol akses IAM”, sumber daya adalah entitas yang dapat Anda izinkan atau tolak aksesnya, seperti yang didefinisikan di the section called “Sumber daya” bagian.

Integrasi MSK

Bagian ini memberikan referensi ke AWS fitur yang terintegrasi dengan Amazon MSK.

Topik

Konektor Amazon Athena untuk Amazon MSK

Konektor Amazon Athena untuk Amazon MSK memungkinkan Amazon Athena menjalankan kueri SQL pada topik Apache Kafka. Gunakan konektor ini untuk melihat topik Apache Kafka sebagai tabel dan pesan sebagai baris di Athena.

Untuk informasi selengkapnya, lihat Konektor MSK Amazon Athena di Panduan Pengguna Amazon Athena.

Penyerapan data streaming Amazon Redshift

Amazon Redshift mendukung konsumsi streaming dari Amazon MSK. Fitur konsumsi streaming Amazon Redshift menyediakan latensi rendah, konsumsi data streaming berkecepatan tinggi dari Amazon MSK ke tampilan terwujud Amazon Redshift. Karena tidak perlu mementaskan data di Amazon S3, Amazon Redshift dapat menelan data streaming dengan latensi yang lebih rendah dan dengan biaya penyimpanan yang lebih rendah. Anda dapat mengonfigurasi konsumsi streaming Amazon Redshift di klaster Amazon Redshift menggunakan pernyataan SQL untuk mengautentikasi dan terhubung ke topik MSK Amazon.

Untuk informasi selengkapnya, lihat Konsumsi streaming di Panduan Pengembang Database Amazon Redshift.

Firehose

Amazon MSK terintegrasi dengan Firehose untuk menyediakan solusi tanpa kode server untuk mengirimkan aliran dari kluster Apache Kafka ke danau data Amazon S3. Firehose adalah layanan

ekstrak, transformasi, dan muat streaming (ETL) yang membaca data dari topik Amazon MSK Kafka Anda, melakukan transformasi seperti konversi ke Parket, dan mengumpulkan serta menulis data ke Amazon S3. Dengan beberapa klik dari konsol, Anda dapat mengatur aliran Firehose untuk membaca dari topik Kafka dan mengirimkannya ke lokasi S3. Tidak ada kode untuk ditulis, tidak ada aplikasi konektor, dan tidak ada sumber daya untuk penyediaan. Firehose secara otomatis menskalakan berdasarkan jumlah data yang dipublikasikan ke topik Kafka, dan Anda hanya membayar byte yang dicerna dari Kafka.

Lihat berikut ini untuk informasi selengkapnya tentang fitur ini.

- Menulis ke Kinesis Data Firehose Menggunakan Amazon MSK - Amazon Kinesis Data Firehose di Panduan Pengembang Amazon Data Firehose
- Blog: Amazon MSK Memperkenalkan Pengiriman Data Terkelola dari Apache Kafka ke Danau Data Anda
- Lab: Pengiriman ke Amazon S3 menggunakan Firehose

Mengakses EventBridge Pipa Amazon melalui konsol MSK Amazon

Amazon EventBridge Pipes menghubungkan sumber ke target. Pipa dimaksudkan untuk point-to-point integrasi antara sumber dan target yang didukung, dengan dukungan untuk transformasi dan pengayaan lanjutan. EventBridge Pipes menyediakan cara yang sangat skalabel untuk menghubungkan kluster MSK Amazon Anda ke AWS layanan seperti Step Functions, Amazon SQS, dan API Gateway, serta aplikasi perangkat lunak pihak ketiga sebagai layanan (SaaS) seperti Salesforce.

Untuk menyiapkan pipa, Anda memilih sumber, menambahkan pemfilteran opsional, menentukan pengayaan opsional, dan memilih target untuk data peristiwa.

Pada halaman detail untuk kluster MSK Amazon, Anda dapat melihat pipa yang menggunakan cluster itu sebagai sumbernya. Dari sana, Anda juga bisa:

- Luncurkan EventBridge konsol untuk melihat detail pipa.
- Luncurkan EventBridge konsol untuk membuat pipa baru dengan cluster sebagai sumbernya.

Untuk informasi selengkapnya tentang mengonfigurasi kluster MSK Amazon sebagai sumber pipa, lihat Amazon Managed Streaming for Apache Kafka Kafka cluster sebagai sumber di Panduan

Pengguna Amazon. EventBridge Untuk informasi lebih lanjut tentang EventBridge Pipa secara umum, lihat EventBridge Pipa.

Untuk mengakses EventBridge pipa untuk cluster MSK Amazon tertentu

1. Buka konsol MSK Amazon dan pilih Cluster.
2. Pilih cluster.
3. Pada halaman detail cluster, pilih tab Integrasi.

   Tab Integrasi mencakup daftar pipa apa pun yang saat ini dikonfigurasi untuk menggunakan cluster yang dipilih sebagai sumber, termasuk:

   - nama pipa
   - status saat ini
   - target pipa
   - ketika pipa terakhir dimodifikasi
4. Kelola pipa untuk cluster MSK Amazon Anda sesuai keinginan:

   Untuk mengakses detail lebih lanjut tentang pipa

   - Pilih pipa.

     Ini meluncurkan halaman detail Pipe EventBridge konsol.

   Untuk membuat pipa baru

   - Pilih Connect Amazon MSK cluster ke pipa.

     Ini meluncurkan halaman Buat pipa EventBridge konsol, dengan kluster MSK Amazon ditentukan sebagai sumber pipa. Untuk informasi selengkapnya, lihat Membuat EventBridge pipa di Panduan EventBridge Pengguna Amazon.

   - Anda juga dapat membuat pipa untuk cluster dari halaman Clusters. Pilih cluster, dan, dari menu Actions, pilih Create EventBridge Pipe.

Versi Apache Kafka

Saat Anda membuat cluster MSK Amazon, Anda menentukan versi Apache Kafka mana yang ingin Anda miliki di dalamnya. Anda juga dapat memperbarui versi Apache Kafka dari cluster yang ada. Topik di bagian ini membantu Anda memahami garis waktu untuk dukungan versi Kafka dan saran untuk praktik terbaik.

Topik

Versi Apache Kafka yang didukung

Amazon Managed Streaming for Apache Kafka (Amazon MSK) mendukung versi Apache Kafka dan Amazon MSK berikut. Komunitas Apache Kafka menyediakan sekitar 12 bulan dukungan untuk versi setelah tanggal rilisnya. Untuk lebih jelasnya, periksa kebijakan Apache Kafka EOL (akhir hayat).

Versi Apache Kafka yang didukung

Versi Apache Kafka	Tanggal rilis MSK	Akhir tanggal dukungan
1.1.1	--	2024-06-05
2.1.0	--	2024-06-05
2.2.1	2019-07-31	2024-06-08
2.3.1	2019-12-19	2024-06-08
2.4.1	2020-04-02	2024-06-08
2.4.1.1	2020-09-09	2024-06-08
2.5.1	2020-09-30	2024-06-08
2.6.0	2020-10-21	2024-09-11
2.6.1	2021-01-19	2024-09-11

Versi Apache Kafka	Tanggal rilis MSK	Akhir tanggal dukungan
2.6.2	2021-04-29	2024-09-11
2.6.3	2021-12-21	2024-09-11
2.7.0	2020-12-29	2024-09-11
2.7.1	2021-05-25	2024-09-11
2.7.2	2021-12-21	2024-09-11
2.8.0	--	2024-09-11
2.8.1	2022-10-28	2024-09-11
2.8.2 berjenjang	2022-10-28	Akan diumumkan
3.1.1	2022-06-22	2024-09-11
3.2.0	2022-06-22	2024-09-11
3.3.1	2022-10-26	2024-09-11
3.3.2	2023-03-02	2024-09-11
3.4.0	2023-05-04	2025-06-17
3.5.1 (disarankan)	2023-09-26	--
3.6.0	2023-11-16	--
3.7.x	2024-05-29	--

Untuk informasi selengkapnya tentang kebijakan dukungan versi MSK Amazon, lihatKebijakan dukungan versi MSK Amazon.

Apache Kafka versi 3.7.x (dengan penyimpanan berjenjang siap produksi)

Apache Kafka versi 3.7.x di MSK mencakup dukungan untuk Apache Kafka versi 3.7.0. Anda dapat membuat cluster atau meng-upgrade cluster yang ada untuk menggunakan versi 3.7.x yang baru.

Dengan perubahan penamaan versi ini, Anda tidak lagi harus mengadopsi versi perbaikan tambalan yang lebih baru seperti 3.7.1 ketika dirilis oleh komunitas Apache Kafka. Amazon MSK akan secara otomatis memperbarui 3.7.x untuk mendukung versi patch future setelah tersedia. Ini memungkinkan Anda untuk mendapatkan keuntungan dari perbaikan keamanan dan bug yang tersedia melalui versi perbaikan tambalan tanpa memicu peningkatan versi. Versi perbaikan tambalan yang dirilis oleh Apache Kafka ini tidak merusak kompatibilitas versi dan Anda dapat memperoleh manfaat dari versi perbaikan tambalan baru tanpa khawatir tentang kesalahan baca atau tulis untuk aplikasi klien Anda. Pastikan alat otomatisasi infrastruktur Anda, seperti CloudFormation, diperbarui untuk memperhitungkan perubahan penamaan versi ini.

Amazon MSK sekarang mendukung mode KraFT (Apache Kafka Raft) di Apache Kafka versi 3.7.x. Di Amazon MSK, seperti halnya dengan ZooKeeper node, pengontrol kRAFT disertakan tanpa biaya tambahan kepada Anda, dan tidak memerlukan pengaturan atau manajemen tambahan dari Anda. Anda sekarang dapat membuat cluster baik dalam mode kRAFT atau ZooKeeper mode pada Apache Kafka versi 3.7.x. Dalam mode Kraft, Anda dapat menambahkan hingga 60 broker untuk menampung lebih banyak partisi per cluster, tanpa meminta peningkatan batas, dibandingkan dengan kuota 30-broker pada cluster berbasis Zookeeper. Untuk mempelajari lebih lanjut tentang kRAFT di MSK, lihat mode kRAFT.

Apache Kafka versi 3.7.x juga mencakup beberapa perbaikan bug dan fitur baru yang meningkatkan kinerja. Perbaikan utama termasuk pengoptimalan penemuan pemimpin untuk klien dan opsi pengoptimalan flush segmen log. Untuk daftar lengkap perbaikan dan perbaikan bug, lihat catatan rilis Apache Kafka untuk 3.7.0.

Apache Kafka versi 3.6.0 (dengan penyimpanan berjenjang siap produksi)

Untuk informasi tentang Apache Kafka versi 3.6.0 (dengan penyimpanan berjenjang siap produksi), lihat catatan rilisnya di situs unduhan Apache Kafka.

Amazon MSK akan terus menggunakan dan mengelola Zookeeper untuk manajemen kuorum dalam rilis ini untuk stabilitas.

Amazon MSK versi 3.5.1

Amazon Managed Streaming for Apache Kafka (Amazon MSK) sekarang mendukung Apache Kafka versi 3.5.1 untuk cluster baru dan yang sudah ada. Apache Kafka 3.5.1 mencakup beberapa perbaikan bug dan fitur baru yang meningkatkan kinerja. Fitur utama termasuk pengenalan penugasan partisi rack-aware baru untuk konsumen. Amazon MSK akan terus menggunakan dan

mengelola Zookeeper untuk manajemen kuorum dalam rilis ini. Untuk daftar lengkap perbaikan dan perbaikan bug, lihat catatan rilis Apache Kafka untuk 3.5.1.

Untuk informasi tentang Apache Kafka versi 3.5.1, lihat catatan rilisnya di situs unduhan Apache Kafka.

Amazon MSK versi 3.4.0

Amazon Managed Streaming for Apache Kafka (Amazon MSK) sekarang mendukung Apache Kafka versi 3.4.0 untuk cluster baru dan yang sudah ada. Apache Kafka 3.4.0 mencakup beberapa perbaikan bug dan fitur baru yang meningkatkan kinerja. Fitur utama termasuk perbaikan untuk meningkatkan stabilitas untuk mengambil dari replika terdekat. Amazon MSK akan terus menggunakan dan mengelola Zookeeper untuk manajemen kuorum dalam rilis ini. Untuk daftar lengkap perbaikan dan perbaikan bug, lihat catatan rilis Apache Kafka untuk 3.4.0.

Untuk informasi tentang Apache Kafka versi 3.4.0, lihat catatan rilisnya di situs unduhan Apache Kafka.

Amazon MSK versi 3.3.2

Amazon Managed Streaming for Apache Kafka (Amazon MSK) sekarang mendukung Apache Kafka versi 3.3.2 untuk cluster baru dan yang sudah ada. Apache Kafka 3.3.2 mencakup beberapa perbaikan bug dan fitur baru yang meningkatkan kinerja. Fitur utama termasuk perbaikan untuk meningkatkan stabilitas untuk mengambil dari replika terdekat. Amazon MSK akan terus menggunakan dan mengelola Zookeeper untuk manajemen kuorum dalam rilis ini. Untuk daftar lengkap perbaikan dan perbaikan bug, lihat catatan rilis Apache Kafka untuk 3.3.2.

Untuk informasi tentang Apache Kafka versi 3.3.2, lihat catatan rilisnya di situs unduhan Apache Kafka.

Amazon MSK versi 3.3.1

Amazon Managed Streaming for Apache Kafka (Amazon MSK) sekarang mendukung Apache Kafka versi 3.3.1 untuk cluster baru dan yang sudah ada. Apache Kafka 3.3.1 mencakup beberapa perbaikan bug dan fitur baru yang meningkatkan kinerja. Beberapa fitur utama termasuk penyempurnaan metrik dan partisi. Amazon MSK akan terus menggunakan dan mengelola Zookeeper untuk manajemen kuorum dalam rilis ini untuk stabilitas. Untuk daftar lengkap perbaikan dan perbaikan bug, lihat catatan rilis Apache Kafka untuk 3.3.1.

Untuk informasi tentang Apache Kafka versi 3.3.1, lihat catatan rilisnya di situs unduhan Apache Kafka.

Amazon MSK versi 3.1.1

Amazon Managed Streaming for Apache Kafka (Amazon MSK) sekarang mendukung Apache Kafka versi 3.1.1 dan 3.2.0 untuk cluster baru dan yang sudah ada. Apache Kafka 3.1.1 dan Apache Kafka 3.2.0 mencakup beberapa perbaikan bug dan fitur baru yang meningkatkan kinerja. Beberapa fitur utama termasuk penyempurnaan metrik dan penggunaan ID topik. MSK akan terus menggunakan dan mengelola Zookeeper untuk manajemen kuorum dalam rilis ini untuk stabilitas. Untuk daftar lengkap perbaikan dan perbaikan bug, lihat catatan rilis Apache Kafka untuk 3.1.1 dan 3.2.0.

Untuk informasi tentang Apache Kafka versi 3.1.1 dan 3.2.0, lihat catatan rilis 3.2.0 dan catatan rilis 3.1.1 di situs unduhan Apache Kafka.

Amazon MSK penyimpanan berjenjang versi 2.8.2.tiered

Rilis ini adalah versi Amazon MSK-only dari Apache Kafka versi 2.8.2, dan kompatibel dengan klien Apache Kafka open source.

Rilis 2.8.2.tiered berisi fungsionalitas penyimpanan berjenjang yang kompatibel dengan API yang diperkenalkan di KIP-405 untuk Apache Kafka. Untuk informasi selengkapnya tentang fitur penyimpanan berjenjang MSK Amazon, lihat. Penyimpanan berjenjang

Apache Kafka versi 2.5.1

Apache Kafka versi 2.5.1 mencakup beberapa perbaikan bug dan fitur baru, termasuk enkripsi dalam perjalanan untuk Apache dan klien administrasi. ZooKeeper Amazon MSK menyediakan ZooKeeper titik akhir TLS, yang dapat Anda kueri dengan operasi. DescribeCluster

Output dari DescribeClusteroperasi termasuk `ZookeeperConnectStringTls` node, yang mencantumkan titik akhir zookeeper TLS.

Contoh berikut menunjukkan `ZookeeperConnectStringTls` simpul respon untuk `DescribeCluster` operasi:

```
"ZookeeperConnectStringTls": "z-3.awskafkatutorialc.abcd123.c3.kafka.us-
east-1.amazonaws.com:2182,z-2.awskafkatutorialc.abcd123.c3.kafka.us-
east-1.amazonaws.com:2182,z-1.awskafkatutorialc.abcd123.c3.kafka.us-
east-1.amazonaws.com:2182"
```

Untuk informasi tentang penggunaan enkripsi TLS dengan zookeeper, lihat. Menggunakan keamanan TLS dengan Apache ZooKeeper

Untuk informasi lebih lanjut tentang Apache Kafka versi 2.5.1, lihat catatan rilisnya di situs unduhan Apache Kafka.

Amazon MSK perbaikan bug versi 2.4.1.1

Rilis ini adalah versi perbaikan bug khusus Amazon MSK dari Apache Kafka versi 2.4.1. Rilis perbaikan bug ini berisi perbaikan untuk KAFKA-9752, masalah langka yang menyebabkan kelompok konsumen terus menyeimbangkan kembali dan tetap dalam keadaan. `PreparingRebalance` Masalah ini memengaruhi cluster yang menjalankan Apache Kafka versi 2.3.1 dan 2.4.1. Rilis ini berisi perbaikan yang diproduksi komunitas yang tersedia di Apache Kafka versi 2.5.0.

> **Note**
>
> Cluster MSK Amazon yang menjalankan versi 2.4.1.1 kompatibel dengan klien Apache Kafka yang kompatibel dengan Apache Kafka versi 2.4.1.

Kami menyarankan Anda menggunakan MSK bug-fix versi 2.4.1.1 untuk cluster MSK Amazon baru jika Anda lebih suka menggunakan Apache Kafka 2.4.1. Anda dapat memperbarui cluster yang ada yang menjalankan Apache Kafka versi 2.4.1 ke versi ini untuk menggabungkan perbaikan ini. Untuk informasi tentang memutakhirkan klaster yang ada, lihatMemperbarui versi Apache Kafka.

Untuk mengatasi masalah ini tanpa memutakhirkan cluster ke versi 2.4.1.1, lihat Kelompok konsumen terjebak di `PreparingRebalance` negara bagian bagian panduan. Memecahkan masalah kluster MSK Amazon Anda

Apache Kafka versi 2.4.1 (gunakan 2.4.1.1 sebagai gantinya)

> **Note**
>
> Anda tidak dapat lagi membuat cluster MSK dengan Apache Kafka versi 2.4.1. Sebagai gantinya, Anda dapat menggunakan Amazon MSK perbaikan bug versi 2.4.1.1 dengan klien yang kompatibel dengan Apache Kafka versi 2.4.1. Dan jika Anda sudah memiliki cluster MSK dengan Apache Kafka versi 2.4.1, kami sarankan Anda memperbaruinya untuk menggunakan Apache Kafka versi 2.4.1.1 sebagai gantinya.

KIP-392 adalah salah satu Proposal Peningkatan Kafka kunci yang termasuk dalam rilis 2.4.1 Apache Kafka. Peningkatan ini memungkinkan konsumen untuk mengambil dari replika terdekat. Untuk menggunakan fitur ini, atur `client.rack` properti konsumen ke ID Availability Zone konsumen. Contoh ID AZ adalah`use1-az1`. Amazon MSK menetapkan `broker.rack` ID dari Availability Zone broker. Anda juga harus mengatur properti `replica.selector.class` konfigurasi ke`org.apache.kafka.common.replica.RackAwareReplicaSelector`, yang merupakan implementasi dari kesadaran rak yang disediakan oleh Apache Kafka.

Saat Anda menggunakan versi Apache Kafka ini, metrik di tingkat `PER_TOPIC_PER_BROKER` pemantauan hanya muncul setelah nilainya menjadi bukan nol untuk pertama kalinya. Untuk informasi selengkapnya tentang langkah ini, lihat the section called "`PER_TOPIC_PER_BROKER`Pemantauan tingkat".

Untuk informasi tentang cara menemukan ID Availability Zone, lihat ID AZ untuk Sumber Daya Anda di panduan AWS Resource Access Manager pengguna.

Untuk informasi tentang menyetel properti konfigurasi, lihatKonfigurasi.

Untuk informasi selengkapnya tentang KIP-392, lihat Izinkan Konsumen Mengambil dari Replika Terdekat di halaman Confluence.

Untuk informasi lebih lanjut tentang Apache Kafka versi 2.4.1, lihat catatan rilisnya di situs unduhan Apache Kafka.

Dukungan versi Amazon MSK

Topik ini menjelaskan Kebijakan dukungan versi MSK Amazon dan prosedur untukMemperbarui versi Apache Kafka. Jika Anda meningkatkan versi Kafka Anda, ikuti praktik terbaik yang diuraikan di. Praktik terbaik untuk peningkatan versi

Kebijakan dukungan versi MSK Amazon

Bagian ini menjelaskan kebijakan dukungan untuk Amazon MSK mendukung versi Kafka.

- Semua versi Kafka didukung hingga mereka mencapai akhir tanggal dukungan mereka. Untuk detail tentang akhir tanggal dukungan, lihatVersi Apache Kafka yang didukung. Tingkatkan klaster MSK Anda ke versi Kafka yang direkomendasikan atau versi yang lebih tinggi sebelum akhir tanggal dukungan. Untuk detail tentang memperbarui versi Apache Kafka Anda, lihat. Memperbarui versi Apache Kafka Cluster yang menggunakan versi Kafka setelah akhir tanggal dukungannya ditingkatkan secara otomatis ke versi Kafka yang direkomendasikan.

- MSK akan menghapus dukungan untuk cluster yang baru dibuat yang menggunakan versi Kafka dengan tanggal akhir dukungan yang diterbitkan.

Memperbarui versi Apache Kafka

Anda dapat memperbarui cluster MSK yang ada ke versi Apache Kafka yang lebih baru. Anda tidak dapat memperbaruinya ke versi lama. Saat Anda memperbarui versi Apache Kafka dari cluster MSK, periksa juga perangkat lunak sisi klien Anda untuk memastikan versinya memungkinkan Anda menggunakan fitur versi Apache Kafka baru cluster. Amazon MSK hanya memperbarui perangkat lunak server. Itu tidak memperbarui klien Anda.

Untuk informasi tentang cara membuat klaster sangat tersedia selama pembaruan, lihatthe section called “Bangun cluster yang sangat tersedia”.

> **Important**
>
> Anda tidak dapat memperbarui versi Apache Kafka untuk klaster MSK yang melebihi batas yang dijelaskan dalam. the section called “ Ukuran kluster Anda dengan benar: Jumlah partisi per broker”

Memperbarui versi Apache Kafka menggunakan AWS Management Console

1. Buka konsol MSK Amazon dihttps://console.aws.amazon.com/msk/.
2. Pilih cluster MSK tempat Anda ingin memperbarui versi Apache Kafka.
3. Pada tab Properties pilih Upgrade di bagian versi Apache Kafka.

Memperbarui versi Apache Kafka menggunakan AWS CLI

1. Jalankan perintah berikut, ganti *ClusterArn*dengan Amazon Resource Name (ARN) yang Anda peroleh saat membuat cluster Anda. Jika Anda tidak memiliki ARN untuk cluster Anda, Anda dapat menemukannya dengan mencantumkan semua cluster. Untuk informasi selengkapnya, lihat the section called “Daftar cluster”.

   ```
   aws kafka get-compatible-kafka-versions --cluster-arn ClusterArn
   ```

Output dari perintah ini mencakup daftar versi Apache Kafka yang dapat Anda perbarui cluster. Sepertinya contoh berikut.

```
{
    "CompatibleKafkaVersions": [
        {
            "SourceVersion": "2.2.1",
            "TargetVersions": [
                "2.3.1",
                "2.4.1",
                "2.4.1.1",
                "2.5.1"
            ]
        }
    ]
}
```

2. Jalankan perintah berikut, ganti *ClusterArn*dengan Amazon Resource Name (ARN) yang Anda peroleh saat membuat cluster Anda. Jika Anda tidak memiliki ARN untuk cluster Anda, Anda dapat menemukannya dengan mencantumkan semua cluster. Untuk informasi selengkapnya, lihat the section called “Daftar cluster”.

   Ganti *Current-Cluster-Version dengan versi* cluster saat ini. Untuk *TargetVersion*Anda dapat menentukan salah satu versi target dari output dari perintah sebelumnya.

   > **Important**
   >
   > Versi cluster bukan bilangan bulat sederhana. Untuk menemukan versi cluster saat ini, gunakan DescribeClusteroperasi atau perintah AWS CLI deskripsi-cluster. Contoh versi adalahKTVPDKIKX0DER.

```
aws kafka update-cluster-kafka-version --cluster-arn ClusterArn --current-
version Current-Cluster-Version --target-kafka-version TargetVersion
```

Output dari perintah sebelumnya terlihat seperti JSON berikut.

```
{
```

```
    "ClusterArn": "arn:aws:kafka:us-east-1:012345678012:cluster/exampleClusterName/
abcdefab-1234-abcd-5678-cdef0123ab01-2",
    "ClusterOperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-
operation/exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
abcd-4f7f-1234-9876543210ef"
}
```

3. Untuk mendapatkan hasil `update-cluster-kafka-version` operasi, jalankan perintah berikut, ganti *`ClusterOperationArn`* dengan ARN yang Anda peroleh dalam output perintah. `update-cluster-kafka-version`

```
aws kafka describe-cluster-operation --cluster-operation-arn ClusterOperationArn
```

Output dari `describe-cluster-operation` perintah ini terlihat seperti contoh JSON berikut.

```
{
    "ClusterOperationInfo": {
        "ClientRequestId": "62cd41d2-1206-4ebf-85a8-dbb2ba0fe259",
        "ClusterArn": "arn:aws:kafka:us-east-1:012345678012:cluster/
exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2",
        "CreationTime": "2021-03-11T20:34:59.648000+00:00",
        "OperationArn": "arn:aws:kafka:us-east-1:012345678012:cluster-
operation/exampleClusterName/abcdefab-1234-abcd-5678-cdef0123ab01-2/0123abcd-
abcd-4f7f-1234-9876543210ef",
        "OperationState": "UPDATE_IN_PROGRESS",
        "OperationSteps": [
            {
                "StepInfo": {
                    "StepStatus": "IN_PROGRESS"
                },
                "StepName": "INITIALIZE_UPDATE"
            },
            {
                "StepInfo": {
                    "StepStatus": "PENDING"
                },
                "StepName": "UPDATE_APACHE_KAFKA_BINARIES"
            },
            {
                "StepInfo": {
                    "StepStatus": "PENDING"
```

```
                },
                "StepName": "FINALIZE_UPDATE"
            }
        ],
        "OperationType": "UPDATE_CLUSTER_KAFKA_VERSION",
        "SourceClusterInfo": {
            "KafkaVersion": "2.4.1"
        },
        "TargetClusterInfo": {
            "KafkaVersion": "2.6.1"
        }
    }
}
```

Jika `OperationState` memiliki nilai`UPDATE_IN_PROGRESS`, tunggu sebentar, lalu jalankan `describe-cluster-operation` perintah lagi. Ketika operasi selesai, nilai `OperationState` menjadi`UPDATE_COMPLETE`. Karena waktu yang diperlukan Amazon MSK untuk menyelesaikan operasi bervariasi, Anda mungkin perlu memeriksa berulang kali hingga operasi selesai.

Memperbarui versi Apache Kafka menggunakan API

1. Panggil GetCompatibleKafkaVersionsoperasi untuk mendapatkan daftar versi Apache Kafka yang dapat Anda perbarui cluster.
2. Panggil UpdateClusterKafkaVersionoperasi untuk memperbarui cluster ke salah satu versi Apache Kafka yang kompatibel.

Praktik terbaik untuk peningkatan versi

Untuk memastikan kontinuitas klien selama pembaruan bergulir yang dilakukan sebagai bagian dari proses peningkatan versi Kafka, tinjau konfigurasi klien Anda dan topik Apache Kafka Anda sebagai berikut:

- Tetapkan faktor replikasi topik (RF) ke nilai minimum untuk cluster dua-AZ dan nilai minimum 2 untuk cluster tiga-AZ. 3 Nilai RF 2 dapat menyebabkan partisi offline selama penambalan.
- Tetapkan replika sinkronisasi minimum (miniSR) ke nilai maksimum untuk memastikan kumpulan replika partisi dapat mentolerir satu replika yang sedang offline atau kurang direplikasi. `RF - 1`
- Konfigurasikan klien untuk menggunakan beberapa string koneksi broker. Memiliki beberapa broker dalam string koneksi klien memungkinkan untuk failover jika broker tertentu yang

mendukung klien I/O mulai ditambal. Untuk informasi tentang cara mendapatkan string koneksi dengan beberapa broker, lihat Mendapatkan broker bootstrap untuk klaster MSK Amazon.

- Kami menyarankan Anda meningkatkan menghubungkan klien ke versi yang disarankan atau lebih tinggi untuk mendapatkan manfaat dari fitur yang tersedia di versi baru. Upgrade klien tidak tunduk pada tanggal akhir masa pakai (EOL) versi Kafka klaster MSK Anda, dan tidak perlu diselesaikan pada tanggal EOL. Apache Kafka menyediakan kebijakan kompatibilitas klien dua arah yang memungkinkan klien lama bekerja dengan cluster yang lebih baru dan sebaliknya.
- Klien Kafka yang menggunakan versi 3.xx kemungkinan akan datang dengan default berikut: dan. `acks=all` `enable.idempotence=true` `acks=all`berbeda dari default sebelumnya `acks=1` dan memberikan daya tahan ekstra dengan memastikan bahwa semua replika yang sinkron mengakui permintaan produksi. Demikian pula, default untuk `enable.idempotence` sebelumnya`false`. Perubahan menjadi `enable.idempotence=true` sebagai default menurunkan kemungkinan pesan duplikat. Perubahan ini dianggap sebagai pengaturan praktik terbaik dan dapat memperkenalkan sejumlah kecil latensi tambahan yang berada dalam parameter kinerja normal.
- Gunakan versi Kafka yang direkomendasikan saat membuat cluster MSK baru. Menggunakan versi Kafka yang direkomendasikan memungkinkan Anda untuk mendapatkan keuntungan dari fitur Kafka dan MSK terbaru.

Memecahkan masalah kluster MSK Amazon Anda

Informasi berikut dapat membantu Anda memecahkan masalah yang mungkin Anda miliki dengan kluster MSK Amazon Anda. Anda juga dapat memposting masalah Anda ke AWS re:Post.

Topik

Penggantian volume menyebabkan saturasi disk karena kelebihan replikasi

Selama kegagalan perangkat keras volume yang tidak direncanakan, Amazon MSK dapat mengganti volume dengan instance baru. Kafka mengisi kembali volume baru dengan mereplikasi partisi dari broker lain di cluster. Setelah partisi direplikasi dan ditangkap, mereka memenuhi syarat untuk keanggotaan leadership dan in-sync replica (ISR).

Masalah

Dalam broker yang pulih dari penggantian volume, beberapa partisi dengan berbagai ukuran dapat kembali online sebelum yang lain. Ini bisa menjadi masalah karena partisi tersebut dapat melayani lalu lintas dari broker yang sama yang masih mengejar (mereplikasi) partisi lain. Lalu lintas replikasi ini terkadang dapat memenuhi batas throughput volume yang mendasarinya, yaitu 250 MiB per detik dalam kasus default. Ketika saturasi ini terjadi, partisi apa pun yang sudah tertangkap akan terpengaruh, menghasilkan latensi di seluruh cluster untuk setiap broker yang berbagi ISR dengan partisi yang tertangkap (bukan hanya partisi pemimpin karena acks jarak jauh). `acks=all` Masalah ini lebih sering terjadi pada cluster yang lebih besar yang memiliki jumlah partisi yang lebih besar yang ukurannya bervariasi.

Rekomendasi

- Untuk meningkatkan postur I/O replikasi, pastikan pengaturan utas praktik terbaik sudah ada.
- Untuk mengurangi kemungkinan saturasi volume yang mendasarinya, aktifkan penyimpanan yang disediakan dengan throughput yang lebih tinggi. Nilai throughput min 500 MiB/s direkomendasikan untuk kasus replikasi throughput tinggi, tetapi nilai aktual yang dibutuhkan akan bervariasi dengan throughput dan use case. Penyediaan throughput penyimpanan.
- Untuk meminimalkan tekanan replikasi, turunkan `num.replica.fetchers` ke nilai default. 2

Kelompok konsumen terjebak di **`PreparingRebalance`** negara bagian

Jika satu atau lebih grup konsumen Anda terjebak dalam keadaan penyeimbangan kembali terus-menerus, penyebabnya mungkin masalah Apache Kafka KAFKA-9752, yang memengaruhi Apache Kafka versi 2.3.1 dan 2.4.1.

Untuk mengatasi masalah ini, kami sarankan Anda meningkatkan klaster Anda keAmazon MSK perbaikan bug versi 2.4.1.1, yang berisi perbaikan untuk masalah ini. Untuk informasi tentang memperbarui klaster yang ada ke Amazon MSK bug-fix versi 2.4.1.1, lihat. Memperbarui versi Apache Kafka

Solusi untuk menyelesaikan masalah ini tanpa memutakhirkan cluster ke Amazon MSK bug-fix versi 2.4.1.1 adalah dengan mengatur klien Kafka untuk digunakanProtokol keanggotaan statis, atau ke Identifikasi dan reboot node broker koordinasi dari grup konsumen yang macet.

Menerapkan protokol keanggotaan statis

Untuk menerapkan Protokol Keanggotaan Statis di klien Anda, lakukan hal berikut:

1. Atur `group.instance.id` properti konfigurasi Konsumen Kafka Anda ke string statis yang mengidentifikasi konsumen dalam grup.
2. Pastikan bahwa contoh lain dari konfigurasi diperbarui untuk menggunakan string statis.
3. Terapkan perubahan ke Konsumen Kafka Anda.

Menggunakan Protokol Keanggotaan Statis lebih efektif jika batas waktu sesi dalam konfigurasi klien diatur ke durasi yang memungkinkan konsumen untuk pulih tanpa memicu penyeimbangan ulang grup konsumen sebelum waktunya. Misalnya, jika aplikasi konsumen Anda dapat mentolerir ketidaktersediaan 5 menit, nilai yang wajar untuk batas waktu sesi adalah 4 menit, bukan nilai default 10 detik.

> **Note**
>
> Menggunakan Protokol Keanggotaan Statis hanya mengurangi kemungkinan menghadapi masalah ini. Anda mungkin masih mengalami masalah ini bahkan saat menggunakan Protokol Keanggotaan Statis.

Mem-boot ulang node broker koordinasi

Untuk me-reboot node broker koordinator, lakukan hal berikut:

1. Identifikasi koordinator grup menggunakan `kafka-consumer-groups.sh` perintah.
2. Mulai ulang koordinator grup grup konsumen yang macet menggunakan tindakan RebootBrokerAPI.

Kesalahan saat mengirimkan log broker ke Amazon CloudWatch Logs

Saat Anda mencoba menyiapkan klaster untuk mengirim log broker ke Amazon CloudWatch Logs, Anda mungkin mendapatkan salah satu dari dua pengecualian.

Jika Anda mendapatkan `InvalidInput.LengthOfCloudWatchResourcePolicyLimitExceeded` pengecualian, coba lagi tetapi gunakan grup log yang dimulai dengan`/aws/vendedlogs/`. Untuk informasi selengkapnya, lihat Mengaktifkan Logging dari Amazon Web Services tertentu.

Jika Anda mendapatkan `InvalidInput.NumberOfCloudWatchResourcePoliciesLimitExceeded` pengecualian, pilih kebijakan CloudWatch Log Amazon yang ada di akun Anda, dan tambahkan JSON berikut ke dalamnya.

```
{"Sid":"AWSLogDeliveryWrite","Effect":"Allow","Principal":
{"Service":"delivery.logs.amazonaws.com"},"Action":
["logs:CreateLogStream","logs:PutLogEvents"],"Resource":["*"]}
```

Jika Anda mencoba menambahkan JSON di atas ke kebijakan yang ada tetapi mendapatkan kesalahan yang mengatakan Anda telah mencapai panjang maksimum untuk kebijakan yang Anda pilih, coba tambahkan JSON ke salah satu kebijakan Amazon Logs Anda yang lain. CloudWatch Setelah Anda menambahkan JSON ke kebijakan yang ada, coba sekali lagi untuk menyiapkan pengiriman broker-log ke Amazon Logs. CloudWatch

Tidak ada grup keamanan default

Jika Anda mencoba membuat klaster dan mendapatkan kesalahan yang menunjukkan bahwa tidak ada grup keamanan default, itu mungkin karena Anda menggunakan VPC yang dibagikan dengan Anda. Minta administrator Anda untuk memberi Anda izin untuk mendeskripsikan grup keamanan di VPC ini dan coba lagi. Untuk contoh kebijakan yang mengizinkan tindakan ini, lihat Amazon EC2: Mengizinkan Mengelola Grup Keamanan EC2 yang Terkait Dengan VPC Tertentu, Secara Terprogram, dan di Konsol.

Cluster tampak macet dalam status CREATING

Terkadang pembuatan cluster bisa memakan waktu hingga 30 menit. Tunggu selama 30 menit dan periksa status cluster lagi.

Status cluster berubah dari CREATING menjadi FAILED

Coba buat cluster lagi.

Status cluster AKTIF tetapi produsen tidak dapat mengirim data atau konsumen tidak dapat menerima data

- Jika pembuatan klaster berhasil (status klaster`ACTIVE`), tetapi Anda tidak dapat mengirim atau menerima data, pastikan bahwa aplikasi produsen dan konsumen Anda memiliki akses ke klaster. Untuk informasi lebih lanjut, lihat panduan dithe section called “Langkah 3: Buat mesin klien”.

- Jika produsen dan konsumen Anda memiliki akses ke cluster tetapi masih mengalami masalah dalam memproduksi dan mengkonsumsi data, penyebabnya mungkin KAFKA-7697, yang mempengaruhi Apache Kafka versi 2.1.0 dan dapat menyebabkan kebuntuan di satu atau lebih broker. Pertimbangkan untuk bermigrasi ke Apache Kafka 2.2.1, yang tidak terpengaruh oleh bug ini. Untuk informasi tentang cara bermigrasi, lihatMigrasi:.

AWS CLI tidak mengenali Amazon MSK

Jika Anda telah AWS CLI menginstal, tetapi tidak mengenali perintah MSK Amazon, tingkatkan AWS CLI ke versi terbaru. Untuk petunjuk terperinci tentang cara meng-upgrade AWS CLI, lihat Menginstal AWS Command Line Interface. Untuk informasi tentang cara menggunakan perintah AWS CLI untuk menjalankan Amazon MSK, lihatCara kerjanya.

Partisi offline atau replika tidak sinkron

Ini bisa menjadi gejala ruang disk rendah. Lihat the section called “Ruang disk hampir habis”.

Ruang disk hampir habis

Lihat praktik terbaik berikut untuk mengelola ruang disk: the section called “Memantau ruang disk” danthe section called “Sesuaikan parameter retensi data”.

Memori hampir habis

Jika Anda melihat `MemoryUsed` metrik berjalan tinggi atau `MemoryFree` hampir habis, itu tidak berarti ada masalah. Apache Kafka dirancang untuk menggunakan memori sebanyak mungkin, dan mengelolanya secara optimal.

Produser mendapat NotLeaderForPartitionException

Ini sering merupakan kesalahan sementara. Tetapkan parameter `retries` konfigurasi produsen ke nilai yang lebih tinggi dari nilai saat ini.

Partisi yang kurang direplikasi (URP) lebih besar dari nol

`UnderReplicatedPartitions`Metrik adalah salah satu yang penting untuk dipantau. Dalam cluster MSK yang sehat, metrik ini memiliki nilai 0. Jika lebih besar dari nol, itu mungkin karena salah satu alasan berikut.

- Jika `UnderReplicatedPartitions` runcing, masalahnya mungkin cluster tidak disediakan pada ukuran yang tepat untuk menangani lalu lintas masuk dan keluar. Lihat Praktik terbaik.
- Jika `UnderReplicatedPartitions` secara konsisten lebih besar dari 0 termasuk selama periode lalu lintas rendah, masalahnya mungkin Anda telah menetapkan ACL terbatas yang tidak memberikan akses topik ke broker. Untuk mereplikasi partisi, broker harus diberi wewenang untuk topik BACA dan DESKRIPSI. DESCRIBE diberikan secara default dengan otorisasi BACA. Untuk informasi tentang pengaturan ACL, lihat Otorisasi dan ACL di dokumentasi Apache Kafka.

Cluster memiliki topik yang disebut __amazon_msk_canary dan __amazon_msk_canary_state

Anda mungkin melihat bahwa klaster MSK Anda memiliki topik dengan nama `__amazon_msk_canary` dan satu lagi dengan nama`__amazon_msk_canary_state`. Ini

adalah topik internal yang dibuat dan digunakan Amazon MSK untuk kesehatan klaster dan metrik diagnostik. Topik-topik ini dapat diabaikan dalam ukuran dan tidak dapat dihapus.

Replikasi partisi gagal

Pastikan Anda belum menyetel ACL di CLUSTER_ACTIONS.

Tidak dapat mengakses klaster yang mengaktifkan akses publik

Jika klaster Anda mengaktifkan akses publik, tetapi Anda masih tidak dapat mengaksesnya dari internet, ikuti langkah-langkah berikut:

1. Pastikan aturan masuk grup keamanan klaster memungkinkan alamat IP Anda dan port cluster. Untuk daftar nomor port cluster, lihatthe section called “Informasi pelabuhan”. Juga pastikan bahwa aturan keluar grup keamanan memungkinkan komunikasi keluar. Untuk informasi selengkapnya tentang grup keamanan serta aturan masuk dan keluarnya, lihat Grup keamanan untuk VPC Anda di Panduan Pengguna Amazon VPC.
2. Pastikan alamat IP Anda dan port cluster diizinkan dalam aturan masuk ACL jaringan VPC cluster. Tidak seperti grup keamanan, ACL jaringan tidak memiliki kewarganegaraan. Ini berarti Anda harus mengonfigurasi aturan masuk dan keluar. Dalam aturan keluar, izinkan semua lalu lintas (rentang port: 0-65535) ke alamat IP Anda. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus aturan di Panduan Pengguna Amazon VPC.
3. Pastikan Anda menggunakan string bootstrap-broker akses publik untuk mengakses cluster. Kluster MSK yang memiliki akses publik diaktifkan memiliki dua string bootstrap-broker yang berbeda, satu untuk akses publik, dan satu untuk akses dari dalam. AWS Untuk informasi selengkapnya, lihat the section called “Mendapatkan broker bootstrap menggunakan AWS Management Console”.

Tidak dapat mengakses kluster dari dalam AWS: Masalah jaringan

Jika Anda memiliki aplikasi Apache Kafka yang tidak dapat berkomunikasi dengan sukses dengan kluster MSK, mulailah dengan melakukan tes konektivitas berikut.

1. Gunakan salah satu metode yang dijelaskan the section called “Mendapatkan broker bootstrap” untuk mendapatkan alamat broker bootstrap.

2. Pada perintah berikut ganti *bootstrap-broker* dengan salah satu alamat broker yang Anda peroleh pada langkah sebelumnya. Ganti *port-number* dengan 9094 jika cluster diatur untuk menggunakan otentikasi TLS. Jika cluster tidak menggunakan otentikasi TLS, ganti nomor *port* dengan 9092. Jalankan perintah dari mesin klien.

   ```
   telnet bootstrap-broker port-number
   ```

   Dimana nomor port adalah:

   - 9094 jika cluster diatur untuk menggunakan otentikasi TLS.
   - 9092 Jika cluster tidak menggunakan otentikasi TLS.
   - Nomor port yang berbeda diperlukan jika akses publik diaktifkan.

   Jalankan perintah dari mesin klien.
3. Ulangi perintah sebelumnya untuk semua broker bootstrap.

Jika mesin klien dapat mengakses broker, ini berarti tidak ada masalah konektivitas. Dalam hal ini, jalankan perintah berikut untuk memeriksa apakah klien Apache Kafka Anda sudah diatur dengan benar. Untuk mendapatkan *bootstrap-broker*, gunakan salah satu metode yang dijelaskan dalam. the section called “Mendapatkan broker bootstrap” Ganti *topik* dengan nama topik Anda.

```
<path-to-your-kafka-installation>/bin/kafka-console-producer.sh --broker-
list bootstrap-brokers --producer.config client.properties --topic topic
```

Jika perintah sebelumnya berhasil, ini berarti klien Anda diatur dengan benar. Jika Anda masih tidak dapat memproduksi dan mengkonsumsi dari aplikasi, debug masalah di tingkat aplikasi.

Jika mesin klien tidak dapat mengakses broker, lihat subbagian berikut untuk panduan yang didasarkan pada pengaturan mesin klien Anda.

Klien Amazon EC2 dan kluster MSK di VPC yang sama

Jika mesin klien berada dalam VPC yang sama dengan kluster MSK, pastikan grup keamanan klaster memiliki aturan masuk yang menerima lalu lintas dari grup keamanan mesin klien. Untuk informasi tentang mengatur aturan ini, lihat Aturan Grup Keamanan. Untuk contoh cara mengakses cluster dari instans Amazon EC2 yang berada di VPC yang sama dengan cluster, lihat. Memulai

Klien Amazon EC2 dan kluster MSK di berbagai VPC

Jika mesin klien dan cluster berada dalam dua VPC yang berbeda, pastikan hal berikut:

- Kedua VPC diintip.
- Status koneksi peering aktif.
- Tabel rute dari dua VPC diatur dengan benar.

Untuk informasi tentang peering VPC, lihat Bekerja dengan Koneksi Peering VPC.

Klien lokal

Dalam kasus klien lokal yang diatur untuk terhubung ke kluster MSK menggunakan AWS VPN, pastikan hal berikut:

- Status koneksi VPN adalahUP. Untuk informasi tentang cara memeriksa status koneksi VPN, lihat Bagaimana cara memeriksa status terowongan VPN saya saat ini? .
- Tabel rute VPC klaster berisi rute untuk CIDR lokal yang targetnya memiliki format. `Virtual private gateway(vgw-xxxxxxxx)`
- Grup keamanan klaster MSK memungkinkan lalu lintas pada port 2181, port 9092 (jika klaster Anda menerima lalu lintas teks biasa), dan port 9094 (jika klaster Anda menerima lalu lintas terenkripsi TLS).

Untuk panduan AWS VPN pemecahan masalah lainnya, lihat Pemecahan Masalah Client VPN.

AWS Direct Connect

Jika klien menggunakan AWS Direct Connect, lihat Pemecahan Masalah AWS Direct Connect.

Jika panduan pemecahan masalah sebelumnya tidak menyelesaikan masalah, pastikan tidak ada firewall yang memblokir lalu lintas jaringan. Untuk debugging lebih lanjut, gunakan alat seperti `tcpdump` dan `Wireshark` untuk menganalisis lalu lintas dan untuk memastikan bahwa itu mencapai cluster MSK.

Otentikasi gagal: Terlalu banyak koneksi

`Failed authentication ... Too many connects`Kesalahan menunjukkan bahwa broker melindungi dirinya sendiri karena satu atau lebih klien IAM mencoba menghubungkannya dengan

tingkat yang agresif. Untuk membantu broker menerima tingkat koneksi IAM baru yang lebih tinggi, Anda dapat meningkatkan parameter `reconnect.backoff.ms`konfigurasi.

Untuk mempelajari lebih lanjut tentang batas tarif untuk koneksi baru per broker, lihat Kuota MSK Amazon halaman.

MSK Tanpa Server: Pembuatan cluster gagal

Jika Anda mencoba membuat kluster MSK Tanpa Server dan alur kerja gagal, Anda mungkin tidak memiliki izin untuk membuat titik akhir VPC. Verifikasi bahwa administrator Anda telah memberi Anda izin untuk membuat titik akhir VPC dengan mengizinkan tindakan. `ec2:CreateVpcEndpoint`

Untuk daftar lengkap izin yang diperlukan untuk melakukan semua tindakan MSK Amazon, lihat. AWS kebijakan terkelola: AmazonMSK FullAccess

Praktik terbaik

Topik ini menguraikan beberapa praktik terbaik untuk diikuti saat menggunakan Amazon MSK.

Ukuran kluster Anda dengan benar: Jumlah partisi per broker

Tabel berikut menunjukkan jumlah partisi yang disarankan (termasuk replika pemimpin dan pengikut) per broker.

Ukuran broker	Jumlah partisi yang disarankan (termasuk replika pemimpin dan pengikut) per broker
`kafka.t3.small`	300
`kafka.m5.large` atau `kafka.m5.xlarge`	1000
`kafka.m5.2xlarge`	2000
`kafka.m5.4xlarge` ,`kafka.m5.8xlarge` ,`kafka.m5.12xlarge` ,`kafka.m5.16xlarge` , atau `kafka.m5.24xlarge`	4000
`kafka.m7g.large` atau `kafka.m7g.xlarge`	1000
`kafka.m7g.2xlarge`	2000
`kafka.m7g.4xlarge` ,`kafka.m7g.8xlarge` ,`kafka.m7g.12xlarge` , atau `kafka.m7g.16xlarge`	4000

Jika jumlah partisi per broker melebihi nilai yang disarankan dan klaster Anda menjadi kelebihan beban, Anda mungkin dicegah melakukan operasi berikut:

- Perbarui konfigurasi cluster
- Perbarui cluster ke ukuran broker yang lebih kecil

- Kaitkan AWS Secrets Manager rahasia dengan cluster yang memiliki otentikasi SASL/SCRAM

Jumlah partisi yang tinggi juga dapat mengakibatkan metrik Kafka yang hilang pada dan CloudWatch pada pengikisan Prometheus.

Untuk panduan memilih jumlah partisi, lihat Apache Kafka Mendukung 200K Partisi Per Cluster. Kami juga menyarankan Anda melakukan pengujian sendiri untuk menentukan ukuran yang tepat untuk broker Anda. Untuk informasi lebih lanjut tentang ukuran broker yang berbeda, lihatthe section called "Ukuran broker".

Ukuran klaster Anda dengan benar: Jumlah broker per cluster

Untuk menentukan jumlah broker yang tepat untuk klaster MSK Anda dan memahami biaya, lihat spreadsheet Ukuran dan Harga MSK. Spreadsheet ini memberikan perkiraan untuk ukuran cluster MSK dan biaya terkait Amazon MSK dibandingkan dengan cluster Apache Kafka berbasis EC2 yang serupa dan dikelola sendiri. Untuk informasi lebih lanjut tentang parameter input di spreadsheet, arahkan kursor ke deskripsi parameter. Perkiraan yang diberikan oleh lembar ini bersifat konservatif dan memberikan titik awal untuk cluster baru. Kinerja klaster, ukuran, dan biaya tergantung pada kasus penggunaan Anda dan kami sarankan Anda memverifikasi mereka dengan pengujian yang sebenarnya.

Untuk memahami bagaimana infrastruktur yang mendasarinya memengaruhi kinerja Apache Kafka, lihat Praktik terbaik untuk mengukur kluster Apache Kafka Anda dengan benar untuk mengoptimalkan kinerja dan biaya di Blog Big Data. AWS Posting blog memberikan informasi tentang cara mengukur cluster Anda untuk memenuhi persyaratan throughput, ketersediaan, dan latensi Anda. Ini juga memberikan jawaban atas pertanyaan seperti kapan Anda harus meningkatkan versus skala keluar, dan panduan tentang cara terus memverifikasi ukuran cluster produksi Anda.

Optimalkan throughput cluster untuk instans m5.4xl, m7g.4xl, atau yang lebih besar

Saat menggunakan m5.4xl, m7g.4xl, atau instance yang lebih besar, Anda dapat mengoptimalkan throughput cluster dengan menyetel konfigurasi num.io.threads dan num.network.threads.

Num.io.Threads adalah jumlah utas yang digunakan broker untuk memproses permintaan. Menambahkan lebih banyak thread, hingga jumlah core CPU yang didukung untuk ukuran instans, dapat membantu meningkatkan throughput cluster.

Num.network.Threads adalah jumlah thread yang digunakan broker untuk menerima semua permintaan masuk dan mengembalikan tanggapan. Thread jaringan menempatkan permintaan masuk pada antrian permintaan untuk diproses oleh io.threads. Menyetel num.network.threads ke setengah jumlah inti CPU yang didukung untuk ukuran instans memungkinkan penggunaan penuh ukuran instans baru.

> **Important**
>
> Jangan menambah num.network.threads tanpa terlebih dahulu meningkatkan num.io.threads karena ini dapat menyebabkan kemacetan terkait saturasi antrian.

Pengaturan yang disarankan

Ukuran instans	Nilai yang disarankan untuk num.io.threads	Nilai yang disarankan untuk num.network.threads
m5.4xl	16	8
m5.8xl	32	16
m5.12xl	48	24
m5.16xl	64	32
m5.24xl	96	48
m7g.4xlarge	16	8
m7g.8xlarge	32	16
m7g.12xlarge	48	24
m7g.16xlarge	64	32

Gunakan Kafka terbaru AdminClient untuk menghindari masalah ketidakcocokan ID topik

ID topik hilang (Kesalahan: tidak cocok dengan Id topik untuk partisi) saat Anda menggunakan AdminClient versi Kafka yang lebih rendah dari 2.8.0 dengan bendera `--zookeeper` untuk menambah atau menetapkan kembali partisi topik untuk cluster menggunakan Kafka versi 2.8.0 atau lebih tinggi. Perhatikan bahwa `--zookeeper` bendera tidak digunakan lagi di Kafka 2.5 dan dihapus dimulai dengan Kafka 3.0. Lihat Memutakhirkan ke 2.5.0 dari versi 0.8.x apa pun hingga 2.4.x.

Untuk mencegah ketidakcocokan ID topik, gunakan klien Kafka versi 2.8.0 atau lebih tinggi untuk operasi admin Kafka. Atau, klien 2.5 dan lebih tinggi dapat menggunakan `--bootstrap-servers` bendera alih-alih `--zookeeper` bendera.

Bangun cluster yang sangat tersedia

Gunakan rekomendasi berikut sehingga kluster MSK Anda dapat sangat tersedia selama pembaruan (seperti ketika Anda memperbarui ukuran broker atau versi Apache Kafka, misalnya) atau ketika Amazon MSK mengganti broker.

- Siapkan cluster tiga-AZ.
- Pastikan faktor replikasi (RF) minimal 3. Perhatikan bahwa RF 1 dapat menyebabkan partisi offline selama pembaruan bergulir; dan RF 2 dapat menyebabkan hilangnya data.
- Tetapkan replika sinkronisasi minimum (miniSR) ke paling banyak RF - 1. MiniSR yang sama dengan RF dapat mencegah produksi ke cluster selama pembaruan bergulir. MiniSR 2 memungkinkan topik yang direplikasi tiga arah tersedia saat satu replika sedang offline.
- Pastikan string koneksi klien mencakup setidaknya satu broker dari setiap zona ketersediaan. Memiliki beberapa broker dalam string koneksi klien memungkinkan untuk failover ketika broker tertentu offline untuk pembaruan. Untuk informasi tentang cara mendapatkan string koneksi dengan beberapa broker, lihatthe section called “Mendapatkan broker bootstrap”.

Pantau penggunaan CPU

Amazon MSK sangat menyarankan agar Anda mempertahankan total pemanfaatan CPU untuk broker Anda (didefinisikan sebagai`CPU User + CPU System`) di bawah 60%. Ketika Anda memiliki setidaknya 40% dari total CPU cluster Anda yang tersedia, Apache Kafka dapat mendistribusikan kembali beban CPU di seluruh broker di cluster bila diperlukan. Salah satu contoh kapan ini

diperlukan adalah ketika Amazon MSK mendeteksi dan pulih dari kesalahan broker; dalam hal ini, Amazon MSK melakukan pemeliharaan otomatis, seperti menambal. Contoh lain adalah ketika pengguna meminta perubahan ukuran broker atau peningkatan versi; dalam dua kasus ini, Amazon MSK menyebarkan alur kerja bergulir yang membuat satu broker offline pada satu waktu. Ketika broker dengan partisi prospek offline, Apache Kafka menugaskan kembali kepemimpinan partisi untuk mendistribusikan kembali pekerjaan ke broker lain di cluster. Dengan mengikuti praktik terbaik ini, Anda dapat memastikan Anda memiliki ruang kepala CPU yang cukup di cluster Anda untuk mentolerir peristiwa operasional seperti ini.

Anda dapat menggunakan matematika CloudWatch metrik Amazon untuk membuat metrik kompositCPU `User + CPU System`. Atur alarm yang dipicu ketika metrik komposit mencapai pemanfaatan CPU rata-rata 60%. Saat alarm ini dipicu, skala cluster menggunakan salah satu opsi berikut:

- Opsi 1 (disarankan): Perbarui ukuran broker Anda ke ukuran yang lebih besar berikutnya. Misalnya, jika ukuran saat inikafka.m5.large, perbarui cluster yang akan digunakankafka.m5.xlarge. Perlu diingat bahwa ketika Anda memperbarui ukuran broker di cluster, Amazon MSK membawa broker offline secara bergulir dan sementara menugaskan kembali kepemimpinan partisi ke broker lain. Pembaruan ukuran biasanya memakan waktu 10-15 menit per broker.
- Opsi 2: Jika ada topik dengan semua pesan yang dicerna dari produsen yang menggunakan penulisan round-robin (dengan kata lain, pesan tidak dikunci dan pemesanan tidak penting bagi konsumen), perluas cluster Anda dengan menambahkan broker. Tambahkan juga partisi ke topik yang ada dengan throughput tertinggi. Selanjutnya, gunakan `kafka-topics.sh --describe` untuk memastikan bahwa partisi yang baru ditambahkan ditugaskan ke broker baru. Manfaat utama dari opsi ini dibandingkan dengan yang sebelumnya adalah Anda dapat mengelola sumber daya dan biaya lebih terperinci. Selain itu, Anda dapat menggunakan opsi ini jika beban CPU secara signifikan melebihi 60% karena bentuk penskalaan ini biasanya tidak menghasilkan peningkatan beban pada broker yang ada.
- Opsi 3: Perluas cluster Anda dengan menambahkan broker, lalu tetapkan kembali partisi yang ada dengan menggunakan alat penugasan partisi bernama. `kafka-reassign-partitions.sh` Namun, jika Anda menggunakan opsi ini, cluster perlu menghabiskan sumber daya untuk mereplikasi data dari broker ke broker setelah partisi dipindahkan. Dibandingkan dengan dua opsi sebelumnya, ini dapat secara signifikan meningkatkan beban pada cluster pada awalnya. Akibatnya, Amazon MSK tidak merekomendasikan penggunaan opsi ini ketika pemanfaatan CPU di atas 70% karena replikasi menyebabkan beban CPU tambahan dan lalu lintas jaringan. Amazon MSK hanya merekomendasikan penggunaan opsi ini jika dua opsi sebelumnya tidak layak.

Rekomendasi lainnya:

- Pantau total pemanfaatan CPU per broker sebagai proxy untuk distribusi beban. Jika broker memiliki penggunaan CPU yang tidak merata secara konsisten, itu mungkin merupakan tanda bahwa beban tidak didistribusikan secara merata di dalam cluster. Amazon MSK merekomendasikan penggunaan Cruise Control untuk terus mengelola distribusi beban melalui penetapan partisi.
- Pantau produksi dan konsumsi latensi. Menghasilkan dan mengkonsumsi latensi dapat meningkat secara linier dengan pemanfaatan CPU.
- Interval pengikisan JMX: Jika Anda mengaktifkan pemantauan terbuka dengan fitur Prometheus, Anda disarankan untuk menggunakan interval gesekan 60 detik atau lebih tinggi (scrape_interval: 60s) untuk konfigurasi host Prometheus Anda (prometheus.yl). Menurunkan interval scrape dapat menyebabkan penggunaan CPU yang tinggi pada cluster Anda.

Memantau ruang disk

Untuk menghindari kehabisan ruang disk untuk pesan, buat CloudWatch alarm yang mengawasi `KafkaDataLogsDiskUsed` metrik. Ketika nilai metrik ini mencapai atau melebihi 85%, lakukan satu atau lebih tindakan berikut:

- Gunakan the section called “Penskalaan Otomatis”. Anda juga dapat meningkatkan penyimpanan broker secara manual seperti yang dijelaskan dalamthe section called “Penskalaan manual”.
- Kurangi periode penyimpanan pesan atau ukuran log. Untuk informasi tentang cara melakukannya, lihatthe section called “Sesuaikan parameter retensi data”.
- Hapus topik yang tidak digunakan.

Untuk informasi tentang cara mengatur dan menggunakan alarm, lihat Menggunakan CloudWatch Alarm Amazon. Untuk daftar lengkap metrik MSK Amazon, lihat. Memantau klaster

Sesuaikan parameter retensi data

Mengkonsumsi pesan tidak menghapusnya dari log. Untuk mengosongkan ruang disk secara teratur, Anda dapat secara eksplisit menentukan periode waktu penyimpanan, yaitu berapa lama pesan tetap berada di log. Anda juga dapat menentukan ukuran log retensi. Ketika periode waktu retensi atau ukuran log retensi tercapai, Apache Kafka mulai menghapus segmen yang tidak aktif dari log.

Untuk menentukan kebijakan retensi di tingkat klaster, tetapkan satu atau beberapa parameter berikut:`log.retention.hours`,`log.retention.minutes`,`log.retention.ms`, atau`log.retention.bytes`. Untuk informasi selengkapnya, lihat the section called “Konfigurasi kustom”.

Anda juga dapat menentukan parameter retensi di tingkat topik:

- Untuk menentukan periode waktu retensi per topik, gunakan perintah berikut.

```
kafka-configs.sh --bootstrap-server $bs --alter --entity-type topics --entity-
name TopicName --add-config retention.ms=DesiredRetentionTimePeriod
```

- Untuk menentukan ukuran log retensi per topik, gunakan perintah berikut.

```
kafka-configs.sh --bootstrap-server $bs --alter --entity-type topics --entity-
name TopicName --add-config retention.bytes=DesiredRetentionLogSize
```

Parameter retensi yang Anda tentukan pada tingkat topik lebih diutamakan daripada parameter tingkat kluster.

Mempercepat pemulihan log setelah shutdown yang tidak bersih

Setelah shutdown yang tidak bersih, broker dapat mengambil beberapa saat untuk memulai kembali seperti halnya pemulihan log. Secara default, Kafka hanya menggunakan satu utas per direktori log untuk melakukan pemulihan ini. Misalnya, jika Anda memiliki ribuan partisi, pemulihan log dapat memakan waktu berjam-jam untuk diselesaikan. Untuk mempercepat pemulihan log, disarankan untuk menambah jumlah thread menggunakan properti konfigurasi `num.recovery.threads.per.data.dir`. Anda dapat mengaturnya ke jumlah core CPU.

Memantau memori Apache Kafka

Kami menyarankan Anda memantau memori yang digunakan Apache Kafka. Jika tidak, cluster mungkin menjadi tidak tersedia.

Untuk menentukan berapa banyak memori yang digunakan Apache Kafka, Anda dapat memantau metrik. `HeapMemoryAfterGC` `HeapMemoryAfterGC`adalah persentase dari total memori heap yang digunakan setelah pengumpulan sampah. Kami menyarankan Anda membuat CloudWatch alarm yang mengambil tindakan ketika `HeapMemoryAfterGC` meningkat di atas 60%.

Langkah-langkah yang dapat Anda ambil untuk mengurangi penggunaan memori bervariasi. Mereka bergantung pada cara Anda mengkonfigurasi Apache Kafka. Misalnya, jika Anda menggunakan pengiriman pesan transaksional, Anda dapat mengurangi `transactional.id.expiration.ms` nilai dalam konfigurasi Apache Kafka Anda dari `604800000` ms ke `86400000` ms (dari 7 hari menjadi 1 hari). Ini mengurangi jejak memori setiap transaksi.

Jangan tambahkan broker non-MSK

Untuk cluster ZooKeeper berbasis, jika Anda menggunakan ZooKeeper perintah Apache untuk menambahkan broker, broker ini tidak ditambahkan ke cluster MSK Anda, dan Apache Anda ZooKeeper akan berisi informasi yang salah tentang cluster. Hal ini dapat mengakibatkan hilangnya data. Untuk operasi klaster yang didukung, lihatCara kerjanya.

Aktifkan enkripsi dalam transit

Untuk informasi tentang enkripsi dalam perjalanan dan cara mengaktifkannya, lihatthe section called “Enkripsi bergerak”.

Tetapkan kembali partisi

Untuk memindahkan partisi ke broker yang berbeda pada cluster yang sama, Anda dapat menggunakan alat penugasan partisi bernama. `kafka-reassign-partitions.sh` Misalnya, setelah Anda menambahkan broker baru untuk memperluas cluster atau memindahkan partisi untuk menghapus broker, Anda dapat menyeimbangkan kembali cluster itu dengan menugaskan kembali partisi ke broker baru. Untuk informasi tentang cara menambahkan broker ke klaster, lihatthe section called “Memperluas cluster”. Untuk informasi tentang cara menghapus broker dari klaster, lihatthe section called “Hapus broker”. Untuk informasi tentang alat penugasan kembali partisi, lihat Memperluas cluster Anda di dokumentasi Apache Kafka.

Riwayat dokumen untuk Panduan Pengembang MSK Amazon

Tabel berikut menjelaskan perubahan penting pada Panduan Pengembang MSK Amazon.

Pembaruan dokumentasi terbaru: 25 Juni 2024

Perubahan	Deskripsi	Tanggal
Fitur upgrade Graviton di tempat ditambahkan.	Anda dapat memperbarui ukuran broker cluster Anda dari M5 atau T3 ke m7g, atau dari m7g ke M5.	2024-6-25
3.4.0 akhir tanggal dukungan diumumkan.	Akhir tanggal dukungan untuk Apache Kafka versi 3.4.0 adalah 17 Juni 2025.	2024-6-24
Fitur penghapusan broker ditambahkan.	Anda dapat mengurangi kapasitas penyimpanan dan komputasi klaster yang disediakan dengan menghapus kumpulan broker, tanpa dampak ketersediaan, risiko daya tahan data, atau gangguan pada aplikasi streaming data Anda.	2024-5-16
`WriteDataIdempotently` ditambahkan ke AWSMSKReplicatorExecutionRole	WriteDataIdempotently izin ditambahkan ke AWSMSKReplicatorExecutionRole kebijakan untuk mendukung replikasi data antara kluster MSK.	2024-5-16
Broker Graviton M7g dirilis di Brasil dan Bahrain.	Amazon MSK sekarang mendukung Amerika Selatan (sa-east-1, São Paulo) dan	2024-2-07

Perubahan	Deskripsi	Tanggal
	Timur Tengah (me-south-1, Bahrain) ketersediaan wilayah broker M7G menggunakan prosesor Graviton (prosesor berbasis ARM khusus yang dibangun oleh Amazon Web Services). AWS	
Lepaskan broker Graviton M7g ke wilayah China	Amazon MSK sekarang mendukung ketersediaan broker m7g di wilayah China menggunakan prosesor AWS Graviton (prosesor berbasis ARM khusus yang dibuat oleh Amazon Web Services).	2024-01-11
Kebijakan dukungan versi Amazon MSK Kafka	Menambahkan penjelasan tentang kebijakan dukungan versi Kafka yang didukung MSK Amazon. Untuk informasi selengkapnya, lihat versi Apache Kafka.	2023-12-08
Kebijakan peran eksekusi layanan baru untuk mendukung Amazon MSK Replicator.	Amazon MSK menambahkan `AWSMSKReplicatorExecutionRole` kebijakan baru untuk mendukung Amazon MSK Replicator. Untuk informasi selengkapnya, lihat kebijakan AWS terkelola : `AWSMSKReplicatorExecutionRole`.	2023-12-06

Perubahan	Deskripsi	Tanggal
Dukungan Graviton M7g	Amazon MSK sekarang mendukung broker M7g menggunakan prosesor AWS Graviton (prosesor berbasis ARM khusus yang dibuat oleh Amazon Web Services).	2023-11-27
Replikator MSK Amazon	Amazon MSK Replicato r adalah fitur baru yang dapat Anda gunakan untuk mereplikasi data antara kluster MSK Amazon. Amazon MSK Replicator menyertak an pembaruan kebijakan FullAccess AmazonMSK. Untuk informasi selengkapnya, lihat kebijakan AWS terkelola: `AmazonMSKFullAccess`.	2023-09-28
Diperbarui untuk praktik terbaik IAM.	Panduan yang diperbarui untuk menyelaraskan dengan praktik terbaik IAM. Untuk informasi selengkapnya, lihat Praktik terbaik keamanan di IAM.	2023-03-08

Perubahan	Deskripsi	Tanggal
Pembaruan peran terkait layanan untuk mendukung konektivitas pribadi multi-VPC	Amazon MSK sekarang menyertakan pembaruan peran AWSServiceRoleForKafka terkait layanan untuk mengelola antarmuka jaringan dan titik akhir VPC di akun Anda yang membuat broker klaster dapat diakses oleh klien di VPC Anda. Amazon MSK menggunakan izin untuk `DescribeVpcEndpoints`, `ModifyVpcEndpoint` dan. `DeleteVpcEndpoints` Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan untuk Amazon MSK.	2023-03-08
Support untuk Apache Kafka 2.7.2	Amazon MSK sekarang mendukung Apache Kafka versi 2.7.2. Untuk informasi selengkapnya, lihat Versi Apache Kafka yang didukung.	2021-12-21
Support untuk Apache Kafka 2.6.3	Amazon MSK sekarang mendukung Apache Kafka versi 2.6.3. Untuk informasi selengkapnya, lihat Versi Apache Kafka yang didukung.	2021-12-21

Perubahan	Deskripsi	Tanggal
MSK Prarilis Tanpa Server	MSK Serverless adalah fitur baru yang dapat Anda gunakan untuk membuat cluster tanpa server. Untuk informasi selengkapnya, lihat MSK Tanpa Server.	2021-11-29
Support untuk Apache Kafka 2.8.1	Amazon MSK sekarang mendukung Apache Kafka versi 2.8.1. Untuk informasi selengkapnya, lihat Versi Apache Kafka yang didukung.	2021-09-30
MSK Connect	MSK Connect adalah fitur baru yang dapat Anda gunakan untuk membuat dan mengelola konektor Apache Kafka. Untuk informasi selengkapnya, lihat MSK Connect.	2021-09-16
Support untuk Apache Kafka 2.7.1	Amazon MSK sekarang mendukung Apache Kafka versi 2.7.1. Untuk informasi selengkapnya, lihat Versi Apache Kafka yang didukung.	2021-05-25
Support untuk Apache Kafka 2.8.0	Amazon MSK sekarang mendukung Apache Kafka versi 2.8.0. Untuk informasi selengkapnya, lihat Versi Apache Kafka yang didukung.	2021-04-28

Perubahan	Deskripsi	Tanggal
Support untuk Apache Kafka 2.6.2	Amazon MSK sekarang mendukung Apache Kafka versi 2.6.2. Untuk informasi selengkapnya, lihat Versi Apache Kafka yang didukung.	2021-04-28
Support untuk Memperbarui Jenis Broker	Anda sekarang dapat mengubah tipe broker untuk klaster yang ada. Untuk informasi selengkapnya, lihat Memperbarui ukuran broker.	2021-01-21
Support untuk Apache Kafka 2.6.1	Amazon MSK sekarang mendukung Apache Kafka versi 2.6.1. Untuk informasi selengkapnya, lihat Versi Apache Kafka yang didukung.	2021-01-19
Support untuk Apache Kafka 2.7.0	Amazon MSK sekarang mendukung Apache Kafka versi 2.7.0. Untuk informasi selengkapnya, lihat Versi Apache Kafka yang didukung.	2020-12-29

Perubahan	Deskripsi	Tanggal
Tidak Ada Cluster Baru dengan Apache Kafka Versi 1.1.1	Anda tidak dapat lagi membuat cluster MSK Amazon baru dengan Apache Kafka versi 1.1.1. Namun, jika Anda memiliki kluster MSK yang menjalankan Apache Kafka versi 1.1.1, Anda dapat terus menggunakan semua fitur yang saat ini didukung pada cluster yang ada. Untuk informasi selengkapnya, lihat Versi Apache Kafka.	2020-11-24
Metrik Lag Konsumen	Amazon MSK sekarang menyediakan metrik yang dapat Anda gunakan untuk memantau kelambatan konsumen. Untuk informasi selengkapnya, lihat Memantau kluster MSK Amazon.	2020-11-23
Support untuk Cruise Control	Amazon MSK sekarang mendukung Cruise LinkedIn Control. Untuk informasi selengkapnya, lihat Menggunakan Cruise LinkedIn Control untuk Apache Kafka dengan Amazon MSK.	2020-11-17
Support untuk Apache Kafka 2.6.0	Amazon MSK sekarang mendukung Apache Kafka versi 2.6.0. Untuk informasi selengkapnya, lihat Versi Apache Kafka yang didukung.	2020-10-21

Perubahan	Deskripsi	Tanggal
Support untuk Apache Kafka 2.5.1	Amazon MSK sekarang mendukung Apache Kafka versi 2.5.1. Dengan Apache Kafka versi 2.5.1, Amazon MSK mendukung enkripsi dalam perjalanan antara klien dan titik akhir. ZooKeeper Untuk informasi selengkapnya, lihat Versi Apache Kafka yang didukung.	2020-09-30
Ekspansi Otomatis Aplikasi	Anda dapat mengonfigurasi Amazon Managed Streaming for Apache Kafka untuk memperluas penyimpan an klaster secara otomatis sebagai respons terhadap peningkatan penggunaan. Untuk informasi selengkapnya, lihat Penskalaan Otomatis.	2020-09-30
Support untuk Username dan Password Security	Amazon MSK sekarang mendukung login ke cluster menggunakan username dan password. Amazon MSK menyimpan kredensyal di Secrets Manager AWS . Untuk informasi selengkapnya, lihat Otentikasi SASL/SCRAM.	2020-09-17
Support untuk Upgrade Versi Apache Kafka dari Cluster MSK Amazon	Anda sekarang dapat meningkatkan versi Apache Kafka dari cluster MSK yang ada.	2020-05-28

Perubahan	Deskripsi	Tanggal
Support untuk T3.Small Broker Nodes	Amazon MSK sekarang mendukung pembuatan cluster dengan broker Amazon EC2 tipe T3.small.	2020-04-08
Support untuk Apache Kafka 2.4.1	Amazon MSK sekarang mendukung Apache Kafka versi 2.4.1.	2020-04-02
Support untuk Streaming Broker Log	Amazon MSK sekarang dapat melakukan streaming log broker ke CloudWatch Log, Amazon S3, dan Amazon Data Firehose. Firehose dapat, pada gilirannya, mengirimkan log ini ke tujuan yang didukungnya, seperti OpenSearch Layanan.	2020-02-25
Support untuk Apache Kafka 2.3.1	Amazon MSK sekarang mendukung Apache Kafka versi 2.3.1.	2019-12-19
Pemantauan Terbuka	Amazon MSK sekarang mendukung pemantauan terbuka dengan Prometheus.	2019-12-04
Support untuk Apache Kafka 2.2.1	Amazon MSK sekarang mendukung Apache Kafka versi 2.2.1.	2019-07-31
Ketersediaan Umum	Fitur baru termasuk dukungan penandaan, otentikasi, enkripsi TLS, konfigurasi, dan kemampuan untuk memperbarui penyimpanan broker.	2019-05-30

Perubahan	Deskripsi	Tanggal
Support untuk Apache Kafka 2.1.0	Amazon MSK sekarang mendukung Apache Kafka versi 2.1.0.	2019-02-05

AWS Glosarium

Untuk AWS terminologi terbaru, lihat AWS glosarium di Referensi.Glosarium AWS

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.