aws

Guia do Desenvolvedor

Amazon Managed Streaming for Apache Kafka

Amazon Managed Streaming for Apache Kafka: Guia do Desenvolvedor

Table of Contents

Boas-vindas ao Guia do desenvolvedor do Amazon MSK

Boas-vindas ao Guia do desenvolvedor do Amazon MSK. Os tópicos a seguir podem ajudar você a começar a usar este guia com base no que você estiver tentando fazer.

- Crie um cluster do Amazon MSK seguindo o tutorial Conceitos básicos sobre como usar o Amazon MSK.
- Aprofunde-se na funcionalidade do Amazon MSK em Amazon MSK: funcionamento.
- Execute o Apache Kafka sem precisar gerenciar e escalar a capacidade do cluster com MSK Serverless.
- Use o MSK Connect para transmitir dados de e para seu cluster do Apache Kafka.
- Use Replicador do MSK para replicar dados de forma confiável em clusters do Amazon MSK em regiões diferentes ou na mesma AWS região.

Para os destaques, detalhes do produto e preços, consulte a página de serviços do Amazon MSK.

O que é o Amazon MSK?

O Amazon Managed Streaming for Apache Kafka (Amazon MSK) é um serviço totalmente gerenciado que o habilita a criar e executar aplicações que usam o Apache Kafka para processar dados de transmissões. O Amazon MSK fornece as operações do ambiente de gerenciamento, como as operações para criar, atualizar e excluir clusters. Ele permite usar operações do plano de dados do Apache Kafka, como aqueles para produzir e consumir dados. Ele executa versões de código aberto do Apache Kafka. Isso significa que aplicativos, ferramentas e plug-ins existentes de parceiros e da comunidade Apache Kafka são compatíveis sem a necessidade de fazer alterações no código do aplicativo. É possível usar o Amazon MSK para criar clusters com qualquer uma das versões do Apache Kafka listadas em the section called “Versões compatíveis do Apache Kafka”.

Esses componentes descrevem a arquitetura do Amazon MSK:

- Nós de agente: ao criar um cluster do Amazon MSK, especifique quantos nós de agente você deseja que o Amazon MSK crie em cada zona de disponibilidade. O mínimo é um corretor por zona de disponibilidade. Cada zona de disponibilidade tem sua própria sub-rede de nuvem privada virtual (VPC).

- ZooKeeper nós — O Amazon MSK também cria os ZooKeeper nós Apache para você. O Apache ZooKeeper é um servidor de código aberto que permite uma coordenação distribuída altamente confiável.
- Controladores Kraft — A comunidade Apache Kafka desenvolveu o Kraft para substituir o Apache no gerenciamento de metadados nos clusters do Apache ZooKeeper Kafka. No modo Kraft, os metadados do cluster são propagados dentro de um grupo de controladores Kafka, que fazem parte do cluster Kafka, em vez de entre nós. ZooKeeper Os controladores Kraft estão incluídos sem custo adicional para você e não exigem configuração ou gerenciamento adicionais de sua parte.

> **Note**
>
> A partir do Apache Kafka versão 3.7.x no MSK, você pode criar clusters que usam o modo Kraft em vez do modo. ZooKeeper

- Produtores, consumidores e criadores de tópicos: o Amazon MSK permite que você use operações do plano de dados do Apache Kafka para criar tópicos, além de produzir e consumir dados.
- Operações de cluster Você pode usar o AWS Management Console, o AWS Command Line Interface (AWS CLI) ou as APIs no SDK para realizar operações no plano de controle. Por exemplo, você pode criar ou excluir um cluster do Amazon MSK, listar todos os clusters em uma conta, visualizar as propriedades de um cluster e atualizar o número e o tipo de agentes em um cluster.

O Amazon MSK detecta e se recupera automaticamente dos cenários de falha mais comuns para clusters, permitindo que as aplicações produtoras e consumidoras possam continuar suas operações de gravação e leitura com o menor impacto. Quando o Amazon MSK detecta uma falha de agente, ele mitiga a falha ou substitui o agente não íntegro ou inacessível por um novo. Além disso, sempre que possível, ele reutiliza o armazenamento do agente mais antigo para reduzir os dados que o Apache Kafka precisa replicar. Seu impacto na disponibilidade é limitado ao tempo necessário para o Amazon MSK concluir a detecção e a recuperação. Após uma recuperação, os aplicativos de produtor e consumidor podem continuar se comunicando com os mesmos endereços IP do agente usados antes da falha.

Configuração do Amazon MSK

Antes de usar o Amazon MSK pela primeira vez, conclua as seguintes tarefas.

Tarefas

- Inscreva-se para AWS
- Fazer download de bibliotecas e ferramentas

Inscreva-se para AWS

Quando você se inscreve AWS, sua conta da Amazon Web Services é automaticamente cadastrada em todos os serviços AWS, incluindo o Amazon MSK. Você será cobrado apenas pelos serviços que usar.

Se você já tiver uma AWS conta, vá para a próxima tarefa. Se você ainda não possuir uma conta da AWS , use o procedimento a seguir para criar uma.

Para cadastrar uma conta da Amazon Web Services

1. Abra https://portal.aws.amazon.com/billing/signup.
2. Siga as instruções on-line.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica e digitar um código de verificação no teclado do telefone.

   Quando você se inscreve em um Conta da AWS, um Usuário raiz da conta da AWSé criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como uma prática recomendada de segurança, atribua o acesso administrativo para um usuário e use somente o usuário-raiz para executar tarefas que requerem o acesso de usuário-raiz.

Fazer download de bibliotecas e ferramentas

As seguintes bibliotecas e ferramentas podem ajudar você a trabalhar com o Amazon MSK::

- A AWS Command Line Interface (AWS CLI) é compatível com o Amazon MSK. AWS CLI Isso permite que você controle vários Amazon Web Services a partir da linha de comando e os automatize por meio de scripts. Atualize sua versão AWS CLI para a versão mais recente para

garantir que ela tenha suporte aos recursos do Amazon MSK que estão documentados neste guia do usuário. Para obter instruções detalhadas sobre como atualizar a AWS CLI, consulte Como instalar a AWS Command Line Interface. Depois de instalar o AWS CLI, você deve configurá-lo. Para obter informações sobre como configurar o AWS CLI, consulte aws configure.

- A Referência de API do Amazon Managed Streaming for Kafka documenta as operações de API compatíveis com o Amazon MSK.
- Os SDKs da Amazon Web Services para Go, Java, .NET JavaScript, Node.js, PHP, Python e Ruby incluem suporte e amostras do Amazon MSK.

Conceitos básicos sobre como usar o Amazon MSK

Este tutorial mostra um exemplo de como criar um cluster do MSK, produzir e consumir dados e monitorar a integridade do seu cluster usando métricas. Este exemplo não representa todas as opções que você pode escolher ao criar um cluster do MSK. Em diferentes partes deste tutorial, escolhemos as opções padrão para facilitar. Isso não significa que estas são as únicas opções disponíveis para configurar um cluster do MSK ou instâncias de cliente.

Tópicos

Etapa 1: criar um cluster do Amazon MSK

Nesta etapa de Conceitos básicos sobre como usar o Amazon MSK, você vai criar um cluster do Amazon MSK.

Para criar um cluster Amazon MSK usando o AWS Management Console

1. Faça login no AWS Management Console e abra o console do Amazon MSK em https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Selecione Criar cluster.
3. Em Método de criação, deixe a opção Criação rápida selecionada. A opção Criação rápida permite criar um cluster com as configurações padrão.
4. Em Nome do cluster, insira um nome descritivo para o cluster. Por exemplo, **`MSKTutorialCluster`**.
5. Em Propriedades gerais do cluster, escolha Provisionado como o Tipo de cluster.
6. Na tabela em Todas as configurações de cluster, copie e salve os valores das configurações a seguir, pois você precisará deles posteriormente neste tutorial:

- VPC
- Subredes
- Grupos de segurança associados à VPC

7. Selecione Criar cluster.
8. Verifique o Status do cluster na página Resumo do cluster. O status muda de Criando para Ativo conforme o Amazon MSK provisiona o cluster. Quando o status estiver Ativo, você poderá se conectar ao cluster. Para obter mais informações sobre status de cluster, consulte Estados de cluster.

Próxima etapa

Etapa 2: Criar uma função do IAM

Etapa 2: Criar uma função do IAM

Nesta etapa, você executará duas tarefas. A primeira tarefa será a criação de uma política do IAM que conceda acesso para criar tópicos no cluster e enviar dados para esses tópicos. A segunda tarefa será a criação de um perfil do IAM e a associação dessa política a ele. Em uma etapa posterior, você criará uma máquina cliente que vai assumir esse perfil e usá-lo para criar um tópico no cluster e enviar dados para esse tópico.

Para criar uma política do IAM que permita criar tópicos e gravar neles

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Políticas.
3. Escolha Create Policy.
4. Escolha a guia JSON e substitua o JSON na janela do editor com o JSON a seguir.

   Substitua a *região* pelo código da AWS região em que você criou seu cluster. Substitua *Account-ID* pelo seu ID de conta. Substitua *MSK TutorialCluster* pelo nome do seu cluster.

```
{
    "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/*"
            ]
        }
    ]
}
```

Para obter instruções sobre como criar políticas de seguras, consulte the section called “Controle de acesso do IAM”.

5. Escolha Próximo: etiquetas.
6. Selecione Next: Review (Próximo: revisar).
7. Para o nome da política, insira um nome descritivo, como msk-tutorial-policy.
8. Escolha Criar política.

Para criar um perfil do IAM e associar a política a ele

1. No painel de navegação, escolha Perfis.
2. Selecione Criar função.
3. Em Casos de uso comuns, selecione EC2 e então Próximo: permissões.
4. Na caixa de pesquisa, insira o nome da política que você criou anteriormente para este tutorial. Em seguida, marque a caixa à esquerda da política.
5. Escolha Próximo: etiquetas.
6. Selecione Next: Review (Próximo: revisar).
7. Para o nome da política, insira um nome descritivo, como msk-tutorial-role.
8. Selecione Criar função.

Próxima etapa

Etapa 3: criar uma máquina cliente

Etapa 3: criar uma máquina cliente

Nesta etapa de Conceitos básicos sobre como usar o Amazon MSK, você criará uma máquina cliente. Use essa máquina cliente para criar um tópico que produza e consuma dados. Para simplificar, você criará essa máquina cliente na VPC associada ao cluster do MSK para que o cliente possa se conectar facilmente ao cluster.

Como criar uma máquina cliente

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Selecione Iniciar instâncias.
3. Insira um Nome para sua máquina cliente, como **`MSKTutorialClient`**.
4. Deixe a opção AMI do Amazon Linux 2 (HVM) – Kernel 5.10, tipo de volume SSD selecionada para Tipo de imagem de máquina da Amazon (AMI).
5. Deixe o tipo de instância t2.micro selecionado.
6. Na seção Par de chaves, escolha Criar um novo par de chaves. Digite **`MSKKeyPair`** em Nome do par de chaves e, em seguida, escolha Baixar par de chaves. Se preferir, use um par de chaves existente.

7. Expanda a seção Detalhes avançados e escolha o perfil do IAM que você criou na Etapa 2: criar um perfil do IAM.
8. Escolha Iniciar instância.
9. Escolha View Instances (Exibir instâncias). Na coluna Grupos de segurança, escolha o grupo de segurança que está associado à sua nova instância. Copie o ID do grupo de segurança e salve-o para usar posteriormente.
10. Abra o console da Amazon VPC em https://console.aws.amazon.com/vpc/.
11. No painel de navegação, selecione Security Groups.(Grupos de segurança). Encontre o grupo de segurança cujo ID você salvou em the section called “Etapa 1: criar um cluster”.
12. Na guia Regras de entrada, selecione Editar regras de entrada.
13. Escolha Adicionar regra.
14. Na nova regra, escolha All traffic (Todo o tráfego) na coluna Type (Tipo). No segundo campo da coluna Origem, selecione o grupo de segurança da sua máquina cliente. Esse é o grupo cujo nome você salvou após iniciar a instância da máquina cliente.
15. Escolha Salvar regras. Agora, o grupo de segurança do cluster poderá aceitar o tráfego proveniente do grupo de segurança da máquina cliente.

Próxima etapa

Etapa 4: criar um tópico

Etapa 4: criar um tópico

Nesta etapa de Conceitos básicos sobre como usar o Amazon MSK, você instalará bibliotecas e ferramentas do cliente do Apache Kafka na máquina cliente e criará um tópico.

> **Warning**
>
> Os números de versão do Apache Kafka usados neste tutorial são apenas exemplos. Recomendamos usar a mesma versão do cliente que a versão do cluster do MSK. Uma versão mais antiga do cliente pode não ter determinados recursos e correções de erros críticos.

Para encontrar a versão do seu cluster do MSK

1. Acesse https://eu-west-2.console.aws.amazon.com/msk/
2. Selecione o cluster do MSK.
3. Anote a versão do Apache Kafka usada no cluster.
4. Substitua as ocorrências de números de versão do Amazon MSK neste tutorial pela versão obtida na Etapa 3.

Como criar um tópico na máquina cliente

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. No painel de navegação, escolha Instâncias. Em seguida, marque a caixa de seleção ao lado do nome da máquina cliente que você criou em Etapa 3: criar uma máquina cliente.
3. Escolha Actions (Ações) e Connect (Conectar-se). Siga as instruções no console para se conectar à sua máquina cliente.
4. Instale o Java na máquina cliente executando o seguinte comando:

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

5. Execute o comando a seguir para fazer download do Apache Kafka.

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

> **Note**
>
> Se quiser usar um local de espelhamento diferente do usado neste comando, você poderá escolher um local diferente no site do Apache.

6. Execute o comando a seguir no diretório onde você fez download do arquivo TAR na etapa anterior.

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

7. Acesse o diretório `kafka_2.13-{YOUR MSK VERSION}/libs` e execute o seguinte comando para baixar o arquivo JAR do IAM do Amazon MSK. O JAR do IAM do Amazon MSK permite que a máquina cliente acesse o 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. Acesse o diretório `kafka_2.13-{YOUR MSK VERSION}/bin`. Copie e cole as seguintes configurações de propriedade em um novo arquivo. Nomeie e salve o arquivo como **`client.properties`**.

```
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. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
10. Aguarde até que o status do seu cluster esteja Ativo. Isso pode demorar vários minutos. Depois que o status ficar Ativo, escolha o nome do cluster. Isso levará você a uma página com o resumo do cluster.
11. Escolha Exibir informações do cliente.
12. Copie a string de conexão para o endpoint privado.

    Você receberá três endpoints para cada um dos agentes. Só é necessário ter um endpoint de agente para a próxima etapa.
13. Execute o comando a seguir, substituindo *`BootstrapServerString`* por um dos endpoints do broker que você obteve na etapa anterior.

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

    Se o comando tiver êxito, a seguinte mensagem será exibida: `Created topic MSKTutorialTopic.`

Próxima etapa

Etapa 5: produzir e consumir dados

Etapa 5: produzir e consumir dados

Nesta etapa de Conceitos básicos sobre como usar o Amazon MSK, você produzirá e consumirá dados.

Como produzir e consumir mensagens

1. Execute o comando a seguir para iniciar um produtor de console. Substitua String pela *BootstrapServerstring* de conexão em texto simples que você obteve em Criar um tópico. Para obter instruções sobre como recuperar essa string de conexão, consulte Como obter os agentes de bootstrap para um cluster do Amazon MSK.

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

2. Insira a mensagem que desejar e pressione Enter. Repita esta etapa duas ou três vezes. Toda vez que você inserir uma linha e pressionar Enter, essa linha será enviada para o cluster do Apache Kafka como uma mensagem separada.
3. Mantenha a conexão com a máquina cliente aberta e abra uma segunda conexão separada com esse computador em uma nova janela.
4. No comando a seguir, substitua *BootstrapServerString pela string* de conexão em texto simples que você salvou anteriormente. Em seguida, para criar um consumidor no console, execute o comando a seguir com sua segunda conexão com a máquina cliente.

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

   Você começará a ver as mensagens inseridas anteriormente quando usou o comando do produtor do console.
5. Insira mais mensagens na janela do produtor e observe-as aparecerem na janela do consumidor.

Próxima etapa

Etapa 6: Use CloudWatch a Amazon para visualizar as métricas do Amazon MSK

Etapa 6: Use CloudWatch a Amazon para visualizar as métricas do Amazon MSK

Nesta etapa de Introdução ao uso do Amazon MSK, você analisa as métricas do Amazon MSK na Amazon. CloudWatch

Para visualizar as métricas do Amazon MSK em CloudWatch

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, selecione Métricas.
3. Escolha a guia All metrics (Todas as métricas) e escolha AWS/Kafka.
4. Para visualizar métricas no nível de agente, escolha Broker ID, Cluster Name (ID do agente, Nome do cluster). Para métricas no nível de cluster, escolha Cluster Name (Nome do cluster).
5. (Opcional) No painel gráfico, selecione uma estatística e um período de tempo e, em seguida, crie um CloudWatch alarme usando essas configurações.

Próxima etapa

Etapa 7: Excluir os AWS recursos criados para este tutorial

Etapa 7: Excluir os AWS recursos criados para este tutorial

Na etapa final de Conceitos básicos sobre como usar o Amazon MSK, você exclui o cluster do MSK e a máquina cliente que criou para este tutorial.

Para excluir os recursos usando o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. Selecione o nome do seu cluster. Por exemplo, MSK TutorialCluster.
3. Escolha Ações e Excluir.
4. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
5. Escolha a instância que você criou para sua máquina cliente, por exemplo, **`MSKTutorialClient`**.
6. Escolha Estado da instância e Encerrar instância.

Para excluir a política e o perfil do IAM

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Perfis.
3. Na caixa de pesquisa, insira o nome do perfil do IAM que você criou para este tutorial.
4. Selecione o perfil de . Escolha Excluir perfil e confirme a exclusão.
5. No painel de navegação, escolha Políticas.
6. Na caixa de pesquisa, insira o nome da política que você criou para este tutorial.
7. Escolha a política para abrir a respectiva página de resumo. Na página Resumo da política, escolha Editar política.
8. Escolha Excluir.

Amazon MSK: funcionamento

Um cluster do Amazon MSK é o recurso primário do Amazon MSK que você pode criar em sua conta. Os tópicos desta seção descrevem como realizar operações comuns do Amazon MSK. Para obter uma lista de todas as operações que você pode realizar em um cluster do MSK, consulte:

- A AWS Management Console
- A Referência de API do Amazon MSK
- A Referência de comandos da CLI do Amazon MSK

Tópicos

Como criar um cluster do Amazon MSK

> **Important**
>
> Não é possível alterar a VPC de um cluster do Amazon MSK depois de criar o cluster.

Antes de criar um cluster do Amazon MSK, você precisa ter uma Amazon Virtual Private Cloud (VPC) e configurar sub-redes nessa VPC.

Você precisa de duas sub-redes em duas zonas de disponibilidade diferentes na região Oeste dos EUA (Norte da Califórnia). Em todas as outras regiões que disponibilizam o Amazon MSK, é possível especificar duas ou três sub-redes. As suas sub-redes devem estar em diferentes zonas de disponibilidade. Quando você cria um cluster, o Amazon MSK distribui os nós de agente uniformemente pelas sub-redes especificadas.

Tamanhos de corretores

Ao criar um cluster Amazon MSK, você especifica o tamanho dos corretores que deseja que ele tenha. O Amazon MSK oferece suporte aos seguintes tamanhos de corretores:

- 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

Os corretores M7g usam processadores AWS Graviton (processadores personalizados baseados em ARM criados pela Amazon Web Services). Os corretores M7g oferecem melhor desempenho de preço em relação a instâncias M5 comparáveis. Os corretores M7g consomem menos energia do que instâncias M5 comparáveis.

Os corretores M7g Graviton não estão disponíveis nas seguintes regiões: CDG (Paris), CGK (Jacarta), CPT (Cidade do Cabo), DXB (Dubai), HKG (Hong Kong), KIX (Osaka), LHR (Londres), MEL (Melbourne), MXP (Milão), OSU (Leste dos EUA), PDT (Oeste dos EUA), TLV (Tel Aviv), YY YC (Calgary), ZRH (Zurique).

O MSK oferece suporte a corretores M7g em clusters que executam uma das seguintes versões do Kafka:

- 2.8.2. em camadas
- 3.3.2
- 3.4.0
- 3.5.1
- 3.6.0 com armazenamento em camadas

- 3.7.x
- 3.7.x. Kraft

Os corretores M7g e M5 têm maior desempenho de taxa de transferência de linha de base do que os corretores T3 e são recomendados para cargas de trabalho de produção. Os corretores M7g e M5 também podem ter mais partições por corretor do que os corretores T3. Use corretores M7g ou M5 se você estiver executando cargas de trabalho maiores de nível de produção ou precisar de um número maior de partições. Para saber mais sobre os tamanhos de instância M7g e M5, consulte Instâncias de uso geral do Amazon EC2.

Os agentes T3 têm a capacidade de usar créditos de CPU para impulsionar temporariamente o desempenho. Use agentes T3 para desenvolvimento de baixo custo, se você estiver testando cargas de trabalho de streaming pequenas a médias ou se tiver cargas de trabalho de streaming com baixo throughput que apresentem picos temporários no throughput. Recomendamos que você faça um proof-of-concept teste para determinar se os corretores T3 são suficientes para produção ou carga de trabalho crítica. Para saber mais sobre os tamanhos de corretores T3, consulte Instâncias EC2/T3 da Amazon.

Para obter mais informações sobre como escolher os tamanhos dos corretores, consultePráticas recomendadas.

Criando um cluster usando o AWS Management Console

Esse processo descreve a tarefa comum de criar um cluster provisionado usando opções de criação personalizadas. Você pode selecionar outras opções no console MSK para criar um cluster sem servidor.

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. Selecione Criar cluster.
3. Em Método de criação de cluster, escolha Criação personalizada.
4. Especifique um nome de cluster que seja exclusivo e não tenha mais de 64 caracteres.
5. Em Tipo de cluster, escolha Provisionado, que permite especificar o número de agentes, o tamanho do agente e a capacidade de armazenamento do cluster.
6. Selecione a versão do Apache Kafka que você deseja executar nos corretores. Para ver uma comparação dos recursos do MSK que são compatíveis com cada versão do Apache Kafka, selecione Exibir compatibilidade da versão.

7. Dependendo da versão do Apache Kafka selecionada, você pode ter a opção de escolher o modo de metadados do cluster: ou Kraft. ZooKeeper
8. Selecione um tamanho de agente para usar no cluster com base nas necessidades de computação, memória e armazenamento do cluster. Consulte ???.
9. Selecione o número de zonas nas quais os corretores são distribuídos.
10. Especifique o número de corretores que você deseja que o MSK crie em cada zona de disponibilidade. O mínimo é um agente por zona de disponibilidade e o máximo é 30 corretores por cluster para clusters ZooKeeper baseados e 60 corretores por cluster para clusters baseados em Kraft.
11. Selecione a quantidade inicial de armazenamento que você deseja que seu cluster tenha. Você não pode diminuir a capacidade de armazenamento depois de criar o cluster.
12. Dependendo do tamanho do agente (tamanho da instância) selecionado, você pode especificar a taxa de transferência de armazenamento provisionado por agente. Para ativar essa opção, escolha o tamanho do broker (tamanho da instância) kafka.m5.4xlarge ou maior para x86 e kafka.m7g.2xlarge ou maior para instâncias baseadas em Graviton. Consulte ???.
13. Selecione uma opção de modo de armazenamento em cluster, somente armazenamento EBS ou armazenamento em camadas e armazenamento EBS.
14. Se você quiser criar e usar uma configuração de cluster personalizada (ou se você já tiver uma configuração de cluster salva), escolha uma configuração. Caso contrário, você pode criar o cluster usando a configuração de cluster padrão do Amazon MSK. Para obter informações sobre configurações do Amazon MSK, consulte Configuração.
15. Escolha Próximo.
16. Para configurações de rede, escolha a VPC que você deseja usar para o cluster.
17. Com base no número de zonas que você selecionou anteriormente, especifique as zonas de disponibilidade e as sub-redes nas quais os corretores serão implantados. As duas sub-redes devem estar em zonas de disponibilidade diferentes.
18. Você pode selecionar um ou mais grupos de segurança aos quais deseja conceder acesso ao seu cluster (por exemplo, os grupos de segurança das máquinas clientes). Se você especificar grupos de segurança compartilhados com você, deverá garantir que tenha permissões para usá-los. Especificamente, você precisa da permissão `ec2:DescribeSecurityGroups`. Conectando-se a um cluster Amazon MSK.
19. Escolha Próximo.

20. Selecione os métodos de controle de acesso e as configurações de criptografia do cluster para criptografar dados à medida que eles transitam entre clientes e corretores. Para ter mais informações, consulte the section called “Criptografia em trânsito”.
21. Escolha o tipo de chave do KMS que deseja usar para criptografar dados em repouso. Para ter mais informações, consulte the section called “Criptografia em repouso”.
22. Escolha Próximo.
23. Escolha o monitoramento e as tags que você deseja. Isso determina o conjunto de métricas que você obtém. Para ter mais informações, consulte Como monitorar um cluster. Amazon CloudWatch, Prometheus, Broker log delivery ou Cluster tags e, em seguida, selecione Avançar.
24. Revise as configurações do seu cluster. Você pode voltar e alterar as configurações selecionando Anterior para voltar à tela anterior do console ou Editar para alterar as configurações específicas do cluster. Se as configurações estiverem corretas, selecione Criar cluster.
25. Verifique o Status do cluster na página Resumo do cluster. O status muda de Criando para Ativo conforme o Amazon MSK provisiona o cluster. Quando o status estiver Ativo, você poderá se conectar ao cluster. Para obter mais informações sobre status de cluster, consulte Estados de cluster.

Criando um cluster usando o AWS CLI

1. Copie o seguinte JSON e salve-o em um arquivo. Nomeie o arquivo `brokernodegroupinfo.json`. Substitua os IDs de sub-rede no JSON pelos valores que correspondem às suas sub-redes. As sub-redes devem estar em zonas de disponibilidade diferentes. Substitua *“Security-Group-ID”* pelo ID de um ou mais grupos de segurança da VPC cliente. Os clientes associados a esses grupos de segurança têm acesso ao cluster. Se você especificar grupos de segurança que foram compartilhados com você, deverá garantir que você tenha permissões para eles. Especificamente, você precisa da permissão `ec2:DescribeSecurityGroups`. Para obter um exemplo, consulte Amazon EC2: permite o gerenciamento de grupos de segurança do Amazon EC2 associados a uma VPC específica de maneira programática e no console. Por fim, salve o arquivo JSON atualizado no computador em que você o AWS CLI instalou.

```
{
  "InstanceType": "kafka.m5.large",
  "ClientSubnets": [
```

```
    "Subnet-1-ID",
    "Subnet-2-ID"
  ],
  "SecurityGroups": [
    "Security-Group-ID"
  ]
}
```

> **Important**
>
> Especifique exatamente duas sub-redes se estiver usando a região Oeste dos EUA (Norte da Califórnia). Para outras regiões em que o Amazon MSK esteja disponível, especifique duas ou três sub-redes. As sub-redes especificadas devem estar em zonas de disponibilidade distintas. Quando você cria um cluster, o Amazon MSK distribui os nós de agente uniformemente pelas sub-redes especificadas.

2. Execute o AWS CLI comando a seguir no diretório em que você salvou o `brokernodegroupinfo.json` arquivo, substituindo *"Nome do seu cluster" por um nome de* sua escolha. Para *"Monitoring-Level"*, você pode especificar um dos três valores a seguir: `DEFAULT`, `PER_BROKER` ou `PER_TOPIC_PER_BROKER`. Para obter informações sobre esses três níveis diferentes de monitoramento, consulte ???. O parâmetro `enhanced-monitoring` é opcional. Se não especificá-lo no comando `create-cluster`, você obterá o nível de monitoramento `DEFAULT`.

```
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"
```

A saída do comando é semelhante ao JSON a seguir:

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

> **Note**
>
> O comando `create-cluster` pode retornar um erro informando que uma ou mais sub-redes pertencem a zonas de disponibilidade que não têm suporte. Quando isso acontece, o erro indica as zonas de disponibilidade que não têm suporte. Crie sub-redes que não usem as zonas de disponibilidade sem suporte e tente o comando `create-cluster` novamente.

3. Salve o valor da chave `ClusterArn` porque você precisará dele para executar outras ações no cluster.
4. Execute o seguinte comando para verificar o `STATE` do seu cluster. O valor de `STATE` muda de `CREATING` para `ACTIVE` conforme o Amazon MSK provisiona o cluster. Quando o estado for `ACTIVE`, você poderá se conectar ao cluster. Para obter mais informações sobre status de cluster, consulte Estados de cluster.

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

Criação de um cluster com uma configuração personalizada do Amazon MSK usando o AWS CLI

Para obter informações sobre configurações personalizadas do Amazon MSK e como criá-las, consulte Configuração.

1. Salve o seguinte JSON em um arquivo, substituindo *`configuration-arn`* pelo ARN da configuração que você deseja usar para criar o cluster.

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

2. Execute o comando `create-cluster` e use a opção `configuration-info` para apontar para o arquivo JSON que você salvou na etapa anterior. Veja um exemplo a seguir.

```
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
```

Veja a seguir um exemplo de uma resposta bem-sucedida após a execução desse comando.

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

Como criar um cluster usando a API

Para criar um cluster usando a API, consulte CreateCluster.

Como excluir um cluster do Amazon MSK

> **Note**
>
> Se seu cluster tiver uma política de ajuste de escala automático, recomendamos que você remova a política antes de excluir o cluster. Para ter mais informações, consulte Escalabilidade automática.

Excluindo um cluster usando o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. Escolha o cluster do MSK que deseja excluir marcando a caixa de seleção ao lado dele.
3. Escolha Excluir e confirme a exclusão.

Excluindo um cluster usando o AWS CLI

Execute o comando a seguir, *ClusterArn*substituindo-o pelo Amazon Resource Name (ARN) que você obteve ao criar seu cluster. Se você não tiver o ARN do cluster, poderá encontrá-lo listando todos os clusters. Para ter mais informações, consulte the section called “Listar clusters”.

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

Excluir um cluster usando a API

Para excluir um cluster usando a API, consulte DeleteCluster.

Como obter agentes de bootstrap para um cluster do Amazon MSK

Obtendo os corretores de bootstrap usando o AWS Management Console

O termo agentes de bootstrap se refere a uma lista de agentes que um cliente Apache Kafka pode usar como ponto de partida para se conectar ao cluster. Essa lista não inclui necessariamente todos os agentes em um cluster.

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. A tabela mostra todos os clusters da região atual nesta conta. Escolha o nome de um cluster para visualizar sua descrição.
3. Na página Resumo do cluster, escolha Exibir informações do cliente. Isso mostra os corretores de bootstrap, bem como a string de conexão do Apache ZooKeeper .

Obtendo os corretores de bootstrap usando o AWS CLI

Execute o comando a seguir, *ClusterArn*substituindo-o pelo Amazon Resource Name (ARN) que você obteve ao criar seu cluster. Se você não tiver o ARN do cluster, poderá encontrá-lo listando todos os clusters. Para ter mais informações, consulte the section called “Listar clusters”.

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

Para um cluster do MSK que use o the section called “Controle de acesso do IAM”, a saída desse comando é semelhante ao seguinte exemplo de JSON.

```
{
    "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"
}
```

O exemplo a seguir mostra os agentes de bootstrap de um cluster com acesso público ativado. Use o `BootstrapBrokerStringPublicSaslIam` para acesso público e a `BootstrapBrokerStringSaslIam` string para acesso interno 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"
}
```

A string de agentes de bootstrap deve conter três agentes de todas as zonas de disponibilidade nas quais seu cluster do MSK esteja implantado (a menos que haja apenas dois agentes disponíveis).

Como obter os agentes de bootstrap usando a API

Para obter os corretores de bootstrap usando a API, consulte GetBootstrap Corretores.

Listar clusters do Amazon MSK

Listando clusters usando o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. A tabela mostra todos os clusters da região atual nesta conta. Escolha o nome de um cluster para visualizar seus detalhes.

Listando clusters usando o AWS CLI

Execute o seguinte comando .

```
aws kafka list-clusters
```

Listar clusters usando a API

Para listar clusters usando a API, consulte ListClusters.

Gerenciamento de metadados

O Amazon MSK oferece suporte aos modos de gerenciamento de metadados Apache ZooKeeper ou Kraft.

A partir do Apache Kafka versão 3.7.x no Amazon MSK, você pode criar clusters que usam o modo Kraft em vez do modo. ZooKeeper Os clusters baseados em Kraft dependem de controladores dentro do Kafka para gerenciar metadados.

Tópicos

ZooKeeper modo

O Apache ZooKeeper é “um serviço centralizado para manter informações de configuração, nomear, fornecer sincronização distribuída e fornecer serviços de grupo. Todos esses tipos de serviços são usados de uma forma ou de outra por aplicativos distribuídos”, incluindo o Apache Kafka.

Se seu cluster estiver usando o ZooKeeper modo, você pode usar as etapas abaixo para obter a string de ZooKeeper conexão do Apache. No entanto, recomendamos que você use o `BootstrapServerString` para se conectar ao seu cluster e realizar operações administrativas, pois o `--zookeeper` sinalizador foi descontinuado no Kafka 2.5 e foi removido do Kafka 3.0.

Obtendo a string de ZooKeeper conexão do Apache usando o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. A tabela mostra todos os clusters da região atual nesta conta. Escolha o nome de um cluster para visualizar sua descrição.
3. Na página Resumo do cluster, escolha Exibir informações do cliente. Isso mostra os corretores de bootstrap, bem como a string de conexão do Apache ZooKeeper .

Obtendo a string de ZooKeeper conexão do Apache usando o AWS CLI

1. Se não souber o nome de recurso da Amazon (ARN) do cluster, você poderá encontrá-lo listando todos os clusters em sua conta. Para ter mais informações, consulte the section called “Listar clusters”.
2. Para obter a cadeia de ZooKeeper conexão do Apache, junto com outras informações sobre seu cluster, execute o comando a seguir, *`ClusterArn`*substituindo-o pelo ARN do seu cluster.

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

A saída desse comando `describe-cluster` é semelhante ao seguinte JSON de exemplo.

```
{
    "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"
    }
}
```

O JSON de exemplo anterior mostra a chave `ZookeeperConnectString` na saída do comando `describe-cluster`. Copie o valor correspondente a essa chave e salve-o para quando precisar criar um tópico no cluster.

> **Important**
>
> Seu cluster Amazon MSK deve estar no `ACTIVE` estado para que você possa obter a cadeia de ZooKeeper conexão Apache. Quando um cluster ainda está no estado `CREATING`, a saída do comando `describe-cluster` não inclui a `ZookeeperConnectString`. Se esse for o caso, aguarde alguns minutos e execute `describe-cluster` novamente após o cluster atingir o estado `ACTIVE`.

Obtendo a string de ZooKeeper conexão do Apache usando a API

Para obter a string de ZooKeeper conexão do Apache usando a API, consulte DescribeCluster.

Modo Kraft

O Amazon MSK introduziu o suporte para o Kraft (Apache Kafka Raft) na versão 3.7.x do Kafka. A comunidade Apache Kafka desenvolveu o Kraft para substituir o Apache no gerenciamento de metadados nos clusters do Apache ZooKeeper Kafka. No modo Kraft, os metadados do cluster são propagados dentro de um grupo de controladores Kafka, que fazem parte do cluster Kafka, em vez de entre nós. ZooKeeper Os controladores Kraft estão incluídos sem custo adicional para você e não exigem configuração ou gerenciamento adicionais de sua parte. Consulte KIP-500 para obter mais informações sobre o Kraft.

Aqui estão alguns pontos a serem observados sobre o modo Kraft no MSK:

- O modo Kraft só está disponível para novos clusters. Você não pode alternar os modos de metadados depois que o cluster é criado.

- No console MSK, você pode criar um cluster baseado em Kraft escolhendo a versão 3.7.x do Kafka e marcando a caixa de seleção Kraft na janela de criação do cluster.
- Para criar um cluster no modo Kraft usando a API `CreateCluster`ou `CreateClusterV2`as operações do MSK, você deve usar `3.7.x.kraft` como versão. Use `3.7.x` como versão para criar um cluster no ZooKeeper modo.
- O número de partições por corretor é o mesmo no Kraft e nos clusters ZooKeeper baseados. No entanto, o Kraft permite que você hospede mais partições por cluster provisionando mais corretores em um cluster.
- Não são necessárias alterações de API para usar o modo Kraft no Amazon MSK. No entanto, se seus clientes ainda usam a cadeia de `--zookeeper` conexão hoje, você deve atualizá-los para usar a cadeia de `--bootstrap-server` conexão para se conectar ao seu cluster. A `--zookeeper` bandeira está obsoleta na versão 2.5 do Apache Kafka e foi removida a partir da versão 3.0 do Kafka. Portanto, recomendamos que você use as versões recentes do cliente Apache Kafka e a string de `--bootstrap-server` conexão para todas as conexões com seu cluster.
- ZooKeeper O modo continua disponível para todas as versões lançadas, nas quais o zookeeper também é suportado pelo Apache Kafka. Consulte Versões compatíveis do Apache Kafka para obter detalhes sobre o fim do suporte para versões do Apache Kafka e futuras atualizações.
- Você deve verificar se todas as ferramentas que você usa são capazes de usar as APIs de administração do Kafka sem conexões. ZooKeeper Consulte as etapas atualizadas Usando o LinkedIn Cruise Control para Apache Kafka com o Amazon MSK para conectar seu cluster ao Cruise Control. O Cruise Control também tem instruções para executar o Cruise Control sem ZooKeeper.
- Você não precisa acessar diretamente os controladores Kraft do seu cluster para nenhuma ação administrativa. No entanto, se você estiver usando o monitoramento aberto para coletar métricas, também precisará dos endpoints DNS dos seus controladores para coletar algumas métricas não relacionadas ao controlador sobre seu cluster. Você pode obter esses endpoints de DNS no console do MSK ou usando a operação da ListNodesAPI. Consulte as etapas atualizadas Monitoramento aberto com o Prometheus para configurar o monitoramento aberto para clusters baseados em Kraft.
- Não há CloudWatch métricas adicionais que você precise monitorar para clusters do modo Kraft em vez dos clusters do ZooKeeper modo. O MSK gerencia os controladores Kraft usados em seus clusters.

- Você pode continuar gerenciando ACLs usando clusters no modo Kraft usando a cadeia de `--bootstrap-server` conexão. Você não deve usar a cadeia de `--zookeeper` conexão para gerenciar ACLs. Consulte ACLs do Apache Kafka.
- No modo Kraft, os metadados do seu cluster são armazenados em controladores Kraft dentro do Kafka e não em nós externos. ZooKeeper Portanto, você não precisa controlar o acesso aos nós do controlador separadamente, como você faz com ZooKeeper os nós.

Gerenciamento de armazenamento

O Amazon MSK fornece recursos para ajudar você no gerenciamento do armazenamento em clusters do MSK.

Tópicos

- Armazenamento em camadas
- Como aumentar a escala verticalmente do armazenamento do agente
- Provisionar throughput de armazenamento

Armazenamento em camadas

O armazenamento em camadas é um nível de armazenamento de baixo custo para o Amazon MSK que se expande para armazenamento praticamente ilimitado, tornando econômica a criação de aplicações de streaming de dados.

Você pode criar um cluster Amazon MSK configurado com armazenamento em camadas que equilibra desempenho e custo. O Amazon MSK armazena dados de streaming no nível de armazenamento primário com desempenho otimizado até atingir os limites de retenção de tópico Apache Kafka. Em seguida, o Amazon MSK move automaticamente os dados para o novo nível de armazenamento de baixo custo.

Quando sua aplicação começa a ler dados do armazenamento em camadas, você pode esperar um aumento na latência de leitura nos primeiros bytes. Ao começar a ler os dados restantes sequencialmente do nível de baixo custo, você pode esperar latências semelhantes às do nível de armazenamento primário. Você não precisa provisionar nenhum armazenamento para o armazenamento em camadas de baixo custo nem gerenciar a infraestrutura. É possível armazenar qualquer quantidade de dados e pagar somente pelo que for usado. Esse recurso é compatível com as APIs introduzidas no KIP-405: armazenamento em camadas do Kafka.

Veja alguns dos recursos do armazenamento em camadas:

- Você pode escalar para armazenamento praticamente ilimitado. Você não precisa adivinhar como escalar sua infraestrutura do Apache Kafka.
- Você pode reter dados por mais tempo em seus tópicos do Apache Kafka ou aumentar seu armazenamento de tópicos, sem a necessidade de aumentar o número de agentes.
- Ele fornece um buffer de segurança de maior duração para lidar com atrasos inesperados no processamento.
- Você pode reprocessar dados antigos em sua ordem de produção exata com seu código de processamento de stream existente e as APIs do Kafka.
- As partições se reequilibram mais rapidamente porque os dados no armazenamento secundário não exigem replicação em discos intermediários.
- Os dados entre os agentes e o armazenamento em camadas se movem dentro da VPC e não trafegam pela Internet.
- Uma máquina cliente pode usar o mesmo processo para se conectar a novos clusters com armazenamento em camadas ativado, assim como para se conectar a um cluster sem o armazenamento em camadas ativado. Consulte Criar uma máquina cliente.

Requisitos de armazenamento em camadas

- Você deve usar a versão 3.0.0 ou superior do cliente Apache Kafka para criar um novo tópico com o armazenamento em camadas ativado. Para fazer a transição de um tópico existente para o armazenamento em camadas, você pode reconfigurar uma máquina cliente que use uma versão do cliente Kafka anterior à 3.0.0 (a versão mínima suportada do Apache Kafka é 2.8.2.) para habilitar o armazenamento em camadas. Consulte Etapa 4: criar um tópico.
- O cluster Amazon MSK com armazenamento em camadas ativado deve usar a versão 3.6.0 ou superior, ou 2.8.2.

Restrições e limitações do armazenamento em camadas

O armazenamento em camadas tem as seguintes restrições e limitações:

- O armazenamento em camadas é aplicado apenas aos clusters do modo provisionado.
- O armazenamento hierárquico não é compatível com o tamanho de corretor t3.small.

- O período mínimo de retenção em armazenamento de baixo custo é de 3 dias. Não há período mínimo de retenção para o armazenamento primário.
- O armazenamento em camadas não oferece suporte a vários diretórios de log em um agente (recursos relacionados ao JBOD).
- O armazenamento em camadas não oferece suporte a tópicos compactados. Certifique-se de que todos os tópicos com armazenamento em camadas ativado tenham a configuração de cleanup.policy somente para “EXCLUIR”.
- O armazenamento em camadas pode ser desabilitado para tópicos individuais, mas não para todo o cluster. Depois de desabilitado, o armazenamento em camadas não pode ser reabilitado para um tópico.
- Se você usa a versão 2.8.2 em camadas do Amazon MSK, você pode migrar somente para outra versão do Apache Kafka compatível com armazenamento em camadas. Se você não quiser continuar usando uma versão compatível com armazenamento em camadas, crie um novo cluster MSK e migre seus dados para ele.
- A kafka-log-dirs ferramenta não pode relatar o tamanho dos dados de armazenamento em camadas. A ferramenta relata somente o tamanho dos segmentos de log no armazenamento primário.

Como os segmentos de log são copiados para o armazenamento em camadas

Quando você habilita o armazenamento em camadas para um tópico novo ou existente, o Apache Kafka copia segmentos de log fechados do armazenamento primário para o armazenamento em camadas.

- O Apache Kafka copia somente segmentos de log fechados. Ele copia todas as mensagens do segmento de log para o armazenamento em camadas.
- Os segmentos ativos não estão qualificados para o armazenamento em camadas. O tamanho do segmento de log (segment.bytes) ou o tempo de rolagem do segmento (segment.ms) controla a taxa de fechamento do segmento e a taxa com a qual o Apache Kafka os copia para o armazenamento em camadas.

As configurações de retenção para um tópico com o armazenamento em camadas habilitado são diferentes das configurações para um tópico sem o armazenamento em camadas habilitado. As regras a seguir controlam a retenção de mensagens em tópicos com o armazenamento em camadas habilitado:

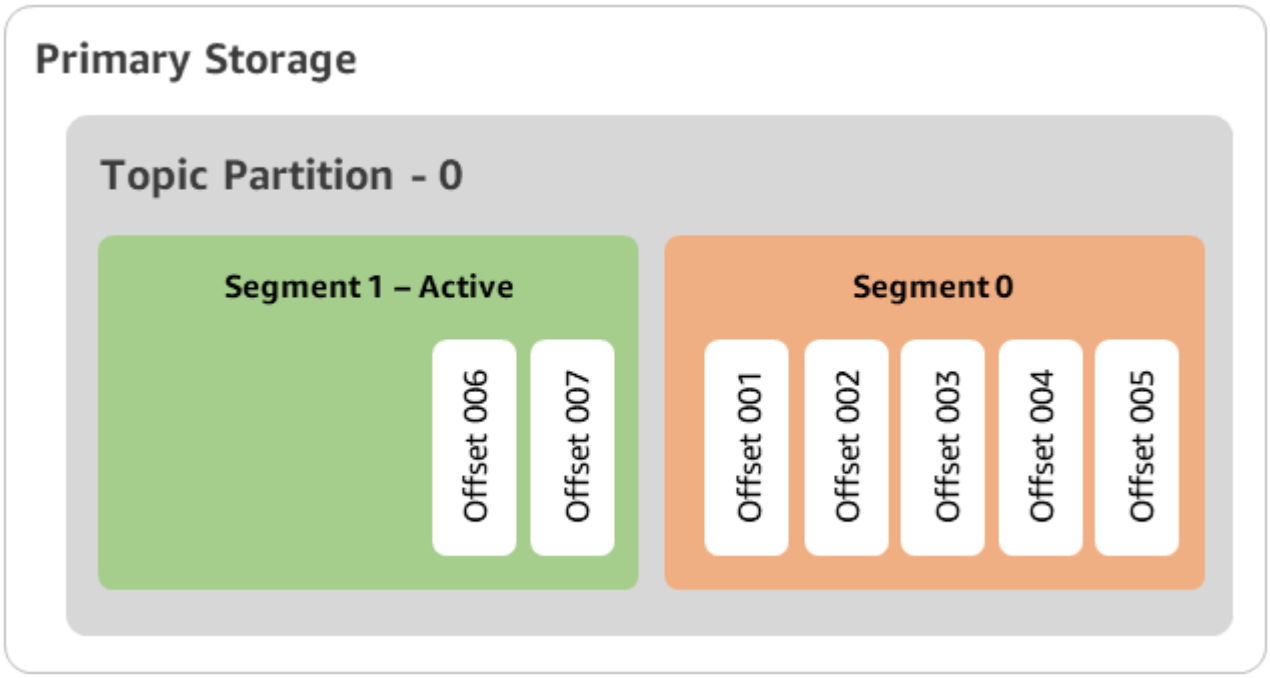
- Você define a retenção no Apache Kafka com duas configurações: log.retention.ms (tempo) e log.retention.bytes (tamanho). Essas configurações determinam a duração total e o tamanho dos dados que o Apache Kafka retém no cluster. Independentemente de você habilitar ou não o modo de armazenamento em camadas, defina essas configurações no nível do cluster. Você pode substituir as configurações no nível do tópico pelas configurações do tópico.
- Ao habilitar o armazenamento em camadas, você também pode especificar por quanto tempo o nível primário de armazenamento de alto desempenho armazena os dados. Por exemplo, se um tópico tiver uma configuração de retenção geral (log.retention.ms) de 7 dias e retenção local (local.retention.ms) de 12 horas, o armazenamento primário do cluster vai reter os dados somente nas primeiras 12 horas. O nível de armazenamento de baixo custo retém os dados por 7 dias completos.
- As configurações usuais de retenção se aplicam ao log completo. Isso inclui suas partes primárias e em camadas.
- As configurações local.retention.ms ou local.retention.bytes controlam a retenção de mensagens no armazenamento primário. Quando os dados atingem os limites de configuração de retenção do armazenamento primário (local.retention.ms/bytes) em um log completo, o Apache Kafka copia os dados do armazenamento primário para o armazenamento em camadas. Assim, os dados ficarão elegíveis para expiração.
- Quando o Apache Kafka copia uma mensagem em um segmento de log para o armazenamento em camadas, ele remove a mensagem do cluster com base nas configurações retention.ms ou retention.bytes.

Exemplo de cenário de armazenamento em camadas

Esse cenário ilustra como um tópico existente que tem mensagens no armazenamento primário se comporta quando o armazenamento em camadas está habilitado. Você habilita o armazenamento em camadas neste tópico ao definir remote.storage.enable como `true`. Neste exemplo, retention.ms está definido como 5 dias e local.retention.ms está definido como 2 dias. Veja a seguir a sequência de eventos quando um segmento expira.

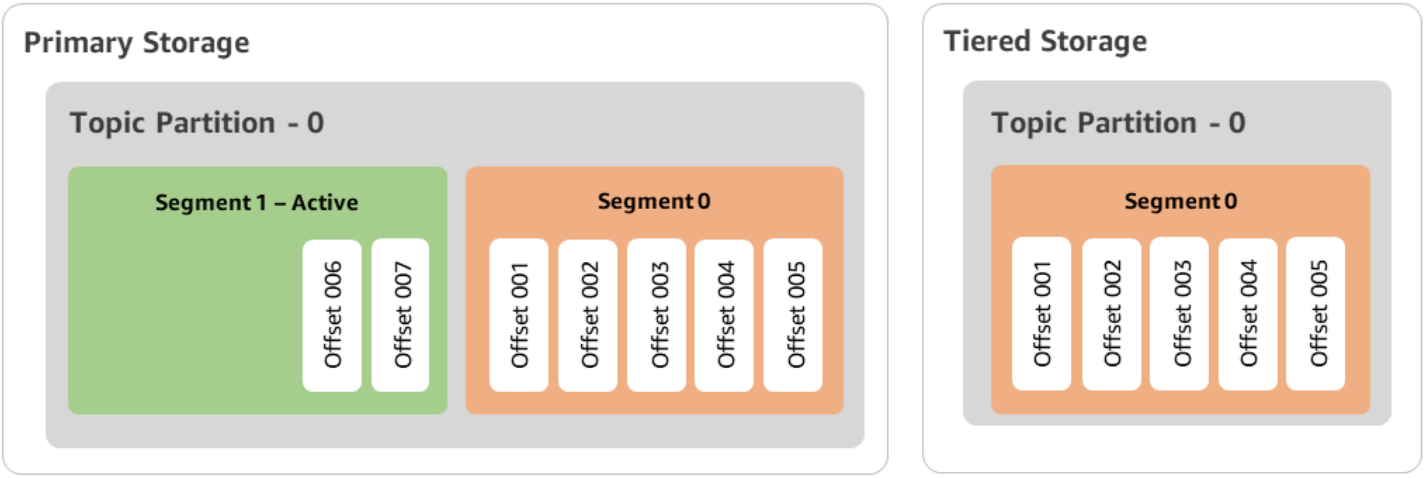
Tempo T0: antes de você habilitar o armazenamento em camadas.

Antes de você habilitar o armazenamento em camadas para este tópico, há dois segmentos de log. Um dos segmentos está ativo para uma partição 0 de tópico existente.



Tempo T1 (< 2 dias): armazenamento em camadas habilitado. Segmento 0 copiado para o armazenamento em camadas.

Após habilitar o armazenamento em camadas para esse tópico, o Apache Kafka copia o segmento 0 de log para o armazenamento em camadas depois que o segmento satisfizer as configurações iniciais de retenção. O Apache Kafka também vai reter a cópia de armazenamento principal do segmento 0. O segmento 1 ativo ainda não está qualificado para a cópia para o armazenamento em camadas. Neste cronograma, o Amazon MSK ainda não aplica nenhuma das configurações de retenção para nenhuma das mensagens no segmento 0 e no segmento 1. (local.retention.bytes/ms, retention.ms/bytes)



Tempo T2: retenção local em vigor.

Após 2 dias, as configurações de retenção primária entram em vigor para o segmento 0 que o Apache Kafka copiou para o armazenamento em camadas. A configuração de local.retention.ms como 2 dias determina isso. Agora, o segmento 0 expira do armazenamento primário. O segmento 1 ativo ainda não está qualificado para expiração nem está qualificado para a cópia para o armazenamento em camadas.

Tempo T3: retenção geral em vigor.

Após 5 dias, as configurações de retenção entram em vigor e o Kafka limpa o segmento 0 de log e as mensagens associadas do armazenamento em camadas. O segmento 1 ainda não está qualificado para expiração nem para cópia para armazenamento em camadas porque está ativo. O segmento 1 ainda não está fechado, portanto não é elegível para a rolagem de segmentos.

Criação de um cluster Amazon MSK com armazenamento hierárquico com o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. Selecione Criar cluster.
3. Escolha Criação personalizada para armazenamento em camadas.
4. Especifique um nome para o cluster.
5. No Tipo de cluster, selecione Provisionado.
6. Escolha uma versão do Amazon Kafka com suporte para armazenamento em camadas a fim de que o Amazon MSK a use para criar o cluster.
7. Especifique um tamanho de corretor diferente de kafka.t3.small.
8. Selecione o número de agentes que deseja que o Amazon MSK crie em cada zona de disponibilidade. O mínimo é de 1 agente por zona de disponibilidade e o máximo é de 30 agentes por cluster.
9. Especifique o número de zonas pelas quais os agentes estão distribuídos.
10. Especifique o número de agentes do Apache Kafka que estão implantados por zona.
11. Selecione Opções de armazenamento. Isso inclui Armazenamento em camadas e armazenamento do EBS para habilitar o modo de armazenamento em camadas.
12. Siga as etapas restantes no assistente de criação de cluster. Quando concluído, o Armazenamento em camadas e o armazenamento do EBS aparecerão como o modo de armazenamento de cluster na visualização Revisar e criar.
13. Selecione Create cluster (Criar cluster).

Criação de um cluster Amazon MSK com armazenamento hierárquico com o AWS CLI

Para habilitar o armazenamento em camadas em um cluster, crie o cluster com a versão correta do Apache Kafka e o atributo para armazenamento em camadas. Siga o exemplo de código abaixo. Além disso, conclua as etapas da próxima seção para Criação de um tópico do Kafka com o armazenamento em camadas habilitado.

Consulte create-cluster para obter uma lista completa dos atributos compatíveis com a criação de clusters.

```
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"
```

Criação de um tópico do Kafka com o armazenamento em camadas habilitado

Para concluir o processo iniciado ao criar um cluster com o armazenamento em camadas habilitado, crie também um tópico com o armazenamento em camadas habilitado com os atributos no exemplo de código adiante. Os atributos específicos para armazenamento em camadas são os seguintes:

- `local.retention.ms` (p. ex., 10 minutos) para configurações de retenção com base no tempo ou `local.retention.bytes` para limites de tamanho de segmentos de log.
- `remote.storage.enable` definido como `true` para habilitar o armazenamento em camadas.

A configuração a seguir usa local.retention.ms, mas você pode substituir esse atributo por local.retention.bytes. Esse atributo controla a quantidade de tempo que pode decorrer ou o número de bytes que o Apache Kafka pode copiar antes que o Apache Kafka copie os dados do armazenamento primário para o armazenamento em camadas. Consulte Configuração no nível de tópico para obter mais detalhes sobre os atributos de configuração compatíveis.

> **Note**
>
> Você deve usar o cliente Apache Kafka versão 3.0.0 ou superior. Essas versões são compatíveis com uma configuração chamada `remote.storage.enable` somente nas versões do cliente do `kafka-topics.sh`. Para habilitar o armazenamento em camadas em um tópico existente usando uma versão anterior do Apache Kafka, consulte a seção Habilitando o armazenamento em camadas em um tópico existente.

```
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
```

Habilitar e desabilitar o armazenamento em camadas em um tópico existente

Estas seções abordam como habilitar e desabilitar o armazenamento em camadas em um tópico que você já criou. Para criar um novo cluster e um tópico com o armazenamento em camadas habilitado, consulte Criação de um cluster com armazenamento em camadas usando o AWS Management Console.

Habilitando o armazenamento em camadas em um tópico existente

Para habilitar armazenamento em camadas em um tópico existente, use a sintaxe de comando `alter` no seguinte exemplo. Quando você habilita o armazenamento em camadas em um tópico existente, você não está restrito a uma determinada versão do cliente Apache Kafka.

```
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'
```

Desabilitar o armazenamento em camadas em um tópico existente

Para desabilitar o armazenamento em camadas em um tópico existente, use a sintaxe de comando `alter` na mesma ordem em que você habilita o armazenamento em camadas.

```
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**
>
> Ao desabilitar o armazenamento em camadas, você exclui completamente os dados do tópico no armazenamento em camadas. O Apache Kafka retém os dados do armazenamento primário, mas ainda aplica as regras de retenção primária com base em `local.retention.ms`. Após desabilitar o armazenamento em camadas em um tópico, não será possível habilitá-lo novamente. Se quiser desabilitar o armazenamento em camadas em um tópico existente, você não estará restrito a uma determinada versão do cliente Apache Kafka.

Habilitando o armazenamento em camadas em um cluster existente usando CLI AWS

> **Note**
>
> Você só pode habilitar o armazenamento em camadas se log.cleanup.policy do seu cluster estiver definido como `delete`, pois tópicos compactados não são compatíveis com o armazenamento em camadas. Posteriormente, você poderá configurar log.cleanup.policy de um tópico individual para `compact` se o armazenamento em camadas não estiver habilitado nesse tópico específico. Consulte Configuração no nível de tópico para obter mais detalhes sobre os atributos de configuração compatíveis.

1. Atualizar a versão do Kafka: as versões de cluster não são números inteiros simples. Para encontrar a versão atual do cluster, use a `DescribeCluster` operação ou o comando da `describe-cluster` AWS CLI. Uma versão de exemplo é `KTVPDKIKX0DER`.

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

2. Edite o modo de armazenamento do cluster. O exemplo de código a seguir mostra a edição do modo de armazenamento do cluster para `TIERED` usando a API `update-storage`.

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

Atualização do armazenamento em camadas em um cluster existente usando o console

> **Note**
>
> Você só pode habilitar o armazenamento em camadas se log.cleanup.policy do seu cluster estiver definido como `delete`, pois tópicos compactados não são compatíveis com o armazenamento em camadas. Posteriormente, você poderá configurar log.cleanup.policy de um tópico individual para `compact` se o armazenamento em camadas não estiver habilitado nesse tópico específico. Consulte Configuração no nível de tópico para obter mais detalhes sobre os atributos de configuração compatíveis.

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. Acesse a página de resumo do cluster e escolha Propriedades.
3. Acesse a seção Armazenamento e escolha Editar modo de armazenamento do cluster.
4. Escolha Armazenamento em camadas e armazenamento do EBS e Salvar as alterações.

Como aumentar a escala verticalmente do armazenamento do agente

É possível aumentar a quantidade de armazenamento do EBS por agente. Você não pode reduzir o armazenamento.

Os volumes de armazenamento permanecem disponíveis durante essa operação de expansão.

> **Important**
>
> Quando o armazenamento for escalado para um cluster do MSK, o armazenamento adicional será disponibilizado imediatamente. No entanto, o cluster requer um período de resfriamento após cada evento de escalabilidade de armazenamento. O Amazon MSK usa esse período de resfriamento para otimizar o cluster antes que ele possa ser escalado novamente. Dependendo do tamanho e da utilização do armazenamento do cluster e do tráfego, esse período pode variar de um mínimo de 6 horas a mais de 24 horas. Isso é aplicável tanto para eventos de escalonamento automático quanto para escalabilidade manual usando a operação UpdateBrokerde armazenamento. Para obter informações sobre como dimensionar corretamente seu armazenamento, consulte Práticas recomendadas.

Você pode usar o armazenamento em camadas para aumentar a escala verticalmente até quantidades ilimitadas de armazenamento para seu agente. Consulte Armazenamento em camadas.

Tópicos

- Escalabilidade automática
- Escalabilidade manual

Escalabilidade automática

Para expandir automaticamente o armazenamento do seu cluster em resposta ao aumento do uso, você pode configurar uma política de ajuste de escala automático de aplicações para o Amazon

MSK. Em uma política de ajuste de escala automático, você define a utilização do disco de destino e a capacidade máxima de escalabilidade.

Antes de usar a escalabilidade automática para o Amazon MSK, você deve avaliar o seguinte:

- **Important**

  Uma ação de escalabilidade de armazenamento só pode ocorrer uma vez a cada 6 horas.

  Recomendamos que você comece com um volume de armazenamento do tamanho certo para suas demandas de armazenamento. Para obter orientação sobre o dimensionamento correto do seu cluster, consulte Dimensione seu cluster adequadamente: número de agentes por cluster.
- O Amazon MSK não reduz o armazenamento em cluster em resposta à redução do uso. O Amazon MSK não é compatível com a redução do tamanho dos volumes de armazenamento. Se precisar reduzir o tamanho do armazenamento em cluster, você deverá migrar seu cluster existente para um cluster com armazenamento menor. Para obter informações sobre a migração de um cluster, consulte Migração.
- O Amazon MSK não é compatível com a redução automática da escala na horizontal nas regiões Ásia-Pacífico (Osaka) e África (Cidade do Cabo).
- Quando você associa uma política de auto-scaling ao seu cluster, o Amazon EC2 Auto Scaling cria automaticamente um alarme da Amazon para rastreamento de alvos. CloudWatch Se você excluir um cluster com uma política de auto-scaling, CloudWatch esse alarme persistirá. Para excluir o CloudWatch alarme, você deve remover uma política de auto-scaling de um cluster antes de excluir o cluster. Para saber mais sobre o monitoramento de destino, consulte Políticas de escalabilidade de monitoramento de destino para o Amazon EC2 Auto Scaling no Guia do usuário do Amazon EC2 Auto Scaling.

Detalhes da política de ajuste de escala automático

Sua política de ajuste de escala automático define a seguinte métrica predefinida para seu cluster:

- Meta de utilização de armazenamento: o limite de utilização de armazenamento usado pelo Amazon MSK para acionar uma operação de ajuste de escala automático. Você pode definir a meta de utilização entre 10% e 80% da capacidade de armazenamento atual. Recomendamos que você defina a meta de utilização do armazenamento entre 50% e 60%.

- Capacidade máxima de armazenamento: o limite máximo de escalabilidade que o Amazon MSK pode definir para o armazenamento do seu agente. Você pode definir a capacidade máxima de armazenamento em até 16 TiB por agente. Para ter mais informações, consulte Cota do Amazon MSK.

Quando o Amazon MSK detecta que sua métrica `Maximum Disk Utilization` é igual ou maior que a configuração `Storage Utilization Target`, ele aumenta sua capacidade de armazenamento em um valor igual ao maior de 2 números: 10 GiB ou 10% do armazenamento atual. Por exemplo, se você tiver 1.000 GiB, esse valor será de 100 GiB. O serviço verifica a utilização do armazenamento a cada minuto. Outras operações de escalabilidade continuam aumentando o armazenamento em uma quantidade igual ao maior de 2 números: 10 GiB ou 10% do armazenamento atual.

Para determinar se ocorreram operações de auto-escalonamento, use a operação. ListClusterOperations

Como configurar a escalabilidade automática para seu cluster do Amazon MSK

Você pode usar o console do Amazon MSK, a API do Amazon MSK ou implementar AWS CloudFormation a escalabilidade automática para armazenamento. CloudFormation o suporte está disponível por meio de Application Auto Scaling.

> **Note**
>
> Você não pode implementar o escalabilidade automática ao criar um cluster. Primeiro, você deve criar o cluster e, em seguida, criar e habilitar uma política de ajuste de escala automático para ele. No entanto, você pode criar a política enquanto o serviço Amazon MSK cria seu cluster.

Configurar a escalabilidade automática usando o AWS Management Console

1. Faça login no AWS Management Console e abra o console do Amazon MSK em https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Na lista de clusters, escolha seu cluster. Isso levará você a uma página com os detalhes sobre o cluster.
3. Na seção Ajuste de escala automático para armazenamento, escolha Configurar.

4. Crie e dê um nome a uma política de ajuste de escala automático. Especifique a meta de utilização do armazenamento, a capacidade máxima de armazenamento e a métrica de destino.
5. Selecione `Save changes`.

Quando você salvas e habilitar a nova política, ela ficará ativa para o cluster. Em seguida, o Amazon MSK expande o armazenamento do cluster quando a meta de utilização do armazenamento é atingida.

Configurar a escalabilidade automática usando a CLI

1. Use o RegisterScalableTargetcomando para registrar um destino de utilização de armazenamento.
2. Use o PutScalingPolicycomando para criar uma política de expansão automática.

Configurar a escalabilidade automática usando a API

1. Use a RegisterScalableTargetAPI para registrar uma meta de utilização de armazenamento.
2. Use a PutScalingPolicyAPI para criar uma política de expansão automática.

Escalabilidade manual

Para aumentar o armazenamento, aguarde que o cluster esteja no estado `ACTIVE`. O escalonamento de armazenamento tem um período de resfriamento de pelo menos 6 horas entre os eventos. Embora a operação disponibilize armazenamento adicional imediatamente, o serviço realiza otimizações em seu cluster que podem levar até 24 horas ou mais. A duração dessas otimizações é proporcional ao tamanho do seu armazenamento.

Ampliando o armazenamento do corretor usando o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. Escolha o cluster do MSK para o qual deseja atualizar o armazenamento do agente.
3. Na seção Armazenamento, escolha Editar.
4. Especifique o volume de armazenamento desejado. Só é possível aumentar a quantidade de armazenamento, não é possível reduzi-la.
5. Escolha Salvar alterações.

Ampliando o armazenamento do corretor usando o AWS CLI

Execute o comando a seguir, *ClusterArn*substituindo-o pelo Amazon Resource Name (ARN) que você obteve ao criar seu cluster. Se você não tiver o ARN do cluster, poderá encontrá-lo listando todos os clusters. Para ter mais informações, consulte the section called “Listar clusters”.

Substitua *Current-Cluster-Version* pela versão atual do cluster.

> **Important**
>
> As versões de cluster não são inteiros simples. Para encontrar a versão atual do cluster, use a DescribeClusteroperação ou o comando AWS CLI describe-cluster. Uma versão de exemplo é `KTVPDKIKX0DER`.

O parâmetro *Target-Volume-in-GiB* representa a quantidade de armazenamento que você deseja que cada agente tenha. Só é possível atualizar o armazenamento de todos os agentes. Não é possível especificar agentes individuais dos quais atualizar o armazenamento. O valor especificado para *Target-Volume-in-GiB* deve ser um número inteiro maior que 100 GiB. O armazenamento por agente após a operação de atualização não pode exceder 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}'
```

Aumentar a escala verticalmente do armazenamento do agente usando a API

Para atualizar o armazenamento de um broker usando a API, consulte UpdateBrokerArmazenamento.

Provisionar throughput de armazenamento

Os agentes do Amazon MSK mantêm os dados em volumes de armazenamento. A E/S de armazenamento é consumida quando os produtores gravam no cluster, quando os dados são replicados entre os agentes e quando os consumidores leem dados que não estão na memória. O throughput de armazenamento em volume é a taxa na qual os dados podem ser gravados e lidos em um volume de armazenamento. O throughput de armazenamento provisionado é a capacidade de especificar essa taxa para os agentes em seu cluster.

Você pode especificar a taxa de transferência provisionada em MiB por segundo para clusters cujos agentes sejam maiores `kafka.m5.4xlarge` ou maiores e se o volume de armazenamento for de 10 GiB ou mais. É possível especificar o throughput provisionado durante a criação do cluster. Você também pode ativar ou desativar o throughput provisionado para um cluster que esteja no estado `ACTIVE`.

Gargalos de throughput

Há várias causas de gargalos no throughput do agente: throughput de volume, throughput de rede do Amazon EC2 para o Amazon EBS e throughput de saída do Amazon EC2. Você pode ativar o throughput do armazenamento provisionado para ajustar o throughput do volume. No entanto, as limitações de throughput do agente podem ser causadas pelo throughput de rede do Amazon EC2 para o Amazon EBS e pelo throughput de saída do Amazon EC2.

O throughput de saída do Amazon EC2 é afetado pelo número de grupos de consumidores e consumidores por grupo de consumidores. Além disso, tanto a taxa de transferência da rede do Amazon EC2 para o Amazon EBS quanto a taxa de transferência de saída do Amazon EC2 são maiores para corretores maiores.

Para volumes com tamanhos de 10 GiB ou mais, você pode provisionar um throughput de armazenamento de 250 MiB por segundo ou mais. O valor de 250 MiB por segundo é o padrão. Para provisionar a taxa de transferência de armazenamento, você deve escolher o tamanho do broker kafka.m5.4xlarge ou maior (ou kafka.m7g.2xlarge ou maior) e especificar a taxa de transferência máxima conforme mostrado na tabela a seguir.

tamanho do corretor	Throughput máximo de armazenamento (MiB/segundo)
kafka.m5.4xlarge	593
kafka.m5.8xlarge	850
kafka.m5.12xlarge	1000
kafka.m5.16xlarge	1000
kafka.m5.24xlarge	1000
kafka.m7g.2xlarge	312,5

tamanho do corretor	Throughput máximo de armazenamento (MiB/segundo)
kafka.m7g.4xlarge	625
kafka.m7g.8xlarge	1000
kafka.m7g.12xlarge	1000
kafka.m7g.16xlarge	1000

Como medir o throughput de armazenamento

Você pode usar as métricas `VolumeReadBytes` e `VolumeWriteBytes` para medir o throughput médio de armazenamento de um cluster. A soma dessas duas métricas fornece o throughput médio de armazenamento em bytes. Para obter o throughput médio de armazenamento de um cluster, defina essas duas métricas como SUM e o período como 1 minuto e então aplique a fórmula a seguir.

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

Para obter mais informações sobre as métricas `VolumeReadBytes` e `VolumeWriteBytes`, consulte the section called “Monitoramento no nível PER_BROKER”.

Atualização da configuração

Você pode atualizar sua configuração do Amazon MSK antes ou depois de ativar o throughput provisionado. No entanto, você não verá o throughput desejado até realizar estas duas ações: atualizar o parâmetro de configuração `num.replica.fetchers` e ativar o throughput provisionado.

Na configuração padrão do Amazon MSK, `num.replica.fetchers` tem um valor de 2. Para atualizar seu `num.replica.fetchers`, você pode usar os valores sugeridos na tabela a seguir. Estes valores são para fins de orientação. Recomendamos ajustar os valores com base no seu caso de uso.

tamanho do corretor	num.replica.fetchers
kafka.m5.4xlarge	4
kafka.m5.8xlarge	8
kafka.m5.12xlarge	14
kafka.m5.16xlarge	16
kafka.m5.24xlarge	16

Sua configuração atualizada pode não entrar em vigor por até 24 horas e isso pode levar mais tempo quando um volume de origem não for totalmente utilizado. No entanto, o desempenho do volume de transição é, no mínimo, igual ao desempenho dos volumes de armazenamento de origem durante o período de migração. Um volume de 1 TiB totalmente utilizado normalmente leva aproximadamente 6 horas para migrar para uma configuração atualizada.

Provisionando a taxa de transferência de armazenamento usando o AWS Management Console

1. Faça login no AWS Management Console e abra o console do Amazon MSK em https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Selecione Criar cluster.
3. Escolha Criação personalizada.
4. Especifique um nome para o cluster.
5. Na seção Armazenamento, escolha Habilitar.
6. Escolha um valor para o throughput de armazenamento por agente.
7. Selecione uma VPC, as zonas e sub-redes, além dos grupos de segurança.
8. Escolha Próximo.
9. Na parte inferior da etapa Segurança, escolha Avançar.
10. Na parte inferior da etapa Monitoramento e tags, escolha Avançar.
11. Revise as configurações do cluster e escolha Criar cluster.

Provisionando a taxa de transferência de armazenamento usando o AWS CLI

Esta seção mostra um exemplo de como você pode usar o AWS CLI para criar um cluster com a taxa de transferência provisionada ativada.

1. Copie e cole o JSON a seguir em um arquivo. Substitua os IDs de sub-rede e de grupo de segurança por valores da sua conta. Nomeie e salve o arquivo como `cluster-creation.json`.

```
{
    "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. Execute o AWS CLI comando a seguir no diretório em que você salvou o arquivo JSON na etapa anterior.

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

Como provisionar o throughput de armazenamento usando a API

Para configurar a taxa de transferência de armazenamento provisionado ao criar um cluster, use a V2. CreateCluster

Atualizando o tamanho do corretor

Você pode escalar seu cluster MSK sob demanda alterando o tamanho de seus corretores sem reatribuir partições do Apache Kafka. Alterar o tamanho dos seus agentes oferece a flexibilidade de ajustar a capacidade computacional do seu cluster MSK com base nas mudanças nas suas cargas de trabalho, sem interromper a E/S do seu cluster. O Amazon MSK usa o mesmo tamanho de agente para todos os agentes em um determinado cluster.

Esta seção descreve como atualizar o tamanho do broker para seu cluster MSK. Você pode atualizar o tamanho do cluster broker de M5 ou T3 para M7g ou de M7g para M5. Esteja ciente de que migrar para uma corretora menor pode diminuir o desempenho e reduzir a taxa de transferência máxima possível por corretora. A migração para uma corretora maior pode aumentar o desempenho, mas pode custar mais.

A atualização do tamanho do corretor acontece de forma contínua enquanto o cluster está em funcionamento. Isso significa que a Amazon MSK derruba uma corretora por vez para realizar a atualização do tamanho da corretora. Para obter informações sobre como tornar um cluster altamente disponível durante uma atualização do tamanho de um corretor, consulte. the section called “Criar clusters altamente disponíveis” Para reduzir ainda mais qualquer impacto potencial na produtividade, você pode realizar a atualização do tamanho do corretor durante um período de baixo tráfego.

Durante uma atualização do tamanho de uma corretora, você pode continuar produzindo e consumindo dados. No entanto, é necessário esperar até que a atualização seja concluída para poder reinicializar os agentes ou invocar qualquer uma das operações de atualização listadas nas operações do Amazon MSK.

Se você quiser atualizar seu cluster para um tamanho de agente menor, recomendamos que você experimente primeiro a atualização em um cluster de teste para ver como isso afeta seu cenário.

> **Important**
>
> Você não pode atualizar um cluster para um tamanho de agente menor se o número de partições por agente exceder o número máximo especificado emthe section called “ Dimensione seu cluster adequadamente: número de partições por agente”.

Atualizando o tamanho do corretor usando o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. Escolha o cluster MSK para o qual você deseja atualizar o tamanho do broker.
3. Na página de detalhes do cluster, encontre a seção Resumo dos corretores e escolha Editar tamanho do corretor.
4. Escolha o tamanho do corretor que você deseja na lista.
5. Salve as alterações.

Atualizando o tamanho do corretor usando o AWS CLI

1. Execute o comando a seguir, *ClusterArn*substituindo-o pelo Amazon Resource Name (ARN) que você obteve ao criar seu cluster. Se você não tiver o ARN do cluster, poderá encontrá-lo listando todos os clusters. Para ter mais informações, consulte the section called “Listar clusters”.

   Substitua *Current-Cluster-Version* pela versão atual do cluster e *TargetType*pelo novo tamanho que você deseja que os corretores tenham. Para saber mais sobre os tamanhos dos corretores, consultethe section called “Tamanhos de corretores”.

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

   Veja a seguir um exemplo de como usar esse comando:

```
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
```

A saída desse comando é semelhante ao seguinte JSON de exemplo.

```
{
    "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. Para obter o resultado da `update-broker-type` operação, execute o comando a seguir, substituindo *`ClusterOperationArn`* pelo ARN obtido na saída do `update-broker-type` comando.

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

A saída desse comando `describe-cluster-operation` é semelhante ao seguinte JSON de exemplo.

```
{
  "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"
    }
```

```
  }
}
```

Se `OperationState` tiver o valor `UPDATE_IN_PROGRESS`, aguarde um pouco e execute o comando `describe-cluster-operation` novamente.

Atualizando o tamanho do corretor usando a API

Para atualizar o tamanho do broker usando a API, consulte UpdateBrokerTipo.

Você pode usar `UpdateBrokerType` para atualizar o tamanho do cluster broker de M5 ou T3 para M7g ou de M7g para M5.

Atualizar a configuração do cluster do Amazon MSK

Para atualizar a configuração de um cluster, certifique-se de que ele esteja no estado `ACTIVE`. Você também deve garantir que o número de partições por agente em seu cluster do MSK esteja abaixo dos limites descritos em the section called “ Dimensione seu cluster adequadamente: número de partições por agente”. Você não pode atualizar a configuração de um cluster que exceda esses limites.

Para obter informações sobre a configuração do MSK, incluindo como criar uma configuração personalizada, quais propriedades você pode atualizar e o que acontece quando você atualiza a configuração de um cluster existente, consulte Configuração.

Atualizando a configuração de um cluster usando o AWS CLI

1. Copie o seguinte JSON e salve-o em um arquivo. Nomeie o arquivo `configuration-info.json`. *`ConfigurationArn`*Substitua pelo Amazon Resource Name (ARN) da configuração que você deseja usar para atualizar o cluster. A string do ARN deve estar entre aspas no seguinte JSON.

   Substitua *`Configuration-Revision`* pela revisão da configuração que você deseja usar. As revisões de configuração são inteiros (números inteiros) que começam em 1. Esse número inteiro não deve estar entre aspas no seguinte JSON.

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

```
}
```

2. Execute o comando a seguir, *ClusterArn*substituindo-o pelo ARN obtido ao criar seu cluster. Se você não tiver o ARN do cluster, poderá encontrá-lo listando todos os clusters. Para ter mais informações, consulte the section called “Listar clusters”.

   Substitua *Path-to-Config-Info-File* pelo caminho para o arquivo de informações de sua configuração. Se você nomeou o arquivo que criou na etapa anterior `configuration-info.json` e o salvou no diretório atual, o *Path-to-Config-Info-File* será `configuration-info.json`.

   Substitua *Current-Cluster-Version* pela versão atual do cluster.

   > **Important**
   >
   > As versões de cluster não são inteiros simples. Para encontrar a versão atual do cluster, use a DescribeClusteroperação ou o comando AWS CLI describe-cluster. Uma versão de exemplo é `KTVPDKIKX0DER`.

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

   Veja a seguir um exemplo de como usar esse comando:

```
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"
```

   A saída desse comando `update-cluster-configuration` é semelhante ao seguinte JSON de exemplo.

```
{
    "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. Para obter o resultado da `update-cluster-configuration` operação, execute o comando a seguir, substituindo *`ClusterOperationArn`* pelo ARN obtido na saída do `update-cluster-configuration` comando.

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

A saída desse comando `describe-cluster-operation` é semelhante ao seguinte JSON de exemplo.

```
{
    "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
            }
        }
    }
}
```

Nesta saída, `OperationType` é `UPDATE_CLUSTER_CONFIGURATION`. Se `OperationState` tiver o valor `UPDATE_IN_PROGRESS`, aguarde um pouco e execute o comando `describe-cluster-operation` novamente.

Atualizar a configuração de um cluster usando a API

Para usar a API para atualizar a configuração de um cluster, consulte UpdateClusterConfiguração.

Expandir um cluster do Amazon MSK

Execute esta operação do Amazon MSK quando você quiser aumentar o número de agentes em seu cluster do MSK. Para expandir um cluster, certifique-se de que ele esteja no estado `ACTIVE`.

> **Important**
>
> Certifique-se de usar esta operação do Amazon MSK se quiser expandir um cluster do MSK. Não tente adicionar agentes a um cluster sem usar essa operação.

Para obter informações sobre como reequilibrar partições depois de adicionar agentes a um cluster, consulte the section called “Reatribuir partições”.

Expandir um cluster usando o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. Escolha o cluster do MSK cujo número de agentes deseja aumentar.
3. Na página de detalhes do cluster, escolha o botão Editar ao lado do cabeçalho Detalhes do agente no nível de cluster.
4. Insira o número de agentes que você deseja que o cluster tenha por zona de disponibilidade e escolha Salvar alterações.

Expandir um cluster usando o AWS CLI

1. Execute o comando a seguir, *`ClusterArn`* substituindo-o pelo Amazon Resource Name (ARN) que você obteve ao criar seu cluster. Se você não tiver o ARN do cluster, poderá encontrá-lo listando todos os clusters. Para ter mais informações, consulte the section called “Listar clusters”.

   Substitua *`Current-Cluster-Version`* pela versão atual do cluster.

   > **Important**
   >
   > As versões de cluster não são inteiros simples. Para encontrar a versão atual do cluster, use a DescribeClusteroperação ou o comando AWS CLI describe-cluster. Uma versão de exemplo é `KTVPDKIKX0DER`.

O parâmetro *Target-Number-of-Brokers* representa o número total de nós de agente que você deseja que o cluster tenha quando essa operação for concluída com êxito. O valor especificado para *Target-Number-of-Brokers* deve ser um número inteiro maior do que o número atual de agentes no cluster. Também deve ser um múltiplo do número de zonas de disponibilidade.

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

A saída dessa operação `update-broker-count` é semelhante ao seguinte JSON.

```
{

    "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. Para obter o resultado da `update-broker-count` operação, execute o comando a seguir, substituindo *ClusterOperationArn* pelo ARN obtido na saída do `update-broker-count` comando.

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

A saída desse comando `describe-cluster-operation` é semelhante ao seguinte JSON de exemplo.

```
{
    "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
        }
    }
}
```

Nesta saída, `OperationType` é `INCREASE_BROKER_COUNT`. Se `OperationState` tiver o valor `UPDATE_IN_PROGRESS`, aguarde um pouco e execute o comando `describe-cluster-operation` novamente.

Expandir um cluster usando a API

Para aumentar o número de corretores em um cluster usando a API, consulte UpdateBrokerCount.

Remover um agente de um cluster Amazon MSK

Use essa operação do Amazon MSK quando quiser remover corretores dos clusters provisionados pelo Amazon Managed Streaming for Apache Kafka (MSK). Você pode reduzir a capacidade de armazenamento e computação do seu cluster removendo conjuntos de corretores, sem impacto na disponibilidade, risco de durabilidade de dados ou interrupção em seus aplicativos de streaming de dados.

Você pode adicionar mais agentes ao seu cluster para lidar com o aumento do tráfego e remover agentes quando o tráfego diminuir. Com a capacidade de adição e remoção de corretores, você pode utilizar melhor sua capacidade de cluster e otimizar seus custos de infraestrutura MSK. A remoção do agente oferece controle em nível de intermediário sobre a capacidade existente do cluster para atender às suas necessidades de carga de trabalho e evitar a migração para outro cluster.

Use o AWS console, a interface de linha de comando (CLI), o SDK ou AWS CloudFormation para reduzir o número de agentes do seu cluster provisionado. O MSK escolhe os agentes que não têm nenhuma partição neles (exceto os tópicos básicos) e impede que os aplicativos produzam dados para esses corretores, ao mesmo tempo que os remove com segurança do cluster.

Você deve remover um agente por zona de disponibilidade, se quiser reduzir o armazenamento e a computação de um cluster. Por exemplo, você pode remover dois agentes de um cluster de duas zonas de disponibilidade ou três corretores de um cluster de três zonas de disponibilidade em uma única operação de remoção de agentes.

Para obter informações sobre como rebalancear partições depois de remover os brokers de um cluster, consulte. the section called “Reatribuir partições”

Você pode remover agentes de todos os clusters provisionados por MSK baseados em M5 e M7g, independentemente do tamanho da instância.

A remoção do broker é suportada nas versões 2.8.1 e superiores do Kafka, inclusive nos clusters do modo Kraft.

Tópicos

Prepare-se para remover os corretores removendo todas as partições

Antes de iniciar o processo de remoção do corretor, primeiro mova todas as partições, exceto aquelas de tópicos `__amazon_msk_canary` e `__amazon_msk_canary_state` das corretoras que você planeja remover. Esses são tópicos internos que o Amazon MSK cria para métricas de saúde e diagnóstico do cluster.

Você pode usar as APIs de administração do Kafka ou o Cruise Control para mover partições para outros corretores que você pretende manter no cluster. Consulte Reatribuir partições.

Exemplo de processo para remover partições

Esta seção é um exemplo de como remover partições do broker que você pretende remover. Suponha que você tenha um cluster com 6 corretores, 2 corretores em cada AZ, e ele tenha quatro tópicos:

- `__amazon_msk_canary`
- `__consumer_offsets`

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

1. Crie uma máquina cliente conforme descrito em Criar uma máquina cliente.
2. Depois de configurar a máquina cliente, execute o comando a seguir para listar todos os tópicos disponíveis em seu cluster.

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

   Neste exemplo, vemos quatro nomes de tópicos`__amazon_msk_canary`, `__consumer_offsets__amazon_msk_connect_offsets_my-mskc-connector_12345678-09e7-c657f7e4ff32-2`,, `msk-brk-rmv` e.
3. Crie um arquivo json chamado `topics.json` na máquina cliente e adicione todos os nomes dos tópicos do usuário, como no exemplo de código a seguir. Você não precisa incluir o nome do `__amazon_msk_canary` tópico, pois esse é um tópico gerenciado pelo serviço que será movido automaticamente quando necessário.

   ```
   {
   "topics": [
   {"topic": "msk-brk-rmv"},
   {"topic": "__consumer_offsets"},
   {"topic": "__amazon_msk_connect_offsets_my-mskc-connector_12345678-09e7-
   c657f7e4ff32-2"}
   ],
   "version":1
   }
   ```
4. Execute o comando a seguir para gerar uma proposta para mover partições para apenas 3 corretores dos 6 corretores no 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. Crie um arquivo chamado `reassignment-file.json` e copie o comando que `proposed partition reassignment configuration` você obteve acima.
6. Execute o comando a seguir para mover as partições que você especificou no`reassignment-file.json`.

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

A saída será semelhante à seguinte:

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

7. Execute o comando a seguir para verificar se todas as partições foram movidas.

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

A saída será semelhante à seguinte. Monitore o status até que todas as partições nos tópicos solicitados tenham sido reatribuídas com sucesso:

```
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. Quando o status indicar que a reatribuição de partição para cada partição foi concluída, monitore as `UserPartitionExists` métricas por 5 minutos para garantir que elas sejam exibidas `0` para os corretores dos quais você moveu as partições. Depois de confirmar isso, você pode continuar removendo o agente do cluster.

Remover um corretor com o AWS Management Console

Para remover corretores com o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. Escolha o cluster MSK que contém os agentes que você deseja remover.
3. Na página de detalhes do cluster, escolha o botão Ações e selecione a opção Editar número de corretores.

4. Insira o número de corretores que você deseja que o cluster tenha por zona de disponibilidade. O console resume o número de corretores nas zonas de disponibilidade que serão removidos. Certifique-se de que é isso que você quer.
5. Escolha Salvar alterações.

Para evitar a remoção acidental do corretor, o console solicita que você confirme que deseja excluir os corretores.

Remova um corretor com a AWS CLI

Execute o comando a seguir, `ClusterArn` substituindo-o pelo Amazon Resource Name (ARN) que você obteve ao criar seu cluster. Se você não tiver o ARN do cluster, poderá encontrá-lo listando todos os clusters. Para obter mais informações, consulte Listando clusters do Amazon MSK. `Current-Cluster-Version`Substitua pela versão atual do cluster.

> **Important**
>
> As versões de cluster não são inteiros simples. Para encontrar a versão atual do cluster, use a DescribeClusteroperação ou o comando AWS CLI describe-cluster. Uma versão de exemplo é `KTVPDKIKX0DER`.

O parâmetro *`Target-Number-of-Brokers`* representa o número total de nós de agente que você deseja que o cluster tenha quando essa operação for concluída com êxito. O valor especificado para o *`número-alvo de corretores`* deve ser um número inteiro menor que o número atual de corretores no cluster. Também deve ser um múltiplo do número de zonas de disponibilidade.

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

A saída dessa operação `update-broker-count` é semelhante ao seguinte JSON.

```
{
"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
        }
    }
}
```

Nesta saída, `OperationType` é `DECREASE_BROKER_COUNT`. Se `OperationState` tiver o valor `UPDATE_IN_PROGRESS`, aguarde um pouco e execute o comando `describe-cluster-operation` novamente.

Remover um corretor com a AWS API

Para remover corretores em um cluster usando a API, consulte UpdateBrokerCount in the Amazon Managed Streaming for Apache Kafka API Reference.

Atualização das configurações de segurança de um cluster

Use essa operação do Amazon MSK para atualizar as configurações de autenticação e criptografia cliente-agente do seu cluster do MSK. Você também pode atualizar a Autoridade de Segurança Privada usada para assinar certificados para autenticação TLS mútua. Você não pode alterar a configuração de criptografia no cluster (agente para agente).

O cluster deve estar no estado `ACTIVE` para que você atualize suas configurações de segurança.

Se você ativar a autenticação usando IAM, SASL ou TLS, também deverá ativar a criptografia entre clientes e agentes. A tabela a seguir mostra as combinações possíveis.

Autenticação	Opções de criptografia cliente-agente	Criptografia agente-agente
Unauthenticated	TLS, PLAINTEXT, TLS_PLAINTEXT	Pode estar ativado ou desativado.

Autenticação	Opções de criptografia cliente-agente	Criptografia agente-agente
mTLS	TLS, TLS_PLAINTEXT	Precisa estar ativado.
SASL/SCRAM	TLS	Precisa estar ativado.
SASL/IAM	TLS	Precisa estar ativado.

Quando a criptografia cliente-agente estiver definida como `TLS_PLAINTEXT` e a autenticação do cliente estiver definida como `mTLS`, o Amazon MSK criará dois tipos de receptores aos quais os clientes se conectarão: um ouvinte para os clientes se conectarem usando a autenticação mTLS com criptografia TLS e outro para os clientes se conectarem sem autenticação ou criptografia (texto simples).

Para obter mais informações sobre as configurações de segurança, consulte Segurança.

Atualizando as configurações de segurança de um cluster usando o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. Selecione o cluster do MSK que deseja atualizar.
3. Na seção Configurações de segurança, escolha Editar.
4. Escolha as configurações de autenticação e criptografia que deseja aplicar para o cluster e, em seguida, escolha Salvar alterações.

Atualizando as configurações de segurança de um cluster usando o AWS CLI

1. Crie um arquivo JSON contendo as configurações de criptografia desejadas para o cluster. Veja um exemplo a seguir.

> **Note**
>
> Você só pode atualizar a configuração de criptografia cliente-agente. Você não pode atualizar a configuração de criptografia no cluster (agente para agente).

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

2. Crie um arquivo JSON contendo as configurações de autenticação desejadas para o cluster. Veja um exemplo a seguir.

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

3. Execute o seguinte AWS CLI comando:

```
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
```

A saída dessa operação `update-security` é semelhante ao seguinte JSON.

```
{

    "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. Para ver o status da `update-security` operação, execute o comando a seguir, substituindo *`ClusterOperationArn`* pelo ARN obtido na saída do `update-security` comando.

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

A saída desse comando `describe-cluster-operation` é semelhante ao seguinte JSON de exemplo.

```
{
```

```
    "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": {}
    }
}
```

Se `OperationState` tiver o valor `PENDING` ou `UPDATE_IN_PROGRESS`, aguarde um pouco e execute o comando `describe-cluster-operation` novamente.

Atualizar as configurações de segurança de um cluster usando a API

Para atualizar as configurações de segurança de um cluster usando a API, consulte UpdateSecurity.

> **Note**
>
> As operações de API AWS CLI e de atualização das configurações de segurança de um cluster são idempotentes. Isso significa que se você invocar a operação de atualização de segurança e especificar uma configuração de autenticação ou criptografia que seja a mesma configuração que o cluster já tem, essa configuração não será alterada.

Como reinicializar um agente para um cluster do Amazon MSK

Use esta operação do Amazon MSK quando quiser reinicializar um agente para seu cluster do MSK. Para reinicializar um agente para um cluster, certifique-se de que o cluster esteja no estado `ACTIVE`.

O serviço Amazon MSK pode reinicializar os agentes do seu cluster do MSK durante a manutenção do sistema, como aplicação de patches ou atualizações de versão. A reinicialização manual de um agente permite testar a resiliência de seus clientes Kafka para determinar como eles respondem à manutenção do sistema.

Reinicializando um corretor usando o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. Escolha o cluster do MSK cujo agente deseja reinicializar.
3. Role para baixo até a seção Detalhes do agente e escolha o agente que deseja reinicializar.
4. Escolha o botão Reiniciar o agente.

Reinicializando um corretor usando o AWS CLI

1. Execute o comando a seguir, *ClusterArn* substituindo-o pelo Amazon Resource Name (ARN) obtido ao criar seu cluster e pelo *BrokerId* ID do broker que você deseja reinicializar.

   > **Note**
   >
   > A operação `reboot-broker` só é compatível com a reinicialização de um agente por vez.

   Se você não tiver o ARN do cluster, poderá encontrá-lo listando todos os clusters. Para ter mais informações, consulte the section called “Listar clusters”.

   Se você não tiver os IDs do agente para seu cluster, poderá encontrá-los listando os nós do agente. Para obter mais informações, consulte list-nodes.

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

   A saída dessa operação `reboot-broker` é semelhante ao seguinte JSON.

   ```
   {

       "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. Para obter o resultado da `reboot-broker` operação, execute o comando a seguir, *`ClusterOperationArn`*substituindo-o pelo ARN obtido na saída do `reboot-broker` comando.

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

A saída desse comando `describe-cluster-operation` é semelhante ao seguinte JSON de exemplo.

```
{
    "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": {}
    }
}
```

Quando a operação de reinicialização estiver concluída, o `OperationState` será `REBOOT_COMPLETE`.

Como reinicializar um agente usando a API

Para reinicializar um agente em um cluster usando a API, consulte RebootBroker.

Impacto da reinicialização do corretor durante a aplicação de patches e outras manutenções

Periodicamente, o Amazon MSK atualiza o software de seus corretores. Essas atualizações não terão impacto nas gravações e leituras de seus aplicativos se você seguir as melhores práticas.

O Amazon MSK usa atualizações contínuas de software para manter a alta disponibilidade de seus clusters. Durante esse processo, os corretores são reiniciados um de cada vez e Kafka transfere automaticamente a liderança para outro corretor on-line. Os clientes Kafka têm mecanismos integrados para detectar automaticamente a mudança na liderança das partições e continuar gravando e lendo dados em um cluster MSK.

Depois que um corretor fica off-line, é normal ver erros transitórios de desconexão em seus clientes. Você também observará por um breve período (até 2 minutos, normalmente menos) alguns picos na latência de leitura e gravação do p99 (normalmente alta em milissegundos, até aproximadamente 2 segundos). Esses picos são esperados e são causados pela reconexão do cliente com uma nova corretora líder; isso não afeta sua produção ou consumo e será resolvido após a reconexão.

Você também observará um aumento na métrica`UnderReplicatedPartitions`, o que é esperado, pois as partições do broker que foi encerrado não estão mais replicando dados. Isso não afeta as gravações e leituras dos aplicativos, pois as réplicas dessas partições hospedadas em outros agentes agora atendem às solicitações.

Após a atualização do software, quando a corretora volta a ficar online, ela precisa “se atualizar” sobre as mensagens produzidas enquanto estava offline. Durante o catch up, você também pode observar um aumento no uso da taxa de transferência do volume e da CPU. Isso não deve ter impacto nas gravações e leituras no cluster se você tiver recursos suficientes de CPU, memória, rede e volume em seus agentes.

Atribuir tags a um cluster do Amazon MSK

É possível atribuir seus próprios metadados na forma de tags a um recurso do Amazon MSK, como um cluster do MSK. Uma tag é um par de chave-valor que você define para o recurso. Usar tags é uma maneira simples, porém poderosa, de gerenciar AWS recursos e organizar dados, incluindo dados de faturamento.

Tópicos

- Conceitos Básicos de Tags
- Monitorar custos usando a marcação
- Restrições de tags
- Atribuição de tags a recursos usando a API do Amazon MSK

Conceitos Básicos de Tags

É possível usar a API do Amazon MSK para concluir as seguintes tarefas:

- Adicionar tags a um recurso do Amazon MSK.
- Listar as tags de um recurso do Amazon MSK.
- Remover as tags de um recurso do Amazon MSK.

É possível usar tags para categorizar os recursos do Amazon MSK. Por exemplo, é possível categorizar os clusters do Amazon MSK por finalidade, proprietário ou ambiente. Como você define a chave e o valor para cada marca, você pode criar um conjunto de categorias personalizado para atender às suas necessidades específicas. Por exemplo, você pode definir um conjunto de tags que ajude a monitorar os clusters por proprietário e aplicativo associado.

Estes são diversos exemplos de tags:

- `Project: Project name`
- `Owner: Name`
- `Purpose: Load testing`
- `Environment: Production`

Monitorar custos usando a marcação

Você pode usar tags para categorizar e monitorar seus AWS custos. Quando você aplica tags aos seus AWS recursos, incluindo clusters do Amazon MSK, seu relatório de alocação de AWS custos inclui o uso e os custos agregados por tags. Você pode organizar seus custos de vários serviços aplicando tags que representam categorias de negócios (como centros de custos, nomes de aplicativos ou proprietários). Para obter mais informações, consulte Usar etiquetas de alocação de custos para relatórios de faturamento personalizados no Manual do usuário do AWS Billing .

Restrições de tags

As restrições a seguir se aplicam a tags no Amazon MSK.

Restrições básicas

- O número máximo de tags por recurso é 50.

- As chaves e os valores de tags diferenciam maiúsculas de minúsculas.
- Não é possível alterar nem editar as tags de um recurso excluído.

Restrições de chaves de marcas

- Cada chave de marca deve ser exclusiva. Se você adicionar uma marca com uma chave que já estiver em uso, sua nova marca existente substituirá o par de chave-valor.
- Não é possível iniciar uma chave de tag com `aws:`, pois esse prefixo é reservado para uso pela AWS. A AWS cria tags que começam com esse prefixo em seu nome, mas você não pode editá-las ou excluí-las.
- As chaves de marca devem ter entre 1 e 128 caracteres Unicode.
- As chaves de marca devem conter os seguintes caracteres: letras Unicode, dígitos, espaço em branco e os seguintes caracteres especiais: `_ . / = + - @`.

Restrições de valor de marcas

- Os valores de marca devem ter entre 0 e 255 caracteres Unicode.
- Os valores de marca podem estar em branco. Caso contrário, elas devem conter os seguintes caracteres: letras Unicode, dígitos, espaço em branco e qualquer um dos seguintes caracteres especiais: `_ . / = + - @`.

Atribuição de tags a recursos usando a API do Amazon MSK

É possível usar as seguintes operações para atribuir ou excluir tags de um recurso do Amazon MSK ou para listar o conjunto atual de tags de um recurso:

- ListTagsForResource
- TagResource
- UntagResource

Configuração do Amazon MSK

O Amazon Managed Streaming for Apache Kafka fornece uma configuração padrão para corretores, tópicos e nós do Apache. ZooKeeper Você também pode criar configurações personalizadas e usá-las para criar novos clusters do MSK ou atualizar clusters existentes. Uma configuração do MSK consiste em um conjunto de propriedades e seus valores correspondentes.

Tópicos

Configurações personalizadas do MSK

É possível usar o Amazon MSK para criar uma configuração personalizada do MSK na qual você define as seguintes propriedades. As propriedades que você não define explicitamente obtêm os valores que têm em the section called “Configuração padrão”. Para obter mais informações sobre as propriedades da configuração, consulte Configuração do Apache Kafka.

Propriedades de configuração do Apache Kafka

Nome	Descrição
allow.everyone.if.no.acl.found	Se você quiser definir essa propriedade como `false`, primeiro defina as ACLs do Apache Kafka para seu cluster. Se definir essa propriedade como `false` e não definir primeiro as ACLs do Apache Kafka, você perderá o acesso ao cluster. Se isso acontecer , é possível atualizar a configuração novamente e definir essa propriedade como `true` para recuperar o acesso ao cluster.
auto.create.topics.enable	Habilita a criação automática de tópicos no servidor.

Nome	Descrição
compression.type	O tipo de compactação final de um determina do tópico. Você pode definir essa proprieda de para os codecs de compactação padrão (`gzip`, `snappy`, `lz4` e `zstd`). Além disso, também aceita `uncompressed` . Esse valor é equivalente a nenhuma compactação. Se você definir o valor como `producer`, isso significa reter o codec de compactação original definido pelo produtor.
connections.max.idle.ms	O tempo limite de conexões ociosas em milissegundos. Os threads do processador de soquete do servidor fecham as conexões que estiverem ociosas há mais tempo que o que o valor definido para essa propriedade.
default.replication.factor	O fator de replicação padrão para tópicos criados automaticamente.
delete.topic.enable	Habilita a operação de exclusão de tópico. Se desativar essa configuração, você não poderá excluir um tópico por meio da ferramenta de administração.
group.initial.rebalance.delay.ms	O período que o coordenador do grupo espera que mais consumidores de dados ingressem em um novo grupo antes de executar a primeira operação de rebalanceamento. Um atraso mais longo significa potencialmente menos rebalanceamentos, mas aumenta o tempo até o início do processamento.

Nome	Descrição
group.max.session.timeout.ms	Tempo limite máximo de sessão para consumidores registrados. Tempos limite mais longos permitem que os consumidores tenham mais tempo para processar mensagens entre pulsações ao custo de mais tempo para detectar falhas.
group.min.session.timeout.ms	Tempo limite mínimo de sessão para consumidores registrados. Tempos limite mais curtos resultam em detecção mais rápida de falhas ao custo de pulsações mais frequente s do consumidor. Isso pode sobrecarregar os recursos do agente.
leader.imbalance.per.broker.percentage	A proporção de desequilíbrio de líder permitida por agente. O controlador aciona um balanceamento de líder caso ele ultrapasse esse valor por agente. Esse valor é especific ado em porcentagem.
log.cleaner.delete.retention.ms	Período de tempo que você deseja que o Apache Kafka mantenha registros excluídos. O valor mínimo é 0.

Nome	Descrição
log.cleaner.min.cleanable.ratio	Essa propriedade de configuração pode ter valores entre 0 e 1. Esse valor determina a frequência na qual o compactador de logs tenta limpar o log (se a compactação de logs estiver habilitada). Por padrão, o Apache Kafka evita limpar um log se mais de 50% do log tiver sido compactado. Essa proporção limita o espaço máximo que o log desperdiça com duplicata s (em 50%, isso significa que até 50% do log pode ser de duplicatas). Uma proporção maior significa menos limpezas mais eficientes, mas também mais espaço desperdiçado no log.
log.cleanup.policy	A política de limpeza padrão para segmentos além da janela de retenção. Uma lista de políticas válidas separadas por vírgulas. As políticas válidas são `delete` e `compact`. Para clusters habilitados para armazenamento em camadas, a política válida é somente `delete`.
log.flush.interval.messages	O número de mensagens acumuladas em uma partição de log antes que as mensagens sejam liberadas para o disco.
log.flush.interval.ms	O período máximo em milissegundos no qual uma mensagem em qualquer tópico permanece na memória antes de ser liberada para o disco. Se você não definir esse valor, o sistema usará o valor em log.flush.schedule r.interval.ms. O valor mínimo é 0.

Nome	Descrição
log.message.timestamp.difference.max.ms	A diferença máxima de tempo entre o carimbo de data/hora em que um agente recebe uma mensagem e o carimbo de data/hora especific ado na mensagem. Se log.message.timest amp.type=CreateTime, uma mensagem será rejeitada se a diferença no timestamp exceder esse limite. Essa configuração será ignorada se log.message.timestamp.type= Time. LogAppend
log.message.timestamp.type	Especifica se o carimbo de data/hora na mensagem é o horário de criação da mensagem ou da adição no log. Os valores permitidos são `CreateTime` e `LogAppend Time` .
log.retention.bytes	Tamanho máximo do log antes de ser excluído.
log.retention.hours	Número de horas para manter um arquivo de log antes de excluí-lo, terciário à propriedade log.retention.ms.
log.retention.minutes	Número de minutos para manter um arquivo de log antes de excluí-lo, secundário à proprieda de log.retention.ms. Se você não definir esse valor, o sistema usará o valor de log.reten tion.hours.
log.retention.ms	Número de milissegundos para manter um arquivo de log antes de excluí-lo (em milissegu ndos). Se não for definido, o valor de log.reten tion.minutes será usado.

Nome	Descrição
log.roll.ms	Tempo máximo para que um novo segmento de log seja implantado (em milissegundos). Se você não definir essa propriedade, o sistema usará o valor de log.roll.hours. O valor mínimo possível para essa propriedade é 1.
log.segment.bytes	Tamanho máximo de um único arquivo de log.
max.incremental.fetch.session.cache.slots	Número máximo de sessões de busca incrementais mantidas.
message.max.bytes	O maior tamanho de lote de registros que o Kafka permite. Se você aumentar esse valor e houver consumidores anteriores à versão 0.10.2, também será necessário aumentar o tamanho de busca dos consumidores para que eles possam buscar lotes de registros desse tamanho. O formato de mensagem mais recente sempre agrupa as mensagens em lotes visando eficiência. As versões anteriores de formato de mensagem não agrupam em lotes os registros não compactados, e, nesse caso, esse limite é aplicável somente a um único registro. É possível definir esse valor por tópico com a configuração max.message.bytes de nível do tópico.

Nome	Descrição
min.insync.replicas	Quando um produtor define acks como `"all"` (ou `"-1"`), o valor em min.insync.replicas especifica o número mínimo de réplicas que devem confirmar uma gravação para que a gravação seja considerada bem-sucedida. Se esse mínimo não puder ser atingido, o produtor cria uma exceção (NotEnoughReplicas ou NotEnoughReplicasAfterAppend). Você pode usar valores em min.insync.replica s e acks para forçar maiores garantias de durabilidade. Por exemplo, você poderia criar um tópico com um fator de replicação de 3, definir min.insync.replicas como 2 e produzir com acks de `"all"`. Isso garante que o produtor gere uma exceção se a maioria das réplicas não receber uma gravação.
num.io.threads	O número de threads que o servidor usa para processar solicitações, que podem incluir E/S de disco.
num.network.threads	O número de threads que o servidor usa para receber solicitações da rede e enviar respostas para ela.
num.partitions	Número padrão de partições de log por tópico.
num.recovery.threads.per.data.dir	O número de threads por diretório de dados a ser usado para recuperar logs na inicialização e para liberá-los no desligamento.

Nome	Descrição
num.replica.fetchers	O número de threads de busca usados para replicar mensagens de um agente de origem. Se você aumentar esse valor, poderá aumentar o nível de paralelismo de E/S no agente seguidor.
offsets.retention.minutes	Depois que um grupo de consumidores perde todos os consumidores (isto é, torna-se vazio), seus deslocamentos são mantidos durante esse período de retenção antes de serem descartados. Para consumidores autônomos (ou seja, que usam atribuição manual), os deslocamentos expiram depois da última confirmação somada a esse período de retenção.
offsets.topic.replication.factor	O fator de replicação do tópico de deslocamento. Defina esse valor mais alto para garantir a disponibilidade. A criação do tópico interno falha até que o tamanho do cluster atenda a esse requisito de fator de replicação.
replica.fetch.max.bytes	O número de bytes de mensagens para tentar buscar para cada partição. Esse não é um máximo absoluto. Se o primeiro lote de registros na primeira partição não vazia da busca for maior que esse valor, o lote de registros será retornado para garantir o progresso. As propriedades message.max.bytes (configuração do agente) ou max.message.bytes (configuração do tópico) definem o tamanho máximo do lote de registros aceito pelo agente.

Nome	Descrição
replica.fetch.response.max.bytes	O número máximo de bytes esperado para toda a resposta de busca. Os registros são buscados em lotes e, se o primeiro lote de registros na primeira partição não vazia da busca for maior que esse valor, o lote de registros ainda será retornado para garantir o progresso. Esse não é um máximo absoluto. As propriedades message.max.bytes (configuração do agente) ou max.message.bytes (configuração do tópico) especificam o tamanho máximo do lote de registros aceito pelo agente.
replica.lag.time.max.ms	Se um seguidor não enviou nenhuma solicitação de busca ou se não consumiu até o deslocamento final do log do líder por pelo menos esse número de milissegundos, o líder remove o seguidor do ISR. MinValue: 10000 MaxValue = 30000
replica.selector.class	O nome da classe totalmente qualificado que implementa. ReplicaSelector O agente usa esse valor para encontrar a réplica de leitura preferencial. Se estiver usando a versão 2.4.1 ou mais recente do Apache Kafka e quiser permitir que os clientes busquem da réplica mais próxima, defina essa propriedade como `org.apache.kafka.common.replica.RackAwareReplicaSelector`. Para ter mais informações, consulte the section called “Apache Kafka versão 2.4.1 (use 2.4.1.1 alternativamente)”.

Nome	Descrição
replica.socket.receive.buffer.bytes	O buffer de recebimento do soquete para solicitações de rede.
socket.receive.buffer.bytes	O buffer SO_RCVBUF dos soquetes do servidor de soquetes. O valor mínimo que você pode definir para essa propriedade é -1. Se o valor for -1, o Amazon MSK usará o sistema operacional padrão.
socket.request.max.bytes	O número máximo de bytes em uma solicitação de soquete.
socket.send.buffer.bytes	O buffer SO_SNDBUF dos soquetes do servidor de soquetes. O valor mínimo que você pode definir para essa propriedade é -1. Se o valor for -1, o Amazon MSK usará o sistema operacional padrão.
transaction.max.timeout.ms	Tempo limite máximo para transações. Se o tempo de transação solicitado de um cliente exceder esse valor, o corretor retornará um erro em InitProducerIdRequest. Isso impede que um cliente use um tempo limite muito grande e pode impedir que os consumidores leiam os tópicos incluídos na transação.
transaction.state.log.min.isr	A configuração de min.insync.replicas substituída para o tópico de transação.
transaction.state.log.replication.factor	O fator de replicação do tópico de transação. Defina essa propriedade com um valor maior para aumentar a disponibilidade. A criação do tópico interno falha até que o tamanho do cluster atenda a esse requisito de fator de replicação.

Nome	Descrição
transactional.id.expiration.ms	O tempo em milissegundos que o coordenador da transação deve aguardar para receber qualquer atualização do status da transação atual antes que o coordenador expire sua ID transacional. Essa configuração também influencia a expiração do ID do produtor porque faz com que os IDs do produtor expirem quando esse tempo decorrer após a última gravação com o ID do produtor fornecido. Os IDs do produtor podem expirar mais cedo se a última gravação de ID do produtor for excluída devido às configurações de retenção do tópico. O valor mínimo para essa propriedade é de 1 milissegundo.
unclean.leader.election.enable	Indica se as réplicas que não estão no conjunto ISR devem atuar como líderes em último recurso, mesmo que isso possa resultar em perda de dados.
zookeeper.connection.timeout.ms	ZooKeeper clusters de modos. Tempo máximo que o cliente espera para estabelecer uma conexão. ZooKeeper Se você não definir esse valor, o sistema usará o valor de zookeeper .session.timeout.ms. MinValue = 6000 MaxValue (inclusive) = 18000

Nome	Descrição
zookeeper.session.timeout.ms	ZooKeeper clusters de modos. O tempo limite da ZooKeeper sessão do Apache em milissegundos. MinValue = 6000 MaxValue (inclusive) = 18000

Para saber como criar uma configuração personalizada do MSK, listar todas as configurações ou descrevê-las, consulte the section called “Operações de configuração”. Para criar um cluster do MSK com uma configuração personalizada do MSK ou para atualizar um cluster com uma nova configuração personalizada, consulte Como funciona.

Quando você atualiza o cluster existente do MSK com uma configuração personalizada do MSK, o Amazon MSK faz reinicializações contínuas quando necessário, empregando as práticas recomendadas para minimizar o tempo de inatividade do cliente. Por exemplo, depois que o Amazon MSK reinicia cada agente, o Amazon MSK tenta deixar o agente recuperar os dados que possam ter sido perdidos pelo agente durante a atualização da configuração antes de avançar para o próximo agente.

Configuração dinâmica

Além das propriedades de configuração fornecidas pelo Amazon MSK, você pode definir dinamicamente as propriedades de configuração em nível de cluster e de agente que não exigem uma reinicialização do agente. É possível definir dinamicamente algumas propriedades de configuração. Trata-se das propriedades que não estão marcadas como somente leitura na tabela em Configurações do agente na documentação do Apache Kafka. Para obter informações sobre a configuração dinâmica e comandos de exemplo, consulte Atualização das configurações do agente na documentação do Apache Kafka.

> **Note**
>
> É possível definir a propriedade `advertised.listeners`, mas não a propriedade `listeners`.

Configuração no nível de tópico

Você pode usar os comandos do Apache Kafka para definir ou modificar propriedades de configuração em nível de tópico para tópicos novos e existentes. Para obter mais informações sobre as propriedades de configuração no nível de tópico e exemplos sobre como defini-las, consulte Configurações no nível de tópico na documentação do Apache Kafka.

Estados de configuração

Uma configuração do Amazon MSK pode estar em um dos seguintes estados. Para realizar uma operação em uma configuração, a configuração deve estar no estado `ACTIVE` ou `DELETE_FAILED`:

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

A configuração padrão do Amazon MSK

Quando você cria um cluster do MSK sem especificar uma configuração personalizada do MSK, o Amazon MSK cria e usa uma configuração padrão com os valores apresentados na tabela a seguir. Para propriedades que não estejam nessa tabela, o Amazon MSK usará os padrões associados à sua versão do Apache Kafka. Para obter uma lista desses valores padrão, consulte Configuração do Apache Kafka.

Valores padrão de configuração

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
allow.everyone.if.no.acl.found	Se nenhum padrão de recurso corresponder a um recurso específico, o recurso não terá ACLs associadas. Nesse caso, se você definir essa propriedade	`true`	`true`

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
	como `true`, todos os usuários terão acesso ao recurso, não apenas os superusuários.		
auto.create.topics.enable	Habilita a criação automática de um tópico no servidor.	`false`	`false`
auto.leader.rebalance.enable	Habilita o equilíbrio de líderes automáticos. Se necessário, um thread em segundo plano verifica e inicia o balanceamento do líder em intervalos regulares.	`true`	`true`
default.replication.factor	Fatores de replicação padrão para tópicos criados automaticamente.	O valor é 3 para clusters em 3 zonas de disponibilidade e 2 para clusters em 2 zonas de disponibilidade.	O valor é 3 para clusters em 3 zonas de disponibilidade e 2 para clusters em 2 zonas de disponibilidade.

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
local.retention.bytes	O tamanho máximo dos segmentos de log locais de uma partição antes que ela exclua os segmentos antigos. Se você não definir esse valor, o sistema usará o valor de log.retention.bytes. O valor efetivo sempre deve ser menor que ou igual ao valor de log.retention.bytes. O valor padrão de -2 indica que não há limite para a retenção local. Isso corresponde à configuração de -1 para retention.ms/bytes. As propriedades local.retention.ms e local.retention.bytes são semelhantes a log.retention, pois são usadas para determinar por quanto tempo os segmentos de log devem permanecer no armazenamento local. As configura	-2 para ilimitado	-2 para ilimitado

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
	ções existentes de log.retention.* são configurações de retenção para a partição do tópico. Isso inclui armazenamento local e remoto. Valores válidos: números inteiros em [-2; +Inf]		

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
local.retention.ms	O número de milissegundos para a retenção do segmento de log local antes da exclusão. Se você não definir esse valor, o Amazon MSK usará o valor de log.retention.ms. O valor efetivo sempre deve ser menor que ou igual ao valor de log.retention.bytes. O valor padrão de -2 indica que não há limite para a retenção local. Isso corresponde à configuração de -1 para retention.ms/bytes. Os valores de local.retention.ms e local.retention.bytes são semelhantes a log.retention. O MSK usa essa configuração para determinar por quanto tempo os segmentos de log devem permanecer no armazenamento local. As configura	-2 para ilimitado	-2 para ilimitado

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
	ções existentes de log.retention.* são configurações de retenção para a partição do tópico. Isso inclui armazenamento local e remoto. Os valores válidos são números inteiros maiores que 0.		

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
log.message.timestamp.difference.max.ms	A diferença máxima permitida entre o timestamp em que um agente recebe uma mensagem e o timestamp especificado na mensagem. Se log.message.timestamp.type=CreateTime, uma mensagem será rejeitada se a diferença no timestamp exceder esse limite. Essa configuração será ignorada se log.message.timestamp.type= Time. LogAppend Para evitar a repetição desnecessária e frequente de registros, a diferença máxima permitida para o carimbo de data/hora não deve ser maior que log.retention.ms.	9223372036854775807	86400000 para Kafka 2.8.2.tiered
log.segment.bytes	O tamanho máximo de um único arquivo de log.	1073741824	134217728

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
min.insync.replicas	Quando um produtor define o valor de confirmações (as confirmações que o produtor receber o agente do Kafka) como `"all"` (ou `"-1"`), o valor em min.insync.replicas especifica o número mínimo de réplicas que devem confirmar uma gravação para que a gravação seja considerada bem-sucedida. Se esse valor não atingir esse mínimo, o produtor gera uma exceção (NotEnoughReplicas ou NotEnough ReplicasAfterAppend). Quando você usar os valores em min.insync.replicas e acks juntos, será possível forçar maiores garantias de durabilidade. Por exemplo, você poderia criar um tópico com um	O valor é 2 para clusters em 3 zonas de disponibilidade e 1 para clusters em 2 zonas de disponibilidade.	O valor é 2 para clusters em 3 zonas de disponibilidade e 1 para clusters em 2 zonas de disponibilidade.

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
	fator de replicação de 3, definir min.insync.replicas como 2 e produzir com acks de `"all"`. Isso garante que o produtor gere uma exceção se a maioria das réplicas não receber uma gravação.		
num.io.threads	O número de threads que o servidor usa para produzir solicitações, que podem incluir E/S de disco.	8	max(8, vCPUs), no qual as vCPUs dependem do tamanho da instância do agente
num.network.threads	O número de threads que o servidor usa para receber solicitações da rede e enviar respostas para a rede.	5	max(5, vCPUs/2), no qual as vCPUs dependem do tamanho da instância do agente
num.partitions	Número padrão de partições de log por tópico.	1	1

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
num.replica.fetchers	O número de threads de busca usados para replicar mensagens de um agente de origem. Se você aumentar esse valor, poderá aumentar o grau de paralelismo de E/S no agente seguidor.	2	max(2, vCPUs/4), no qual as vCPUs dependem do tamanho da instância do agente
remote.log.msk.disable.policy	Usado com remote.storage.enable para desabilitar o armazenamento em camadas. Defina essa política como Excluir para indicar que os dados no armazenamento em camadas são excluídos quando você definir remote.storage.enable como falso.	N/D	DELETE
remote.log.reader.threads	O tamanho do pool de threads do leitor de logs remoto. Usado no agendamento de tarefas para buscar dados do armazenamento remoto.	N/D	max(10, vCPUs*0,67), no qual as vCPUs dependem do tamanho da instância do agente

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
remote.storage.enable	Se definido como verdadeiro, habilita o armazenamento em camadas (remoto) para um tópico. Desabilita o armazenamento em camadas no nível de tópico se definido como falso e se remote.log.msk.disable.policy estiver definido como Excluir. Ao desabilitar o armazenamento em camadas, você exclui dados do armazenamento remoto. Ao desabilitar o armazenamento em camadas para um tópico, não será possível habilitá-lo novamente.	false	verdadeiro

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
replica.lag.time.max.ms	Se um seguidor não enviou nenhuma solicitação de busca ou se não consumiu até o deslocamento final do log do líder por pelo menos esse número de milissegundos, o líder remove o seguidor do ISR.	30000	30000

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
retention.ms	Campo obrigatório. O tempo mínimo é de 3 dias. Não há padrão porque a configuração é obrigatória. O Amazon MSK usa o valor retention.ms com local.retention.ms para determinar quando os dados são movidos do armazenamento local para o armazenamento em camadas. O valor local.retention.ms especifica quando mover dados do armazenamento local para o armazenamento em camadas. O valor retention.ms especifica quando remover dados do armazenamento em camadas (ou seja, remoção do cluster). Valores válidos: números inteiros em [-1; +Inf]	Mínimo de 259.200.000 milissegundos (3 dias). Use -1 para retenção infinita.	Mínimo de 259.200.000 milissegundos (3 dias). Use -1 para retenção infinita.

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
socket.receive.buffer.bytes	O buffer SO_RCVBUF dos soquetes do servidor de soquetes. Se o valor for -1, o sistema operacional padrão será usado.	102400	102400
socket.request.max.bytes	Número máximo de bytes em uma solicitação de soquete.	104857600	104857600
socket.send.buffer.bytes	O buffer SO_SNDBUF dos soquetes do servidor de soquetes. Se o valor for -1, o sistema operacional padrão será usado.	102400	102400
unclean.leader.election.enable	Indica se você deseja que as réplicas que não estão no conjunto ISR devem atuar como líderes em último recurso, mesmo que isso possa resultar em perda de dados.	verdadeiro	false
zookeeper.session.timeout.ms	O tempo limite da ZooKeeper sessão do Apache em milissegundos.	18000	18000

Nome	Descrição	Valor padrão para cluster de armazenamento sem camadas	Valor padrão para cluster de armazenamento em camadas
zookeeper.set.acl	O cliente definido para usar ACLs seguras.	false	false

Para obter mais informações sobre como definir valores de configuração personalizada, consulte the section called “Configurações personalizadas”.

Diretrizes para configuração de armazenamento em camadas no nível de tópico

Veja a seguir as configurações e limitações padrão quando você configura o armazenamento em camadas no nível de tópico.

- O Amazon MSK não é compatível com tamanhos menores de segmentos de log para tópicos com o armazenamento em camadas ativado. Se você quiser criar um segmento, há um tamanho mínimo de segmento de log de 48 MiB ou um tempo mínimo de rolagem do segmento de 10 minutos. Esses valores são mapeados para as propriedades segment.bytes e segment.ms.
- O valor de local.retention.ms/bytes não pode ser igual ou exceder o de retention.ms/bytes. Essa é a configuração de retenção de armazenamento em camadas.
- O valor padrão para local.retention.ms/bytes é -2. Isso significa que o valor retention.ms é usado para local.retention.ms/bytes. Nesse caso, os dados permanecem no armazenamento local e no armazenamento em camadas (uma cópia em cada) e expiram juntos. Para essa opção, uma cópia dos dados locais é mantida no armazenamento remoto. Nesse caso, os dados lidos do tráfego de consumo vêm do armazenamento local.
- O valor padrão para retention.ms é de 7 dias. Não há limite de tamanho padrão para retention.bytes.
- O valor mínimo para retention.ms/bytes é -1. Isso significa retenção infinita.
- O valor mínimo para local.retention.ms/bytes é -2. Isso significa retenção infinita para o armazenamento local. Esse valor corresponde à configuração de retention.ms/bytes como -1.
- A configuração no nível de tópico para retention.ms é obrigatória para tópicos com armazenamento em camadas ativado. O mínimo de retention.ms é de 3 dias.

Operações de configuração do Amazon MSK

Este tópico descreve como criar configurações personalizadas do MSK e como executar operações nelas. Para obter informações sobre como usar configurações do MSK para criar ou atualizar clusters, consulte Como funciona.

Este tópico contém as seguintes seções:

Para criar uma configuração do MSK

1. Crie um arquivo para especificar as propriedades de configuração que você deseja definir e os valores que deseja atribuir a elas. Veja a seguir o conteúdo de um arquivo de configuração de exemplo.

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

log.roll.ms = 604800000
```

2. Execute o AWS CLI comando a seguir e substitua *config-file-path pelo caminho* para o arquivo em que você salvou sua configuração na etapa anterior.

> **Note**
>
> O nome que você escolher para sua configuração deve corresponder ao seguinte regex: "^[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
```

Veja a seguir um exemplo de uma resposta bem-sucedida após a execução desse comando.

```
{
    "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. O comando anterior retorna um nome do recurso da Amazon (ARN) para sua nova configuração. Salve esse ARN porque você precisará dele ao se referir a essa configuração em outros comandos. Se você perder o ARN da configuração, poderá listar todas as configurações da sua conta para encontrá-lo novamente.

Para atualizar uma configuração do MSK

1. Crie um arquivo para especificar as propriedades de configuração que você deseja atualizar e os valores que deseja atribuir a elas. Veja a seguir o conteúdo de um arquivo de configuração de exemplo.

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

min.insync.replicas = 2
```

2. Execute o seguinte comando na AWS CLI , substituindo *config-file-path* pelo caminho para o arquivo no qual você salvou a configuração na etapa anterior.

   Substitua *configuration-arn* pelo ARN obtido ao criar a configuração. Se você não tiver salvado o ARN ao criar a configuração, poderá usar o comando `list-configurations` para listar todas as configurações em sua conta. A configuração que você deseja ver na lista aparecerá na resposta. O ARN da configuração também aparece nessa lista.

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

3. Veja a seguir um exemplo de uma resposta bem-sucedida após a execução desse comando.

```
{
    "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
    }
}
```

Para excluir uma configuração do MSK

O procedimento a seguir mostra como excluir uma configuração que não esteja anexada a um cluster. Não é possível excluir uma configuração anexada a um cluster.

1. Para executar este exemplo, substitua *`configuration-arn`* pelo ARN que você obteve ao criar a configuração. Se você não tiver salvado o ARN ao criar a configuração, poderá usar o comando `list-configurations` para listar todas as configurações em sua conta. A configuração que você deseja ver na lista aparecerá na resposta. O ARN da configuração também aparece nessa lista.

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

2. Veja a seguir um exemplo de uma resposta bem-sucedida após a execução desse comando.

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

Para descrever uma configuração do MSK

1. O seguinte comando retornará metadados sobre a configuração. Para obter uma descrição detalhada da configuração, execute o `describe-configuration-revision`.

Para executar este exemplo, substitua *configuration-arn* pelo ARN que você obteve ao criar a configuração. Se você não tiver salvado o ARN ao criar a configuração, poderá usar o comando `list-configurations` para listar todas as configurações em sua conta. A configuração que você deseja ver na lista aparecerá na resposta. O ARN da configuração também aparece nessa lista.

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

2. Veja a seguir um exemplo de uma resposta bem-sucedida após a execução desse comando.

```
{
    "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"
}
```

Como descrever uma revisão da configuração do MSK

Se você usar o comando `describe-configuration` para descrever uma configuração do MSK, verá os metadados da configuração. Para obter uma descrição da configuração, use o comando `describe-configuration-revision`.

- Execute o seguinte comando, substituindo *configuration-arn* pelo ARN obtido quando você criou a configuração. Se você não tiver salvado o ARN ao criar a configuração, poderá usar o comando `list-configurations` para listar todas as configurações em sua conta. A configuração que você deseja ver na lista aparecerá na resposta. O ARN da configuração também aparece nessa lista.

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

Veja a seguir um exemplo de uma resposta bem-sucedida após a execução desse comando.

```
{
    "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
}
```

O valor de `ServerProperties` é codificado em base64. Se você usar um decodificador em base64 (por exemplo, https://www.base64decode.org/) para decodificá-lo manualmente, obterá o conteúdo do arquivo de configuração original usado para criar a configuração personalizada. Nesse caso, você obtém o seguinte:

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

log.roll.ms = 604800000
```

Como listar todas as configurações do MSK em sua conta para a região atual

- Execute o seguinte comando .

```
aws kafka list-configurations
```

Veja a seguir um exemplo de uma resposta bem-sucedida após a execução desse comando.

```
{
    "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 Serverless

> **Note**
>
> O MSK Serverless está disponível nas regiões Leste dos EUA (Ohio), Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Canadá (Central), Ásia-Pacífico (Mumbai), Ásia-Pacífico (Singapura), Ásia-Pacífico (Sydney), Ásia-Pacífico (Tóquio), Ásia-Pacífico (Seul), Europa (Frankfurt), Europa (Estocolmo) Europa (Irlanda), Europa (Paris) e Europa (Londres).

O MSK Serverless é um tipo de cluster para o Amazon MSK que possibilita que você execute o Apache Kafka sem precisar gerenciar e escalar a capacidade do cluster. Ele provisiona e dimensiona automaticamente a capacidade enquanto gerencia as partições em seu tópico, permitindo que você transmita dados sem pensar em dimensionar ou escalar clusters corretamente. O MSK Serverless oferece um modelo de preço baseado em throughput, para que você pague somente pelo que for usado. Considere usar um cluster com tecnologia sem servidor se suas aplicações precisarem de capacidade de streaming sob demanda que aumente e diminua automaticamente.

O MSK Serverless é totalmente compatível com o Apache Kafka, então você pode usar qualquer aplicação cliente compatível para produzir e consumir dados. Ele também se integra com os seguintes serviços:

- AWS PrivateLink para fornecer conectividade privada
- AWS Identity and Access Management (IAM) para autenticação e autorização usando linguagens Java e não Java. Para obter instruções sobre como configurar clientes para o IAM, consulte Configurar clientes para controle de acesso do IAM.
- AWS Glue Registro de esquemas para gerenciamento de esquemas
- Amazon Managed Service for Apache Flink para processamento de stream com base em Apache Flink
- AWS Lambda para processamento de eventos

> **Note**
>
> O MSK Serverless exige controle de acesso do IAM para todos os clusters. Não há compatibilidade com as listas de controle de acesso (ACLs) do Apache Kafka. Para ter mais informações, consulte the section called “Controle de acesso do IAM”.

Para obter informações sobre cotas de serviço aplicáveis ao MSK Serverless, consulte the section called “Cota para clusters com tecnologia sem servidor”.

Para ajudar você a começar a usar clusters com a tecnologia sem servidor e saber mais sobre as opções de configuração e monitoramento de clusters com a tecnologia sem servidor, consulte o seguinte.

Tópicos

- Conceitos básicos sobre como usar clusters do MSK Serverless
- Configuração para clusters com tecnologia sem servidor
- Monitoramento de clusters com tecnologia sem servidor

Conceitos básicos sobre como usar clusters do MSK Serverless

Este tutorial mostra um exemplo de como você pode criar um cluster do MSK Serverless, criar uma máquina cliente capaz de acessá-lo e usar o cliente para criar tópicos no cluster e gravar dados nesses tópicos. Este exercício não representa todas as opções que você pode escolher ao criar um cluster com a tecnologia sem servidor. Em diferentes partes deste exercício, escolhemos as opções padrão para facilitar. Isso não significa que são as únicas opções que funcionam para configurar um cluster com a tecnologia sem servidor. Você também pode usar a API AWS CLI ou a Amazon MSK. Para obter mais informações, consulte a Referência 2.0 da API do Amazon MSK.

Tópicos

- Etapa 1: criar um cluster do MSK Serverless
- Etapa 2: Criar uma função do IAM
- Etapa 3: criar uma máquina cliente
- Etapa 4: criar um tópico do Apache Kafka
- Etapa 5: produzir e consumir dados
- Etapa 6: excluir recursos

Etapa 1: criar um cluster do MSK Serverless

Nesta etapa, você executará duas tarefas. Primeiro, você cria um cluster do MSK Serverless com as configurações padrão. Em seguida, você reúne informações sobre o cluster. Essas são as

informações que você precisará em etapas posteriores ao criar um cliente capaz de enviar dados para o cluster.

Para criar um cluster com a tecnologia sem servidor

1. Faça login no AWS Management Console e abra o console do Amazon MSK em https://console.aws.amazon.com/msk/home.
2. Selecione Criar cluster.
3. Em Método de criação, deixe a opção Criação rápida selecionada. A opção Criação rápida permite criar um cluster com a tecnologia sem servidor com as configurações padrão.
4. Em Nome do cluster, insira um nome descritivo, como **`msk-serverless-tutorial-cluster`**.
5. Em Propriedades gerais do cluster, escolha Tecnologia sem servidor como o Tipo de cluster. Use os valores padrão para nos itens restantes das Propriedades gerais do cluster.
6. Observe a tabela em Todas as configurações do cluster. Essa tabela lista os valores padrão para configurações importantes, como rede e disponibilidade, e indica se você pode alterar cada configuração depois de criar o cluster. Para alterar uma configuração antes de criar o cluster, você deve escolher a opção Criação personalizada em Método de criação.

   > **Note**
   >
   > Você pode conectar clientes de até cinco VPCs diferentes com clusters MSK Serverless. Para ajudar as aplicações clientes a migrarem para outra zona de disponibilidade no caso de uma interrupção, você deve especificar pelo menos duas sub-redes em cada VPC.

7. Selecione Criar cluster.

Para reunir informações sobre o cluster

1. Na seção Resumo do cluster, escolha Exibir informações do cliente. Esse botão permanece esmaecido até que o Amazon MSK conclua a criação do cluster. Pode ser necessário esperar alguns minutos até o botão ficar ativo para poder usá-lo.
2. Copie a string sob o rótulo Endpoint. Essa é a string do seu servidor bootstrap.
3. Escolha a guia Properties (Propriedades).

4. Na seção Configurações de rede, copie os IDs das sub-redes e do grupo de segurança e salve-os, pois você precisará dessas informações posteriormente para criar uma máquina cliente.
5. Escolha qualquer uma das sub-redes. Isso abrirá o console da Amazon VPC. Localize o ID da Amazon VPC associada à sub-rede. Salve esse ID da Amazon VPC para uso posterior.

Próxima etapa

Etapa 2: Criar uma função do IAM

Etapa 2: Criar uma função do IAM

Nesta etapa, você executará duas tarefas. A primeira tarefa será a criação de uma política do IAM que conceda acesso para criar tópicos no cluster e enviar dados para esses tópicos. A segunda tarefa será a criação de um perfil do IAM e a associação dessa política a ele. Em uma etapa posterior, criaremos uma máquina cliente que vai assumir esse perfil e usá-lo para criar um tópico no cluster e enviar dados para esse tópico.

Para criar uma política do IAM que permita criar tópicos e gravar neles

1. Abra o console IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Políticas.
3. Escolha Create Policy.
4. Escolha a guia JSON e substitua o JSON na janela do editor com o JSON a seguir.

   Substitua *região* pelo código da Região da AWS na qual você criou o cluster. Substitua *Account-ID* pelo seu ID de conta. *msk-serverless-tutorial-cluster*Substitua pelo nome do seu cluster sem servidor.

```
{
    "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/*"
            ]
        }
    ]
}
```

Para obter instruções sobre como criar políticas de seguras, consulte the section called “Controle de acesso do IAM”.

5. Escolha Próximo: etiquetas.
6. Selecione Next: Review (Próximo: revisar).
7. Em nome da política, insira um nome descritivo, como **`msk-serverless-tutorial-policy`**.
8. Escolha Criar política.

Para criar um perfil do IAM e associar a política a ele

1. No painel de navegação, escolha Perfis.

2. Selecione Criar função.
3. Em Casos de uso comuns, selecione EC2 e então Próximo: permissões.
4. Na caixa de pesquisa, insira o nome da política que você criou anteriormente para este tutorial. Em seguida, marque a caixa à esquerda da política.
5. Escolha Próximo: etiquetas.
6. Selecione Next: Review (Próximo: revisar).
7. Em nome do perfil, insira um nome descritivo, como **`msk-serverless-tutorial-role`**.
8. Selecione Criar função.

Próxima etapa

Etapa 3: criar uma máquina cliente

Etapa 3: criar uma máquina cliente

Nesta etapa, você executará duas tarefas. A primeira tarefa é criar uma instância do Amazon EC2 para usar como uma máquina cliente do Apache Kafka. A segunda tarefa é instalar as ferramentas Java e Apache Kafka na máquina.

Como criar uma máquina cliente

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Escolha Iniciar instância.
3. Insira um Nome descritivo para sua máquina cliente, como **`msk-serverless-tutorial-client`**.
4. Deixe a opção AMI do Amazon Linux 2 (HVM) – Kernel 5.10, tipo de volume SSD selecionada para Tipo de imagem de máquina da Amazon (AMI).
5. Deixe o tipo de instância t2.micro selecionado.
6. Na seção Par de chaves, escolha Criar um novo par de chaves. Insira **`MSKServerlessKeyPair`** para Nome do par de chaves. Em seguida, escolha Baixar o par de chaves. Se preferir, use um par de chaves existente.
7. Em Configurações de rede, escolha Editar.
8. Em VPC, insira o ID da nuvem privada virtual (VPC) para o seu cluster com a tecnologia sem servidor. Trata-se da VPC baseada no serviço da Amazon VPC, cujo ID você salvou após a criação do cluster.

9. Em Sub-rede, escolha a sub-rede cujo ID você salvou depois de criar o cluster.
10. Em Firewall (grupos de segurança), selecione o grupo de segurança associado ao cluster. Esse valor funcionará se esse grupo de segurança tiver uma regra de entrada que permita tráfego do grupo de segurança para ele. Com essa regra, os membros do mesmo grupo de segurança podem se comunicar entre eles. Para obter mais informações, consulte Regras de grupos de segurança no Guia do desenvolvedor da Amazon VPC.
11. Expanda a seção Detalhes avançados e escolha o perfil do IAM que você criou na Etapa 2: Criar uma função do IAM.
12. Escolha Executar.
13. No painel de navegação à esquerda, escolha Instances (Instâncias). Em seguida, escolha a caixa de seleção na linha que representa sua instância recém-criada do Amazon EC2. Deste ponto em diante, chamamos essa instância de máquina cliente.
14. Escolha Conectar e siga as instruções para se conectar à máquina cliente.

Para configurar as ferramentas do cliente Apache Kafka na máquina cliente

1. Para instalar o Java, execute o seguinte comando na máquina cliente:

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

2. Para obter as ferramentas do Apache Kafka necessárias para criar tópicos e enviar dados, execute os seguintes comandos:

```
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. Acesse o diretório `kafka_2.12-2.8.1/libs` e execute o seguinte comando para baixar o arquivo JAR do IAM do Amazon MSK. O JAR do IAM do Amazon MSK permite que a máquina cliente acesse o 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. Acesse o diretório `kafka_2.12-2.8.1/bin`. Copie e cole as seguintes configurações de propriedade em um novo arquivo. Nomeie e salve o arquivo como `client.properties`.

```
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
```

Próxima etapa

Etapa 4: criar um tópico do Apache Kafka

Etapa 4: criar um tópico do Apache Kafka

Nesta etapa, você usa a máquina cliente criada anteriormente para criar um tópico no cluster com tecnologia sem servidor.

Para criar um tópico e gravar dados nele

1. No comando `export` a seguir, substitua *`my-endpoint`* pela string do servidor de bootstrap que você salvou depois de criar o cluster. Em seguida, acesse o diretório `kafka_2.12-2.8.1/bin` na máquina cliente e execute o comando `export`.

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

2. Execute o comando a seguir para criar um tópico chamado `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
   ```

Próxima etapa

Etapa 5: produzir e consumir dados

Etapa 5: produzir e consumir dados

Nesta etapa, você produz e consome dados usando o tópico que criou na etapa anterior.

Como produzir e consumir mensagens

1. Execute o comando a seguir para criar um produtor de console.

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

2. Insira a mensagem que desejar e pressione Enter. Repita esta etapa duas ou três vezes. Sempre que você inserir uma linha e pressionar Enter, essa linha será enviada para o cluster como uma mensagem separada.
3. Mantenha a conexão com a máquina cliente aberta e abra uma segunda conexão separada com esse computador em uma nova janela.
4. Use sua segunda conexão com a máquina cliente para criar um consumidor no console executando o comando a seguir. Substitua *my-endpoint* pela string do servidor de bootstrap que você salvou após criar o 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
   ```

   Você começará a ver as mensagens inseridas anteriormente quando usou o comando do produtor do console.
5. Insira mais mensagens na janela do produtor e observe-as aparecerem na janela do consumidor.

Próxima etapa

Etapa 6: excluir recursos

Etapa 6: excluir recursos

Nesta etapa, você excluirá os recursos que criou neste tutorial.

Para excluir o cluster

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/home.
2. Na lista de clusters, escolha o cluster que criou para este tutorial.
3. Em Ações, escolha Excluir cluster.
4. Insira `delete` no campo e escolha Excluir.

Para interromper a máquina cliente

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Na lista de instâncias do Amazon EC2, escolha a máquina cliente que você criou para este tutorial.
3. Escolha Estado da instância e Encerrar instância.
4. Escolha Encerrar.

Para excluir a política e o perfil do IAM

1. Abra o console IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Perfis.
3. Na caixa de pesquisa, insira o nome do perfil do IAM que você criou para este tutorial.
4. Selecione o perfil de . Escolha Excluir perfil e confirme a exclusão.
5. No painel de navegação, escolha Políticas.
6. Na caixa de pesquisa, insira o nome da política que você criou para este tutorial.
7. Escolha a política para abrir a respectiva página de resumo. Na página Resumo da política, escolha Editar política.
8. Escolha Excluir.

Configuração para clusters com tecnologia sem servidor

O Amazon MSK define propriedades de configuração do agente para clusters com tecnologia sem servidor. Você não pode alterar essas configurações de propriedades de configuração do agente. Porém, é possível definir as seguintes propriedades de configuração de tópico.

Propriedade de configuração	Padrão	Editável	Valor máximo permitido
cleanup.policy	Delete	Sim, mas somente no momento da criação do tópico	
compression.type	Produtor	Sim	

Propriedade de configuração	Padrão	Editável	Valor máximo permitido
max.message.bytes	104858	Sim	8 MiB
message.timestamp.difference.max.ms	long.max	Sim	
message.timestamp.type	CreateTime	Sim	
retention.bytes	250 GiB	Sim	250 GiB
retention.ms	7 dias	Sim	Ilimitado

Você também pode usar os comandos do Apache Kafka para definir ou modificar propriedades de configuração no nível de tópico para tópicos novos ou existentes. Para obter mais informações sobre as propriedades de configuração no nível de tópico e exemplos de como defini-las, consulte Configurações no nível de tópico na documentação oficial do Apache Kafka.

Monitoramento de clusters com tecnologia sem servidor

O Amazon MSK se integra à Amazon CloudWatch para que você possa coletar, visualizar e analisar métricas para seu cluster MSK Serverless. As métricas mostradas na tabela a seguir estão disponíveis para todos os clusters com tecnologia sem servidor. Como essas métricas são publicadas como pontos de dados individuais para cada partição no tópico, recomendamos visualizá-las como uma estatística “SUM” a fim de obter a visualização no nível de tópico.

O Amazon MSK publica `PerSec` métricas com CloudWatch uma frequência de uma vez por minuto. Isso significa que a estatística “SUM” para um período de um minuto representa com precisão os dados por segundo para métricas `PerSec`. Para coletar dados por segundo por um período superior a um minuto, use a seguinte expressão CloudWatch matemática:`m1 * 60/PERIOD(m1)`.

Métricas disponíveis no nível de monitoramento DEFAULT

Nome	Quando visível	Dimensões	Descrição
`BytesInPerSec`	Após um produtor gravar em um tópico	Nome do cluster, tópico	O número de bytes por segundo recebidos dos clientes. Essa métrica está disponível para cada tópico.
`BytesOutPerSec`	Após um grupo de consumidores consumir de um tópico	Nome do cluster, tópico	O número de bytes por segundo enviados aos clientes. Essa métrica está disponível para cada tópico.
`FetchMessageConversionsPerSec`	Após um grupo de consumidores consumir de um tópico	Nome do cluster, tópico	O número de conversões de mensagens de busca por segundo para o tópico.
`EstimatedMaxTimeLag`	Após um grupo de consumidores consumir de um tópico	Nome do cluster, grupo de consumidores, tópico	Uma estimativa de tempo da MaxOffsetLag métrica.
`MaxOffsetLag`	Após um grupo de consumidores consumir de um tópico	Nome do cluster, grupo de consumidores, tópico	O atraso máximo de deslocamento entre todas as partições em um tópico.
`MessagesInPerSec`	Após um produtor gravar em um tópico	Nome do cluster, tópico	O número de mensagens recebidas por segundo para o tópico.
`ProduceMessageConversionsPerSec`	Após um produtor gravar em um tópico	Nome do cluster, tópico	O número de conversões de mensagens de produção por segundo para o tópico.
`SumOffsetLag`	Após um grupo de consumido	Nome do cluster, grupo	O atraso de deslocamento agregado para todas as partições em um tópico.

Nome	Quando visível	Dimensões	Descrição
	res consumir de um tópico	de consumidores, tópico	

Para visualizar as métricas do MSK Serverless

1. Faça login no AWS Management Console e abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, em Métricas, escolha Todas as métricas.
3. Nas métricas, pesquise o termo **`kafka`**.
4. Escolha AWS/Kafka/Nome do cluster, tópico ou AWS/Kafka/Nome do cluster, grupo de consumidores, tópico para ver métricas diferentes.

MSK Connect

O que é o MSK Connect?

O MSK Connect é um recurso do Amazon MSK que facilita o streaming de dados de e para os clusters do Apache Kafka. O MSK Connect usa o Kafka Connect 2.7.1, uma estrutura de código aberto para conectar clusters do Apache Kafka a sistemas externos, como bancos de dados, índices de pesquisa e sistemas de arquivos. Com o MSK Connect, você pode implantar conectores totalmente gerenciados criados para o Kafka Connect que movem dados para ou extraem dados de datastores populares, como Amazon S3 e Amazon Service. OpenSearch Você pode implantar conectores desenvolvidos por terceiros, como o Debezium, para transmitir logs de alterações de bancos de dados para um cluster do Apache Kafka ou implantar um conector existente sem alterações no código. Os conectores escalam automaticamente para se ajustar às mudanças na carga, e você paga apenas pelos recursos que usa.

Use conectores de origem para importar dados de sistemas externos para seus tópicos. Com conectores de coletor, você pode exportar dados de seus tópicos para sistemas externos.

O MSK Connect é compatível com conectores para qualquer cluster do Apache Kafka com conectividade com uma Amazon VPC, seja um cluster do MSK ou um cluster do Apache Kafka hospedado de maneira independente.

O MSK Connect monitora continuamente a integridade e o estado de entrega dos conectores, corrige e gerencia o hardware subjacente e dimensiona automaticamente a escala dos conectores para corresponder às mudanças no throughput.

Para começar a usar o MSK Connect, consulte the section called “Conceitos básicos”.

Para saber mais sobre os AWS recursos que você pode criar com o MSK Connectthe section called “Connectors”, consultethe section called “Plug-ins”, e. the section called “Operadores”

Para obter informações sobre a API do MSK Connect, consulte a Referência de API do Amazon MSK Connect.

Conceitos básicos sobre como usar o MSK Connect

Este é um step-by-step tutorial que usa o AWS Management Console para criar um cluster MSK e um conector de coletor que envia dados do cluster para um bucket S3.

Tópicos

Etapa 1: configurar os recursos necessários

Nesta etapa, você cria os seguintes recursos necessários para esse cenário inicial:

- Um bucket do S3 para servir como destino que recebe dados do conector.
- Um cluster do MSK para o qual você enviará dados. Em seguida, o conector lerá os dados desse cluster e os enviará para o bucket S3 de destino.
- Uma perfil do IAM que permite ao conector gravar no bucket do S3 de destino.
- Um endpoint da Amazon VPC para possibilitar o envio de dados da Amazon VPC que tem o cluster e o conector para o Amazon S3.

Para criar um bucket do S3

1. Faça login AWS Management Console e abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.
2. Selecione Criar bucket.
3. Para o nome do bucket, insira um nome descritivo, como `mkc-tutorial-destination-bucket`.
4. Role para baixo e escolha Criar bucket.
5. Na lista de buckets, escolha o bucket recém-criado.
6. Selecione Criar pasta.
7. Digite `tutorial` para o nome da pasta, depois role para baixo e escolha Criar pasta.

Para criar um cluster

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.

2. No painel esquerdo, em Clusters do MSK, escolha Clusters.
3. Selecione Criar cluster.
4. Escolha Criação personalizada.
5. Insira `mkc-tutorial-cluster` para o nome do cluster.
6. Em Propriedades gerais do cluster, escolha Provisionado como o tipo de cluster.
7. Em Rede, escolha uma Amazon VPC. Em seguida, selecione as zonas de disponibilidade e as sub-redes que deseja usar. Lembre-se dos IDs da Amazon VPC e das sub-redes que você selecionou, pois precisará deles posteriormente neste tutorial.
8. Em Métodos de controle de acesso, verifique se somente o Acesso não autenticado está selecionado.
9. Em Criptografia, certifique-se de que somente Texto simples esteja selecionado.
10. Continue com o assistente e escolha Criar cluster. Você será redirecionado para a página detalhes do cluster. Nessa página, em Grupos de segurança aplicados, encontre o ID do grupo de segurança. Lembre-se desse ID porque você precisará dele posteriormente neste tutorial.

Para criar o perfil do IAM capaz de gravar no bucket de destino

1. Abra o console IAM em https://console.aws.amazon.com/iam/.
2. No painel esquerdo, em Gerenciamento de acesso, escolha Perfis.
3. Selecione Criar função.
4. Em Ou selecione um serviço para visualizar seus casos de uso, escolha S3.
5. Role para baixo e, em Selecione seu caso de uso, escolha S3 novamente.
6. Escolha Next: Permissions (Próximo: permissões).
7. Escolha Criar política. Isso abrirá uma nova guia no seu navegador, na qual você criará a política. Deixe a guia de criação de perfil original aberta, pois retornaremos a ela mais tarde.
8. Escolha a guia JSON e substitua o texto na janela com a política a seguir.

```
{
  "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. Escolha Próximo: etiquetas.
10. Selecione Next: Review (Próximo: revisar).
11. Insira `mkc-tutorial-policy` para o nome da política e escolha Criar política.
12. Escolha o botão Atualizar na guia do navegador em que você estava criando o perfil.
13. Encontre a `mkc-tutorial-policy` e selecione-a escolhendo o botão à esquerda.
14. Escolha Próximo: etiquetas.
15. Selecione Next: Review (Próximo: revisar).
16. Insira `mkc-tutorial-role` para o nome do perfil e exclua o texto na caixa de descrição.
17. Selecione Criar função.

Para permitir que o MSK Connect assuma o perfil

1. No console do IAM, em Gerenciamento de acesso no painel esquerdo, escolha Perfis.

2. Encontre e escolha o `mkc-tutorial-role`.
3. Na página Resumo do perfil, escolha a guia Relações de confiança.
4. Selecione Edit trust relationship (Editar relação de confiança).
5. Substitua a política de confiança existente pelo seguinte JSON.

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

6. Escolha Update Trust Policy.

Para criar um endpoint da Amazon VPC da VPC do cluster para o Amazon S3

1. Abra o console do Amazon VPC em https://console.aws.amazon.com/vpc/.
2. No painel esquerdo, escolha Endpoints.
3. Escolha Criar endpoint.
4. Em Nome do serviço, escolha o serviço com.amazonaws.us-east-1.s3 e o tipo Gateway.
5. Escolha a VPC do cluster e, em seguida, selecione a caixa à esquerda da tabela de rotas associada às sub-redes do cluster.
6. Escolha Criar endpoint.

Próxima etapa

Etapa 2: criar um plug-in personalizado

Etapa 2: criar um plug-in personalizado

Um plug-in contém o código que define a lógica do conector. Nesta etapa, você criará um plug-in personalizado contendo o código para o Lenses Amazon S3 Sink Connector. Em uma etapa

posterior, ao criar o conector do MSK, você especificará que seu código está nesse plug-in personalizado. Você pode usar o mesmo plug-in para criar vários conectores do MSK com configurações diferentes.

Para criar o plug-in personalizado

1. Baixe o conector do S3.
2. Faça upload do arquivo ZIP para um bucket do S3 ao qual você tenha acesso. Para obter informações sobre como fazer upload de arquivos para o Amazon S3, consulte Carregar objetos no Guia do usuário do Amazon S3.
3. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
4. No painel esquerdo, expanda MSK Connect e escolha Plug-ins personalizados.
5. Escolha Criar plug-in personalizado.
6. Selecione Browse S3 (Navegar no S3).
7. Na lista de buckets, encontre o bucket no qual você fez o upload do arquivo ZIP e escolha-o.
8. Na lista de objetos no bucket, marque o botão de seleção à esquerda do arquivo ZIP e selecione o botão Escolher.
9. Insira `mkc-tutorial-plugin` para o nome do plug-in personalizado e escolha Criar plug-in personalizado.

Pode levar AWS alguns minutos para concluir a criação do plug-in personalizado. Quando o processo de criação estiver concluído, você verá a seguinte mensagem em um banner na parte superior da janela do navegador.

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

Próxima etapa

Etapa 3: criar a máquina cliente e o tópico do Apache Kafka

Etapa 3: criar a máquina cliente e o tópico do Apache Kafka

Nesta etapa, você vai criar uma instância do Amazon EC2 para usar como uma instância do cliente do Apache Kafka. Em seguida, você usará essa instância para criar um tópico no cluster.

Como criar uma máquina cliente

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Selecione Iniciar instâncias.
3. Insira um Nome para sua máquina cliente, como **`mkc-tutorial-client`**.
4. Deixe a opção AMI do Amazon Linux 2 (HVM) – Kernel 5.10, tipo de volume SSD selecionada para Tipo de imagem de máquina da Amazon (AMI).
5. Escolha o tipo de instância t2.xlarge.
6. Na seção Par de chaves, escolha Criar um novo par de chaves. Digite **`mkc-tutorial-key-pair`** em Nome do par de chaves e, em seguida, escolha Baixar par de chaves. Se preferir, use um par de chaves existente.
7. Escolha Iniciar instância.
8. Escolha View Instances (Exibir instâncias). Na coluna Grupos de segurança, escolha o grupo de segurança que está associado à sua nova instância. Copie o ID do grupo de segurança e salve-o para usar posteriormente.

Para permitir que o cliente recém-criado envie dados para o cluster

1. Abra o console do Amazon VPC em https://console.aws.amazon.com/vpc/.
2. No painel esquerdo, em SEGURANÇA, escolha Grupos de segurança). Na coluna ID do grupo de segurança, localize o grupo de segurança do cluster. Você salvou o ID desse grupo de segurança ao criar o cluster em the section called “Etapa 1: configurar os recursos necessários”. Escolha esse grupo de segurança marcando a caixa à esquerda de sua linha. Certifique-se de que nenhum outro grupo de segurança seja selecionado simultaneamente.
3. Na metade inferior da tela, escolha a guia Regras de entrada.
4. Escolha Editar regras de entrada.
5. Na parte inferior esquerda da tela, escolha Adicionar regra.
6. Na nova regra, escolha All traffic (Todo o tráfego) na coluna Type (Tipo). No campo à direita da coluna Origem, insira o ID do grupo de segurança da máquina cliente. Trata-se do ID do grupo de segurança que você salvou após criar a máquina cliente.
7. Escolha Salvar regras. Agora, seu cluster do MSK aceitará todo o tráfego do cliente criado no procedimento anterior.

Para criar um tópico

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Na tabela de instâncias, escolha `mkc-tutorial-client`.
3. Na parte superior da tela, escolha Connect e siga as instruções para se conectar à instância.
4. Instale o Java na instância do cliente executando o seguinte comando:

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

5. Execute o comando a seguir para fazer download do Apache Kafka.

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

> **Note**
>
> Se quiser usar um local de espelhamento diferente do usado neste comando, você poderá escolher um local diferente no site do Apache.

6. Execute o comando a seguir no diretório onde você fez download do arquivo TAR na etapa anterior.

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

7. Acesse o diretório kafka_2.12-2.2.1.
8. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
9. No painel esquerdo, escolha Clusters e, em seguida, escolha o nome `mkc-tutorial-cluster`.
10. Escolha Exibir informações do cliente.
11. Copie a string de conexão em texto simples.
12. Selecione Done (Concluído).
13. Execute o comando a seguir na instância do cliente (`mkc-tutorial-client`), *`bootstrapServerString`*substituindo-o pelo valor que você salvou ao visualizar as informações do cliente do cluster.

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

Se o comando tiver êxito, a seguinte mensagem será exibida: `Created topic mkc-tutorial-topic.`

Próxima etapa

Etapa 4: criar conector

Etapa 4: criar conector

Para criar o conector

1. Faça login no AWS Management Console e abra o console do Amazon MSK em https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. No painel esquerdo, expanda MSK Connect e escolha Conectores.
3. Escolha Criar conector.
4. Na lista de plug-ins, escolha `mkc-tutorial-plugin` e escolha Próximo.
5. Para o nome do conector, insira `mkc-tutorial-connector`.
6. Na lista de clusters, escolha `mkc-tutorial-cluster`.
7. Copie a seguinte configuração e cole no campo de configuração do conector.

```
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. Em Permissões de acesso, escolha `mkc-tutorial-role`.

9. Escolha Próximo. Na página Segurança, escolha Próximo novamente.
10. Na página Logs, escolha Próximo.
11. Em Revisar e criar, escolha Criar conector.

Próxima etapa

Etapa 5: enviar dados

Etapa 5: enviar dados

Nesta etapa, você envia dados para o tópico do Apache Kafka que você criou anteriormente e, em seguida, procura esses mesmos dados no bucket do S3 de destino.

Para enviar dados para o cluster do MSK

1. Na pasta `bin` da instalação do Apache Kafka na instância do cliente, crie um arquivo de texto chamado `client.properties` com o conteúdo a seguir.

   ```
   security.protocol=PLAINTEXT
   ```

2. Execute o comando a seguir para criar um produtor de console. *`BootstrapBrokerString`*Substitua pelo valor obtido ao executar o comando anterior.

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

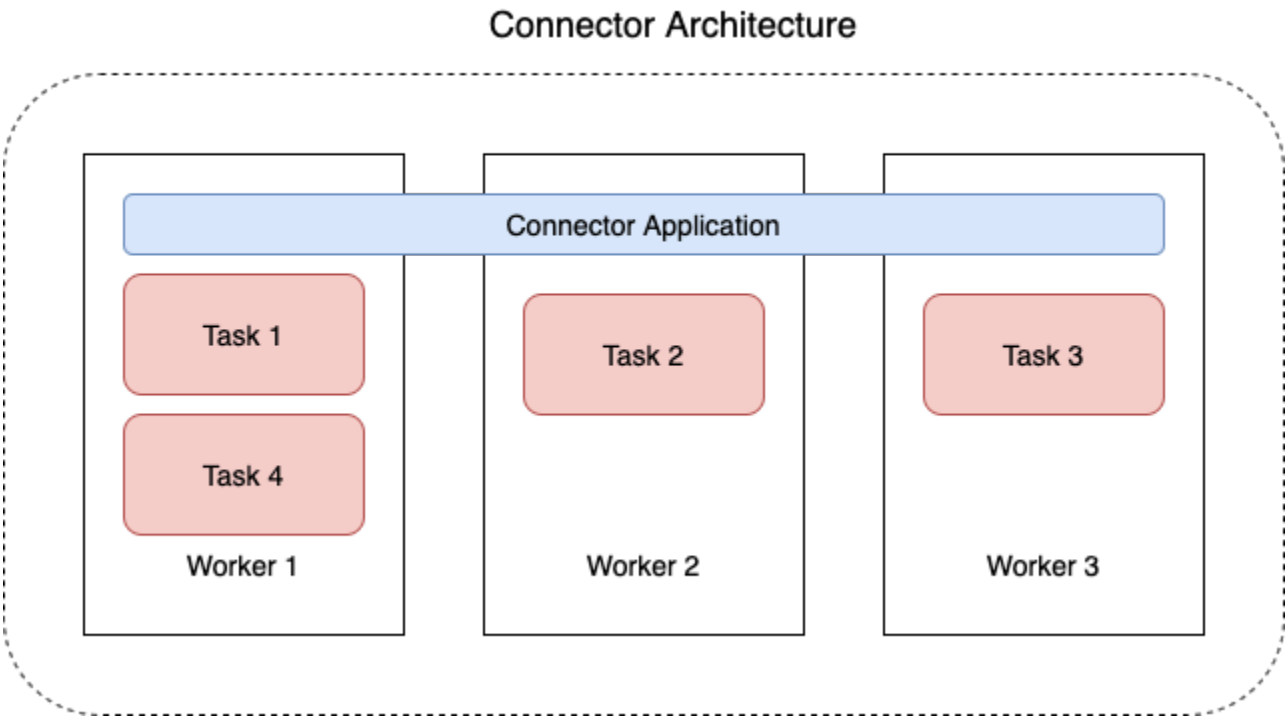
3. Insira a mensagem que desejar e pressione Enter. Repita esta etapa duas ou três vezes. Toda vez que você inserir uma linha e pressionar Enter, essa linha será enviada para o cluster do Apache Kafka como uma mensagem separada.
4. Verifique o bucket do Amazon S3 de destino para encontrar as mensagens que você enviou na etapa anterior.

Connectors

Um conector integra sistemas externos e serviços da Amazon ao Apache Kafka, copiando continuamente dados de streaming de uma fonte de dados para o cluster do Apache Kafka ou copiando continuamente os dados do cluster para um coletor de dados. Antes de entregar os dados

a um destino, um conector também pode executar uma lógica leve, como transformação, conversão de formato ou filtragem de dados. Os conectores de origem extraem dados de uma fonte de dados e os enviam para o cluster, enquanto os conectores coletam dados do cluster e os enviam para um coletor de dados.

O diagrama a seguir mostra a arquitetura de um conector. Um operador é um processo de máquina virtual Java (JVM) que executa a lógica do conector. Cada operador cria um conjunto de tarefas que são executadas em threads paralelos e fazem o trabalho de copiar os dados. As tarefas não armazenam o estado e, portanto, podem ser iniciadas, interrompidas ou reiniciadas a qualquer momento para fornecer um pipeline de dados resiliente e escalável.



Capacidade do conector

A capacidade total de um conector depende do número de operadores que o conector tem, bem como do número de MSK Connect Units (MCUs – Unidades do MSK Connect) por operador. Cada MCU representa 1 vCPU de computação e 4 GiB de memória. A memória da MCU pertence à memória total de uma instância de trabalho e não à memória de pilha em uso.

Os funcionários do MSK Connect consomem endereços IP nas sub-redes fornecidas pelo cliente. Cada trabalhador usa um endereço IP de uma das sub-redes fornecidas pelo cliente. Você deve garantir que tenha endereços IP disponíveis suficientes nas sub-redes fornecidas a uma

CreateConnector solicitação para considerar a capacidade especificada, especialmente ao escalar automaticamente conectores em que o número de trabalhadores pode flutuar.

Para criar um conector, você deve escolher entre um dos dois modos de capacidade a seguir.

- Provisionado: escolha esse modo se você conhecer os requisitos de capacidade do seu conector. Você especifica dois valores:
  - O número de operadores.
  - O número de MCUs por operador.
- Escalonamento automático: escolha esse modo se os requisitos de capacidade do seu conector forem variáveis ou se você não os conhecer com antecedência. Quando você usa o modo de escalabilidade automática, o Amazon MSK Connect substitui a propriedade `tasks.max` do seu conector por um valor proporcional ao número de operadores em execução no conector e ao número de MCUs por operador.

  Você especifica três conjuntos de valores:
  - O número mínimo e máximo de operadores.
  - Os percentuais de expansão e de redução da utilização da CPU, que são determinados pela métrica. `CpuUtilization` Quando a métrica `CpuUtilization` do conector excede o percentual de expansão, o MSK Connect aumenta o número de operadores em execução no conector. Quando a métrica `CpuUtilization` fica abaixo do percentual de expansão, o MSK Connect diminui o número de operadores. O número de operadores sempre permanece dentro dos números mínimo e máximo que você especifica ao criar o conector.
  - O número de MCUs por operador.

Para obter mais informações sobre operadores, consulte the section called “Operadores”. Para saber mais sobre as métricas do MSK Connect, consulte the section called “Monitoramento”.

Como criar um conector

Criando um conector usando o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. No painel esquerdo, em MSK Connect, escolha Conectores.
3. Escolha Criar conector.

4. Você pode escolher entre usar um plug-in personalizado existente para criar o conector ou criar primeiro um novo plug-in personalizado. Para obter informações sobre plug-ins personalizados e como criá-los, consulte the section called “Plug-ins”. Neste procedimento, vamos supor que você tenha um plug-in personalizado que deseja usar. Na lista de plug-ins personalizados, encontre o que você deseja usar, marque a caixa à esquerda e escolha Próximo.
5. Insira um nome e, se desejar, uma descrição.
6. Escolha o cluster ao qual deseja se conectar.
7. Especifique a configuração do conector. Os parâmetros de configuração que você precisa especificar dependerão do tipo de conector que você deseja criar. No entanto, alguns parâmetros são comuns a todos os conectores, por exemplo, os parâmetros `connector.class` e `tasks.max`. Veja a seguir um exemplo de configuração para o Conector de coletor Confluent para Amazon S3.

```
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. Em seguida, configure a capacidade do conector. Você pode escolher entre dois modos de capacidade: provisionado e escalonado automaticamente. Para obter informações sobre essas duas opções, consulte the section called “Capacity”.
9. Escolha a configuração padrão do operador ou uma configuração personalizada do operador. Para obter informações sobre como criar configurações personalizadas de operador, consulte the section called “Operadores”.
10. Em seguida, você especifica o perfil de execução do serviço. Essa deve ser uma função do IAM que o MSK Connect possa assumir e que conceda ao conector todas as permissões necessárias para acessar os AWS recursos necessários. Essas permissões dependem da lógica do conector. Para obter informações sobre como criar essa função, consulte the section called “Perfil de execução do serviço”.
11. Escolha Próximo, revise as informações de segurança e escolha Próximo novamente.

12. Especifique as opções de registro em log que deseja e escolha Próximo. Para obter informações sobre registro em log, consulte the section called “Registro em log”.
13. Escolha Criar conector.

Para usar a API MSK Connect para criar um conector, consulte CreateConnector.

Plug-ins

Um plug-in é um AWS recurso que contém o código que define a lógica do conector. Você carrega um arquivo JAR (ou um arquivo ZIP contendo um ou mais arquivos JAR) em um bucket do S3 e especifica a localização do bucket ao criar o plug-in. Ao criar um conector, você especifica o plug-in que deseja que o MSK Connect use para ele. A relação dos plug-ins com os conectores é one-to-many: Você pode criar um ou mais conectores do mesmo plug-in.

Para obter informações sobre como desenvolver o código para um conector, consulte o Guia de desenvolvimento de conectores na documentação do Apache Kafka.

Criando um plug-in personalizado usando o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. No painel esquerdo, em MSK Connect e escolha Plug-ins personalizados.
3. Escolha Criar plug-in personalizado.
4. Selecione Browse S3 (Navegar no S3).
5. Na lista de buckets do S3, escolha o bucket que contém o arquivo JAR ou ZIP do plug-in.
6. Na lista de objetos, marque a caixa à esquerda do arquivo JAR ou ZIP do plug-in e clique em Escolher.
7. Escolha Criar plug-in personalizado.

Para usar a API MSK Connect para criar um plug-in personalizado, consulte CreateCustomPlugin.

Operadores

Um operador é um processo de máquina virtual Java (JVM) que executa a lógica do conector. Cada operador cria um conjunto de tarefas que são executadas em threads paralelos e fazem o trabalho de copiar os dados. As tarefas não armazenam o estado e, portanto, podem ser iniciadas,

interrompidas ou reiniciadas a qualquer momento para fornecer um pipeline de dados resiliente e escalável. Alterações no número de operadores, seja devido a um evento de escalonamento ou devido a falhas inesperadas, são detectadas automaticamente pelos demais operadores. Eles se organizam para reequilibrar as tarefas no conjunto de operadores restantes. Os operadores do Connect usam os grupos de consumidores do Apache Kafka para coordenar e reequilibrar.

Se os requisitos de capacidade do seu conector forem variáveis ou difíceis de estimar, você pode deixar o MSK Connect escalar o número de operadores conforme necessário entre um limite inferior e um limite superior que você determina. Como alternativa, você pode especificar o número exato de operadores que deseja executar em sua lógica de conector. Para ter mais informações, consulte the section called “Capacity”.

Os trabalhadores do MSK Connect consomem endereços IP

Os funcionários do MSK Connect consomem endereços IP nas sub-redes fornecidas pelo cliente. Cada trabalhador usa um endereço IP de uma das sub-redes fornecidas pelo cliente. Você deve garantir que tenha endereços IP disponíveis suficientes nas sub-redes fornecidas a uma CreateConnector solicitação para considerar a capacidade especificada, especialmente ao escalar automaticamente conectores em que o número de trabalhadores pode flutuar.

Tópicos

Configuração padrão de operador

O MSK Connect fornece a seguinte configuração padrão de operador:

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

Propriedades de configuração de operador compatíveis

O MSK Connect fornece uma configuração padrão de operador. Também há a opção de criar uma configuração personalizada de operador para usar com os conectores. A lista a seguir inclui

informações sobre as propriedades de configuração do operador compatíveis ou não com o Amazon MSK Connect.

- As propriedades `key.converter` e `value.converter` são obrigatórias.
- O MSK Connect é compatível com as seguintes propriedades de configuração de `producer.`.

```
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
```

- O MSK Connect é compatível com as seguintes propriedades de configuração de `consumer.`.

```
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
```

- Todas as outras propriedades de configuração que não comecem com os prefixos `producer.` ou `consumer.` são compatíveis, exceto as propriedades a seguir.

```
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
```

Para obter mais informações sobre as propriedades de configuração do operador e o que elas representam, consulte Configurações do Kafka para o Connect na documentação do Apache Kafka.

Criação de uma configuração personalizada de operador

Criando uma configuração de trabalhador personalizada usando o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. No painel esquerdo, em MSK Connect, escolha Configurações do operador.
3. Escolha Criar configuração de operador.
4. Insira um nome e uma descrição opcional e, em seguida, adicione as propriedades e os valores para os quais você deseja defini-los.
5. Escolha Criar configuração de operador.

Para usar a API MSK Connect para criar uma configuração de trabalho, consulte CreateWorkerConfiguration.

Gerenciamento de deslocamentos do conector de origem usando `offset.storage.topic`

Esta seção fornece informações para ajudar você a gerenciar os deslocamentos do conector de origem usando o tópico de deslocamento de armazenamento. O tópico de deslocamento de armazenamento é um tópico interno que o Kafka Connect usa para armazenar deslocamentos de configuração de conectores e tarefas.

Como usar o tópico padrão de deslocamento de armazenamento

Por padrão, o Amazon MSK Connect gera um novo tópico de deslocamento de armazenamento em seu cluster do Kafka para cada conector que você cria. O MSK estrutura o nome do tópico padrão usando partes do ARN do conector. Por exemplo, `__amazon_msk_connect_offsets_my-mskc-connector_12345678-09e7-4abc-8be8-c657f7e4ff32-2`.

Como especificar seu próprio tópico de deslocamento de armazenamento

Para fornecer continuidade de deslocamento entre conectores de origem, você pode usar um tópico de deslocamento de armazenamento de sua escolha em vez do tópico padrão. Especificar um tópico de deslocamento de armazenamento ajuda você a realizar tarefas como criar um conector de origem que retoma a leitura desde o último deslocamento de um conector anterior.

Para especificar um tópico de deslocamento de armazenamento, você fornece um valor para a propriedade `offset.storage.topic` em sua configuração de operador antes de criar

um conector. Se quiser reutilizar o tópico de deslocamento de armazenamento para consumir deslocamentos de um conector criado anteriormente, você deverá dar ao novo conector o mesmo nome do conector antigo. Se você criar um tópico personalizado de deslocamento de armazenamento, deverá definir `cleanup.policy` como `compact` na configuração do tópico.

> **Note**
>
> Se você especificar um tópico de deslocamento de armazenamento ao criar um conector de coletor, o MSK Connect criará o tópico se ele ainda não existir. No entanto, o tópico não será usado para armazenar deslocamentos de conectores.
> Em vez disso, os deslocamentos do conector do coletor serão gerenciados usando o protocolo de grupo de consumidores Kafka. Cada conector de coletor cria um grupo chamado `connect-{CONNECTOR_NAME}`. Enquanto o grupo de consumidores existir, todos os conectores de coletor sucessivos que você criar com o mesmo valor `CONNECTOR_NAME` continuarão a partir do último deslocamento confirmado.

Example Especificar um tópico de deslocamento de armazenamento para recriar um conector de origem com uma configuração atualizada

Suponha que você tenha um conector de Change Data Capture (CDC – Captura de dados de alteração) e queira modificar a configuração do conector sem perder seu lugar no fluxo do CDC. Não é possível atualizar a configuração do conector existente, mas você pode excluir o conector e criar um novo com o mesmo nome. Para informar ao novo conector por onde começar a leitura no fluxo do CDC, você pode especificar o tópico de deslocamento de armazenamento do conector antigo em sua configuração de operador. As etapas a seguir demonstram como concluir essa tarefa.

1. Em sua máquina cliente, execute o comando a seguir para encontrar o nome do tópico de deslocamento de armazenamento do seu conector. Substitua *<bootstrapBrokerString>* pela string do agente de bootstrap do seu cluster. Para obter instruções sobre como obter sua string de agente de bootstrap, consulte Como obter agentes de bootstrap para um cluster do Amazon MSK.

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

   A saída a seguir mostra uma lista de todos os tópicos do cluster, incluindo qualquer tópico de conector interno padrão. Neste exemplo, o conector CDC existente usa o tópico padrão

de deslocamento de armazenamento criado pelo MSK Connect. É por isso que o tópico de deslocamento de armazenamento é chamado de `__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. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
3. Escolha seu conector na lista Conectores. Copie e salve o conteúdo do campo Configuração do conector para que você possa modificá-lo e usá-lo na criação do novo conector.
4. Selecione Excluir para excluir o conector. Em seguida, insira o nome do conector no campo de entrada de texto para confirmar a exclusão.
5. Crie uma configuração personalizada de operador com valores adequados ao seu cenário. Para obter instruções, consulte Criação de uma configuração personalizada de operador.

   Em sua configuração de operador, você deve especificar o nome do tópico de deslocamento de armazenamento que você recuperou anteriormente como o valor de `offset.storage.topic`, assim como na configuração a seguir.

```
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**

   Você deve dar ao seu novo conector o mesmo nome do conector antigo.

Crie um novo conector usando a configuração de operador que você definiu na etapa anterior. Para obter instruções, consulte Como criar um conector.

Considerações

Considere o seguinte ao gerenciar os deslocamentos do conector de origem.

- Para especificar um tópico de deslocamento de armazenamento, forneça o nome do tópico do Kafka no qual os deslocamentos do conector são armazenados como o valor `offset.storage.topic` em sua configuração de operador.
- Tenha cuidado ao fazer alterações na configuração de um conector. A alteração dos valores da configuração pode resultar em um comportamento não intencional do conector se um conector de origem usar valores da configuração para os principais registros de deslocamento. Recomendamos que você consulte a documentação do seu plug-in para obter orientação.
- Personalize o número padrão de partições: além de personalizar a configuração do operador adicionando `offset.storage.topic`, você pode personalizar o número de partições para os tópicos de deslocamento e armazenamento de status. As partições padrão para tópicos internos são as seguintes.
  - `config.storage.topic`: 1, não configurável, deve ser tópico de partição única
  - `offset.storage.topic`: 25, configurável fornecendo `offset.storage.partitions`
  - `status.storage.topic`: 5, configurável fornecendo `status.storage.partitions`
- Exclusão manual de tópicos: o Amazon MSK Connect cria novos tópicos internos do Kafka Connect (o nome do tópico começa com `__amazon_msk_connect`) em cada implantação de conectores. Tópicos antigos anexados a conectores excluídos não são removidos automaticamente porque tópicos internos, como `offset.storage.topic`, podem ser reutilizados entre conectores. No entanto, você pode excluir manualmente tópicos internos não utilizados criados pelo MSK Connect. Os tópicos internos são nomeados segundo o formato `__amazon_msk_connect_<offsets|status|configs>_`*`connector_name_connector_id`*.

  É possível usar a expressão regular `__amazon_msk_connect_<offsets|status|configs>_`*`connector_name_connector_id`* para excluir os tópicos internos. Você não deve excluir um tópico interno que esteja sendo usado atualmente por um conector em execução.

- Usar o mesmo nome para os tópicos internos criados pelo MSK Connect: se quiser reutilizar o tópico de deslocamento de armazenamento para consumir deslocamentos de um conector criado anteriormente, você deverá dar ao novo conector o mesmo nome do conector antigo. A propriedade `offset.storage.topic` pode ser definida usando a configuração do operador para atribuir o mesmo nome ao `offset.storage.topic` e reutilizada entre conectores diferentes. Essa configuração é descrita em Gerenciamento de deslocamentos de conectores. O MSK Connect não permite que conectores diferentes compartilhem `config.storage.topic` e `status.storage.topic`. Esses tópicos são criados sempre que você cria um novo conector no MSKC. Eles são nomeados automaticamente de acordo com o formato `__amazon_msk_connect_<status|configs>_`*`connector_name_connector_id`* e, portanto, são diferentes nos diferentes conectores que você cria.

Externalizando informações confidenciais usando provedores de configuração

Este exemplo mostra como externalizar informações confidenciais para o Amazon MSK Connect usando um provedor de configuração de código aberto. Um provedor de configuração permite que você especifique variáveis, em vez de texto simples, em uma configuração de conector ou de operador, e os operadores em execução em seu conector resolvem essas variáveis em runtime. Isso evita que credenciais e outros segredos sejam armazenados em texto simples. O provedor de configuração no exemplo suporta a recuperação de parâmetros de configuração do AWS Secrets Manager, Amazon S3 e Systems Manager (SSM). Na Etapa 2, você verá como configurar o armazenamento e a recuperação de informações confidenciais para o serviço que deseja configurar.

Tópicos

Etapa 1: criar um plug-in personalizado e fazer o upload para o S3

Para criar um plug-in personalizado, crie um arquivo zip que contenha o conector e o msk-config-provider executando os seguintes comandos em sua máquina local.

Para criar um plug-in personalizado usando uma janela de terminal e o Debezium como conector

Use a AWS CLI para executar comandos como superusuário com credenciais que permitem acessar seu bucket do S3. AWS Para obter informações sobre como instalar e configurar a AWS CLI, consulte Introdução à AWS CLI no Guia do usuário.AWS Command Line Interface Para obter informações sobre o uso da AWS CLI com o Amazon S3, consulte Usando o Amazon S3 com a AWS CLI no Guia do usuário.AWS Command Line Interface

1. Em uma janela de terminal, crie uma pasta nomeada `custom-plugin` no seu espaço de trabalho usando o comando a seguir.

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

2. Baixe a versão estável mais recente do plug-in MySQL Connector no site do Debezium usando o comando a seguir.

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

   Extraia o arquivo gzip baixado na pasta `custom-plugin` usando o comando a seguir.

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

3. Baixe o arquivo zip do provedor de configuração do MSK usando o comando a seguir.

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

   Extraia o arquivo zip baixado na `custom-plugin` pasta usando o comando a seguir.

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

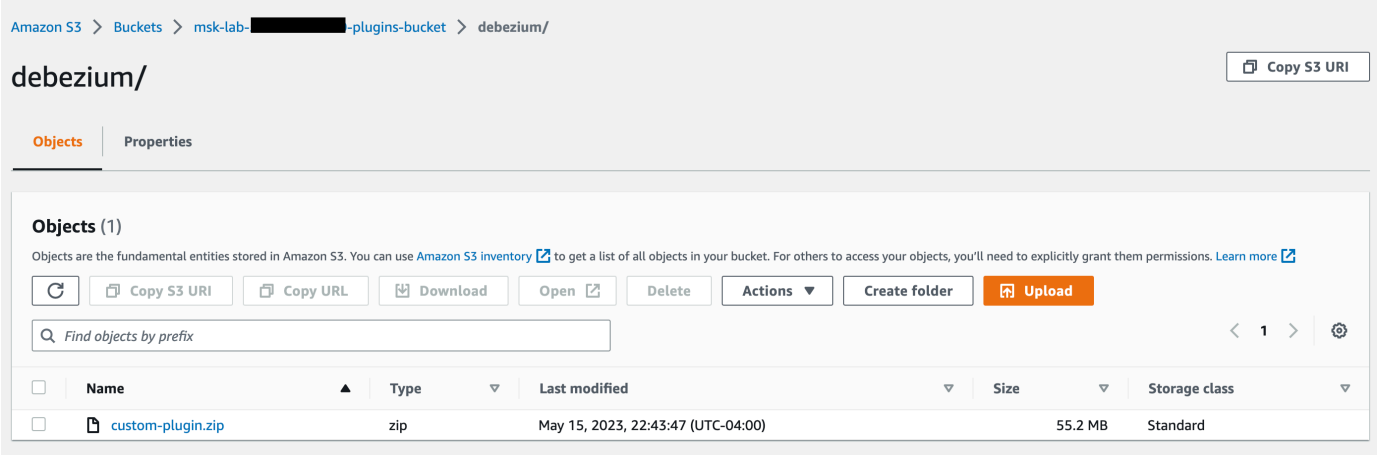
4. Compacte o conteúdo do provedor de configuração do MSK da etapa acima e do conector personalizado em um só arquivo chamado `custom-plugin.zip`.

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

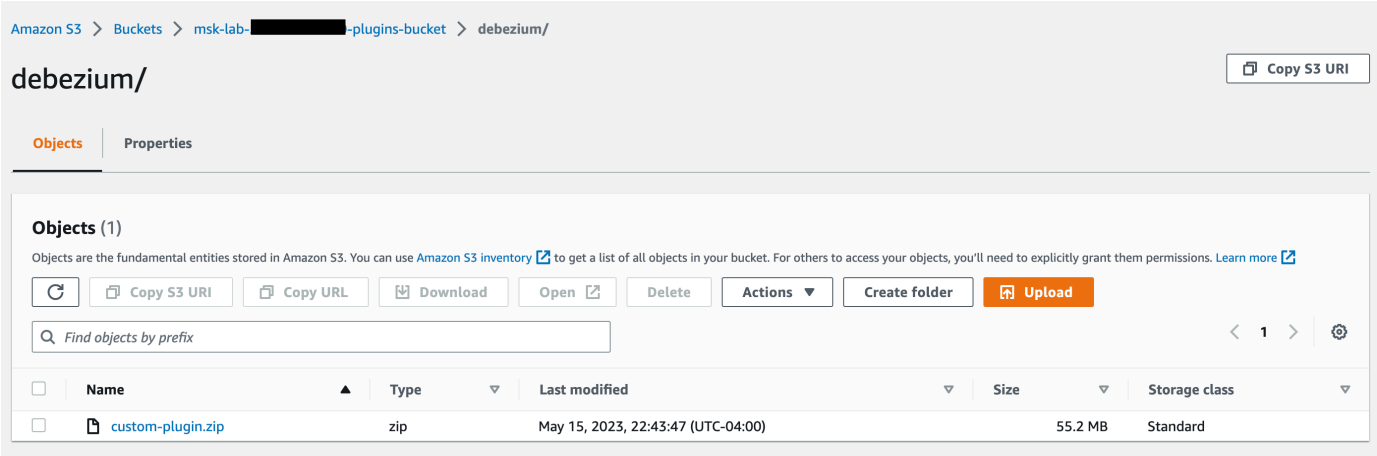
5. Faça upload do arquivo para o S3 para referência posterior.

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

6. No console do Amazon MSK, na seção MSK Connect, escolha Plug-in personalizado, depois escolha Criar plug-in personalizado e navegue no bucket s3:<***S3_URI_BUCKET_LOCATION***> do S3 para selecionar o arquivo ZIP do plug-in personalizado que você acabou de enviar.



7. Insira **`debezium-custom-plugin`** para o nome do plug-in. Opcionalmente, insira uma descrição e escolha Criar um plug-in personalizado.



Etapa 2: configurar parâmetros e permissões para diferentes provedores

Você pode configurar valores de parâmetros nestes três serviços:

- Secrets Manager
- Systems Manager Parameter Store
- S3: Simple Storage Service

Selecione uma das guias abaixo para obter instruções sobre como configurar parâmetros e permissões relevantes para esse serviço.

Configure in Secrets Manager

Para configurar valores de parâmetros no Secrets Manager

1. Abra o console do Secrets Manager.
2. Crie um novo segredo para armazenar suas credenciais ou segredos. Para obter instruções, consulte Criar um AWS Secrets Manager segredo no Guia AWS Secrets Manager do usuário.
3. Copie o ARN do seu segredo.
4. Adicione as permissões do Secrets Manager do exemplo de política a seguir ao seu perfil de execução de serviço. Substitua *<arn:aws:secretsmanager:us-east-1:123456789000:secret*: -1234> pelo ARN do seu segredo. MySecret
5. Adicione a configuração do operador e as instruções do conector.

```
{
        "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. Para usar o provedor de configuração do Secrets Manager, copie as seguintes linhas de código na caixa de texto de configuração do operador na Etapa 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. Para o provedor de configuração do Secrets Manager, copie as seguintes linhas de código na configuração do conector na Etapa 4.

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

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

Você também pode usar a etapa acima com mais provedores de configuração.

Configure in Systems Manager Parameter Store

Para configurar valores de parâmetros no Systems Manager Parameter Store

1. Abra o console do Systems Manager.
2. No painel de navegação, selecione Parameter Store (Repositório de parâmetros).
3. Crie um novo parâmetro para armazenar no Systems Manager. Para obter instruções, consulte Criar um parâmetro do Systems Manager (console) no Guia AWS Systems Manager do usuário.
4. Copie o ARN do seu parâmetro.

5. Adicione as permissões do Systems Manager do exemplo de política a seguir ao seu perfil de execução de serviço. Substitua *<arn:aws:ssm:us-east-1:123456789000:parameter/>* pelo ARN do seu parâmetro. MyParameterName

```
{
        "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. Para usar o provedor de configuração do Parameter Store, copie as seguintes linhas de código na caixa de texto de configuração do operador na Etapa 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. Para o provedor de configuração do Parameter Store, copie as seguintes linhas de código na configuração do conector na Etapa 5.

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

Você também pode agrupar as duas etapas acima com mais provedores de configuração.

Configure in Amazon S3

Para configurar objetos/arquivos no Amazon S3

1. Abra o console Amazon S3.
2. Carregue um objeto para um bucket no S3. Para obter instruções, consulte Carregar objetos.
3. Copie o ARN do seu objeto.
4. Adicione as permissões de leitura de objeto do Amazon S3 do exemplo de política a seguir ao seu perfil de execução de serviço. Substitua *<arn:aws:s3:::MY_S3_BUCKET/path/to/custom-plugin.zip>* pelo ARN do seu objeto.

```
{
    "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. Para usar o provedor de configuração do Amazon S3, copie as seguintes linhas de código na caixa de texto de configuração do operador na Etapa 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. Para o provedor de configuração do Amazon S3, copie as seguintes linhas de código na configuração do conector na Etapa 4.

```
#Example implementation for S3 object

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

Você também pode agrupar as duas etapas acima com mais provedores de configuração.

Etapa 3: criar uma configuração personalizada de operador com informações sobre seu provedor de configuração

1. Selecione as Configurações do operador na seção Amazon MSK Connect.
2. Selecione Criar configuração de operador.
3. Digite `SourceDebeziumCustomConfig` na caixa de texto Nome da configuração do operador. A descrição é opcional.
4. Copie o código de configuração relevante com base nos provedores desejados e cole-o na caixa de texto de Configuração do operador.
5. Este é um exemplo da configuração de operador para todos os três provedores:

```
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. Clique em Criar configuração de operador.

Etapa 4: criar o conector

1. Crie um novo conector usando as instruções em Criar um novo conector.
2. Escolha o arquivo `custom-plugin.zip` que você enviou para o bucket do S3 em ??? como origem do plug-in personalizado.
3. Copie o código de configuração relevante com base nos provedores desejados e cole-o no campo Configuração do conector.
4. Este é um exemplo da configuração do conector para todos os três provedores:

```
#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. Selecione Usar uma configuração personalizada e escolha SourceDebeziumCustomConfigno menu suspenso Configuração do trabalhador.
6. Siga as etapas restantes das instruções em Criar conector.

Considerações

Considere o seguinte ao usar o provedor de configuração do MSK com o Amazon MSK Connect:

- Atribua as permissões adequadas ao usar os provedores de configuração para o perfil de execução de serviços do IAM.
- Defina os provedores de configuração nas configurações de trabalho e sua implementação na configuração do conector.
- Valores confidenciais de configuração podem aparecer nos registros do conector se um plug-in não definir esses valores como secretos. O Kafka Connect trata valores de configuração indefinidos da mesma forma que qualquer outro valor de texto simples. Para saber mais, consulte Como evitar que segredos apareçam nos logs do conector.
- Por padrão, o MSK Connect reinicia frequentemente um conector quando o conector usa um provedor de configuração. Para desativar esse comportamento de reinicialização, você pode definir o valor `config.action.reload` como `none` na configuração do conector.

Perfis e políticas do IAM para o MSK Connect

Tópicos

Perfil de execução do serviço

> **Note**
>
> O Amazon MSK Connect não é compatível com o uso do perfil vinculado a serviço como o perfil de execução do serviço. É necessário criar um perfil de execução do serviço distinto. Para obter instruções sobre como criar uma função personalizada do IAM, consulte Como criar uma função para delegar permissões a um AWS serviço no Guia do usuário do IAM.

Ao criar um conector com o MSK Connect, você precisa especificar um perfil do AWS Identity and Access Management (IAM) para usar com ele. Seu perfil de execução do serviço deve ter a seguinte política de confiança para que o MSK Connect possa assumi-lo. Para obter informações sobre as chaves de contexto de condição, consulte the section called “Prevenção contra o ataque do “substituto confuso” em todos os serviços”.

```
{
  "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"
        }
      }
    }
  ]
}
```

Se o cluster Amazon MSK que você deseja usar com seu conector for um cluster que usa autenticação do IAM, será necessário adicionar a seguinte política de permissões ao perfil de execução do serviço do conector. Para obter informações sobre como encontrar o UUID do cluster e estruturar ARNs de tópicos, consulte the section called “Recursos”.

```
{
    "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-*"
            ]
        }
    ]
}
```

Dependendo do tipo de conector, talvez você também precise anexar à função de execução do serviço uma política de permissões que permita o acesso aos AWS recursos. Por exemplo, se seu conector precisar enviar dados para um bucket do S3, o perfil de execução do serviço deverá ter uma política de permissões que conceda permissão para gravar nesse bucket. Para fins de teste, você pode usar uma das políticas predefinidas do IAM que dão acesso total, como `arn:aws:iam::aws:policy/AmazonS3FullAccess`. No entanto, por motivos de segurança, recomendamos que você use a política mais restritiva que permita que seu conector leia da AWS fonte ou grave no AWS coletor.

Exemplos de políticas do IAM para o MSK Connect

Para fornecer acesso total a todas as funcionalidades do MSK Connect a um usuário não administrador, anexe uma política como a seguinte ao perfil do IAM do usuário.

```
{
    "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"
        }
    ]
}
```

Prevenção contra o ataque do “substituto confuso” em todos os serviços

O problema “confused deputy” é um problema de segurança em que uma entidade que não tem permissão para executar uma ação pode coagir uma entidade mais privilegiada a executá-la. Em AWS, a falsificação de identidade entre serviços pode resultar no problema confuso do deputado. A imitação entre serviços pode ocorrer quando um serviço (o serviço de chamada) chama outro serviço (o serviço chamado). O serviço de chamada pode ser manipulado para utilizar as suas permissões para atuar nos recursos de outro cliente em que, de outra forma, ele não teria permissão para acessar. Para evitar isso, o AWS fornece ferramentas que ajudam você a proteger seus dados para todos os serviços com entidades principais de serviço que receberam acesso aos recursos em sua conta.

Recomendamos usar as chaves de contexto de condição global `aws:SourceArn` e `aws:SourceAccount` em políticas de recursos para limitar as permissões que o MSK Connect concede a outro serviço para o recurso. Se o valor `aws:SourceArn` não contiver o ID da conta (p. ex., um ARN de um bucket do Amazon S3 não contiver o ID da conta), você deverá usar ambas as chaves de contexto de condição global para limitar as permissões. Se você utilizar ambas as chaves de contexto de condição global e o valor de `aws:SourceArn` contiver o ID da conta, o valor de `aws:SourceAccount` e a conta no valor de `aws:SourceArn` deverão utilizar o mesmo ID de conta quando utilizados na mesma declaração da política. Utilize `aws:SourceArn` se quiser que apenas um recurso seja associado ao acesso entre serviços. Use `aws:SourceAccount` se quiser permitir que qualquer recurso nessa conta seja associado ao uso entre serviços.

No caso do MSK Connect, o valor de `aws:SourceArn` deve ser um conector do MSK.

A maneira mais eficaz de se proteger do problema 'confused deputy' é usar a chave de contexto de condição global `aws:SourceArn` com o ARN completo do recurso. Se você não souber o ARN completo do recurso ou se estiver especificando vários recursos, use a chave de condição de contexto global `aws:SourceArn` com curingas (*) para as partes desconhecidas do ARN. Por exemplo, *`arn:aws:kafkaconnect:us-east-1:123456789012:connector/*`* representa todos os conectores que pertencem à conta com o ID 123456789012 na região Leste dos EUA (Norte da Virgínia).

O exemplo a seguir mostra como é possível usar as chaves de contexto de condição globais `aws:SourceArn` e `aws:SourceAccount` no MSK Connect para evitar o problema "confused deputy". Substitua *`Account-ID`* e *`MSK-Connector-ARN`* por suas informações.

```
{
  "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 políticas gerenciadas para o MSK Connect

Uma política AWS gerenciada é uma política autônoma criada e administrada por AWS. AWS as políticas gerenciadas são projetadas para fornecer permissões para muitos casos de uso comuns, para que você possa começar a atribuir permissões a usuários, grupos e funções.

Lembre-se de que as políticas AWS gerenciadas podem não conceder permissões de privilégio mínimo para seus casos de uso específicos porque estão disponíveis para uso de todos os AWS clientes. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo cliente da específicas para seus casos de uso.

Você não pode alterar as permissões definidas nas políticas AWS gerenciadas. Se AWS atualizar as permissões definidas em uma política AWS gerenciada, a atualização afetará todas as identidades principais (usuários, grupos e funções) às quais a política está anexada. AWS é mais provável que atualize uma política AWS gerenciada quando uma nova AWS service (Serviço da AWS) é lançada ou novas operações de API são disponibilizadas para serviços existentes.

Para mais informações, consulte Políticas gerenciadas pela AWS no Manual do usuário do IAM.

AWS política gerenciada: AmazonMSK ConnectReadOnlyAccess

Essa política concede ao usuário as permissões necessárias para listar e descrever os recursos do MSK Connect.

É possível anexar a política `AmazonMSKConnectReadOnlyAccess` a suas identidades do IAM.

```
{
    "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 política gerenciada: KafkaConnectServiceRolePolicy

Essa política concede ao serviço MSK Connect as permissões necessárias para criar e gerenciar interfaces de rede que tenham a tag `AmazonMSKConnectManaged:true`. Essas interfaces de rede permitem que a rede do MSK Connect acesse os recursos em sua Amazon VPC, como um cluster do Apache Kafka ou uma origem ou um coletor.

Você não pode se vincular KafkaConnectServiceRolePolicy às suas entidades do IAM. Essa política é anexada a um perfil vinculado a serviço que permite que o MSK Connect realize ações em seu nome.

```
{
 "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"
    }
   }
  }
```

```
 ]
}
```

Atualizações do MSK Connect para políticas AWS gerenciadas

Veja detalhes sobre as atualizações das políticas AWS gerenciadas do MSK Connect desde que esse serviço começou a rastrear essas alterações.

Alteração	Descrição	Data
Atualização da política somente leitura do MSK Connect	O MSK Connect atualizou a ConnectReadOnlyAccess política do AmazonMSK para remover as restrições nas operações de listagem.	13 de outubro de 2021
O MSK Connect começou a monitorar alterações	O MSK Connect começou a monitorar as mudanças em suas políticas AWS gerenciadas.	14 de setembro de 2021

Uso de perfis vinculados a serviço para o MSK Connect

O Amazon MSK Connect usa funções AWS Identity and Access Management vinculadas a serviços (IAM). Um perfil vinculado a serviço é um tipo especial de perfil do IAM vinculado diretamente ao MSK Connect. As funções vinculadas ao serviço são predefinidas pelo MSK Connect e incluem todas as permissões que o serviço exige para chamar outros AWS serviços em seu nome.

Um perfil vinculado a serviço facilita a configuração do MSK Connect porque você não precisa adicionar as permissões necessárias manualmente. O MSK Connect define as permissões dos perfis vinculados ao serviço e, exceto se definido de outra forma, somente o MSK Connect pode assumir seus perfis. As permissões definidas incluem a política de confiança e a política de permissões, que não pode ser anexada a nenhuma outra entidade do IAM.

Para obter informações sobre outros serviços compatíveis com funções vinculadas a serviços, consulte Serviços da AWS compatíveis com o IAM e procure os serviços que contenham Sim na

coluna Função vinculada a serviço. Escolha um Sim com um link para visualizar a documentação do perfil vinculado a esse serviço.

Permissões de perfil vinculado a serviço para o MSK Connect

O MSK Connect usa a função vinculada ao serviço chamada — AWSServiceRoleForKafkaConnectPermite que o Amazon MSK Connect acesse os recursos da Amazon em seu nome.

A função AWSServiceRoleForKafkaConnect vinculada ao serviço confia no `kafkaconnect.amazonaws.com` serviço para assumir a função.

Para obter mais informações sobre a política de permissões usada pelo perfil, consulte the section called “KafkaConnectServiceRolePolicy”.

Você deve configurar permissões para que uma entidade do IAM (por exemplo, um usuário, grupo ou função) crie, edite ou exclua uma função vinculada a serviço. Para mais informações, consulte Permissões de perfil vinculado ao serviço no Guia do usuário do IAM.

Criação de um perfil vinculado a serviço para o MSK Connect

Não é necessário criar manualmente uma função vinculada a serviço. Quando você cria um conector na AWS Management Console, na ou na AWS API AWS CLI, o MSK Connect cria a função vinculada ao serviço para você.

Se excluir esse perfil vinculado ao serviço e precisar criá-lo novamente, você poderá usar esse mesmo processo para recriar o perfil em sua conta. Quando você cria um conector, o MSK Connect cria um perfil vinculado a serviço para você novamente.

Edição de um perfil vinculado a serviço para o MSK Connect

O MSK Connect não permite que você edite a função vinculada ao AWSServiceRoleForKafkaConnect serviço. Depois que você criar um perfil vinculado ao serviço, não poderá alterar o nome do perfil, pois várias entidades podem fazer referência ao perfil. No entanto, você poderá editar a descrição do perfil usando o IAM. Para obter mais informações, consulte Editar uma função vinculada a serviço no Guia do usuário do IAM.

Exclusão de um perfil vinculado a serviço para o MSK Connect

Você pode usar o console do IAM AWS CLI ou a AWS API para excluir manualmente a função vinculada ao serviço. Para isso, primeiro é necessário excluir manualmente todos os conectores

do MSK Connect e excluir o perfil manualmente. Para mais informações, consulte Excluir um perfil vinculado ao serviço no Guia do usuário do IAM.

Regiões compatíveis com perfis vinculados a serviço do MSK Connect

O MSK Connect é compatível com perfis vinculados a serviço em todas as regiões nas quais o serviço esteja disponível. Para obter mais informações, consulte Regiões e endpoints da AWS.

Como habilitar o acesso à Internet para o Amazon MSK Connect

Se o seu conector para o Amazon MSK Connect precisar de acesso à Internet, recomendamos que você use as seguintes configurações Amazon Virtual Private Cloud (VPC) para habilitar esse acesso.

- Configure seu conector com sub-redes privadas.
- Crie um gateway NAT público ou uma instância NAT pública para sua VPC em uma sub-rede pública. Para obter mais informações, consulte a página Conectar sub-redes à Internet ou a outras VPCs usando dispositivos NAT no Guia do usuário da Amazon Virtual Private Cloud.
- Permita o tráfego de saída de suas sub-redes privadas para seu gateway ou instância NAT.

Como configurar um gateway NAT para o Amazon MSK Connect

As etapas a seguir mostram como configurar um gateway NAT para permitir o acesso à Internet para um conector. Você deve concluir estas etapas antes de criar um conector em uma sub-rede privada.

Pré-requisitos

Certifique-se de ter os seguintes itens.

- O ID do Amazon Virtual Private Cloud (VPC) associado ao seu cluster. Por exemplo, vpc-123456ab.
- Os IDs das sub-redes privadas em sua VPC. Por exemplo, subnet-a1b2c3de, subnet-f4g5h6ij etc. Você deve configurar seu conector com sub-redes privadas.

Para habilitar o acesso à Internet para seu conector

1. Abra o Amazon Virtual Private Cloud console em https://console.aws.amazon.com/vpc/.
2. Crie uma sub-rede pública para seu gateway NAT com um nome descritivo e anote o ID da sub-rede. Para obter instruções detalhadas, consulte Criar uma sub-rede na VPC.

3. Crie um gateway da Internet para que a VPC possa se comunicar com a Internet e anote o ID do gateway. Anexe o gateway da internet à sua VPC. Para obter mais instruções, consulte Criar e anexar um gateway da Internet à VPC.
4. Provisione um gateway NAT público para que os hosts em suas sub-redes privadas possam acessar sua sub-rede pública. Ao criar o gateway NAT, selecione a sub-rede pública que você criou anteriormente. Para obter instruções, consulte Create a NAT gateway (Criar um gateway NAT)
5. Configure suas tabelas de rotas. Para concluir essa configuração, você deve ter duas tabelas de rotas no total. Você já deve ter uma tabela de rotas principal criada automaticamente junto com sua VPC. Nesta etapa, você cria uma tabela de rotas adicional para sua sub-rede pública.

   a. Use as configurações a seguir para modificar a tabela de rotas principal da sua VPC para que suas sub-redes privadas roteiem o tráfego para seu gateway NAT. Para obter instruções, consulte Trabalhar com tabelas de rotas no Guia do usuário do Amazon Virtual Private Cloud.

      Tabela de rotas MSKC privado

Propriedade	Valor
Name tag	Recomendamos que você atribua uma tag de nome descritivo a essa tabela de rotas para ajudar na identificação dela. Por exemplo, MSKC privado.
Sub-redes associadas	Suas sub-redes privadas
Uma rota para habilitar o acesso à Internet para o MSK Connect	• Destino: 0.0.0.0/0 • Alvo: o ID do seu gateway NAT Por exemplo, nat-12a345bc6789efg1h.
Uma rota local para o tráfego interno	• Destino: 10.0.0.0/16 Esse valor pode ser diferente dependendo do bloco CIDR da sua VPC. • Alvo: local

   b. Siga as instruções em Criar uma tabela de rotas personalizada para criar uma tabela de rotas para sua sub-rede pública. Ao criar a tabela, insira um nome descritivo no campo

Tag de nome para ajudar você a identificar a qual sub-rede a tabela está associada. Por exemplo, MSKC público.

c. Configure sua tabela de rotas MSKC público usando as configurações a seguir.

Propriedade	Valor
Name tag	MSKC público ou um nome descritivo diferente que você escolher
Sub-redes associadas	Sua sub-rede pública com gateway NAT
Uma rota para habilitar o acesso à Internet para o MSK Connect	• Destino: 0.0.0.0/0 • Alvo: o ID do seu gateway da Internet Por exemplo, igw-1a234bc5.
Uma rota local para o tráfego interno	• Destino: 10.0.0.0/16 Esse valor pode ser diferente dependendo do bloco CIDR da sua VPC. • Alvo: local

Nomes de host DNS privados

Com o suporte a nomes de host DNS privados no MSK Connect, você pode configurar conectores para consultar nomes de domínio públicos ou privados. O suporte dependerá dos servidores DNS especificados no Conjunto de opções de DHCP da VPC.

Um conjunto de opções de DHCP é um grupo de configurações de rede que instâncias do EC2 usam em uma VPC para comunicação pela rede da VPC. Cada VPC tem um conjunto padrão de opções de DHCP, mas você pode criar um conjunto personalizado de opções de DHCP se quiser que as instâncias em sua VPC usem um servidor de DNS diferente para a resolução de nomes de domínio em vez do servidor DNS fornecido pela Amazon. Consulte Conjuntos de opções de DHCP na Amazon VPC.

Antes da inclusão da capacidade/recurso de resolução de DNS privado no MSK Connect, os conectores usavam o serviço de resolvedores de DNS da VPC para consultas de DNS de um conector do cliente. Os conectores não usavam os servidores DNS definidos nos conjuntos de opções de DHCP da VPC do cliente para a resolução de DNS.

Os conectores só podiam consultar nomes de host nas configurações de conectores do cliente ou em plug-ins que fossem resolvíveis publicamente. Eles não conseguiam resolver nomes de host privados definidos em uma zona hospedada de maneira privada nem usar servidores DNS em outra rede de clientes.

Sem o DNS privado, os clientes que optaram por tornar seus bancos de dados, data warehouses e sistemas como o Secrets Manager em sua própria VPC inacessíveis à Internet não poderiam trabalhar com conectores do MSK. Geralmente os clientes usam nomes de host DNS privados para atender à postura de segurança corporativa.

Tópicos

Como configurar um conjunto de opções de DHCP da VPC para seu conector

Os conectores usam automaticamente os servidores DNS definidos em seu conjunto de opções de DHCP da VPC quando o conector é criado. Antes de criar um conector, certifique-se de configurar o conjunto de opções de DHCP da VPC para os requisitos de resolução de nome de host DNS do seu conector.

Os conectores criados antes da disponibilização do recurso de nome de host DNS privado no MSK Connect continuam usando a configuração de resolução de DNS anterior sem necessidade de modificação.

Se você precisar apenas de uma resolução de nome de host DNS que possa ser resolvida publicamente em seu conector, para facilitar a configuração, recomendamos usar a VPC padrão da sua conta ao criar o conector. Consulte o Servidor DNS da Amazon no Guia do usuário da Amazon VPC para obter mais informações sobre o servidor DNS fornecido pela Amazon ou sobre o Amazon Route 53 Resolver.

Se você precisar resolver nomes de host DNS privados, certifique-se de que a VPC transmitida durante a criação do conector tenha suas opções de DHCP configuradas corretamente. Para obter mais informações, consulte Trabalhar com conjuntos de opções de DHCP no Guia do usuário da Amazon VPC.

Ao configurar um conjunto de opções de DHCP para resolução de nome de host DNS privado, certifique-se de que o conector possa acessar os servidores DNS personalizados que você configurar no conjunto de opções de DHCP. Caso contrário, a criação do conector falhará.

Após personalizar o conjunto de opções de DHCP da VPC, os conectores criados posteriormente nessa VPC usarão os servidores DNS que você especificou no conjunto de opções. Se você alterar o conjunto de opções após criar um conector, o conector adotará as configurações do novo conjunto de opções em alguns minutos.

Atributos de DNS para sua VPC

Certifique-se de ter os atributos de DNS da VPC configurados corretamente conforme descrito em Atributos de DNS em sua VPC e Nomes de host DNS no Guia do usuário da Amazon VPC.

Consulte Como resolver consultas de DNS entre VPCs e sua rede no Guia do desenvolvedor do Amazon Route 53 para obter informações sobre o uso de endpoints de resolução de entrada e de saída para conectar outras redes à sua VPC e trabalhar com seu conector.

Tratamento de falhas

Esta seção descreve possíveis falhas na criação de conectores associadas à resolução de DNS e ações sugeridas para resolver os problemas.

Falha	Ação sugerida
A criação do conector falhará se uma consulta de resolução de DNS falhar ou se os servidores DNS estiverem inacessíveis pelo conector.	Você pode ver falhas na criação de conectores devido a consultas malsucedidas de resolução de DNS em seus CloudWatch registros, se tiver configurado esses registros para seu conector. Verifique as configurações do servidor DNS e garanta a conectividade de rede com os servidores DNS pelo conector.
Se você alterar a configuração dos servidores DNS no conjunto de opções de DHCP da VPC enquanto um conector estiver em execução, as consultas de resolução de DNS do conector poderão falhar. Se a resolução de DNS falhar,	Você pode ver falhas na criação de conectores devido a consultas malsucedidas de resolução de DNS em seus CloudWatch registros, se tiver configurado esses registros para seu conector.

Falha	Ação sugerida
algumas das tarefas do conector podem entrar em um estado de falha.	As tarefas com falha deverão reiniciar automaticamente para que o conector volte a funcionar. Se isso não acontecer, você pode entrar em contato com o suporte para reiniciar as tarefas que falharam no conector ou recriar o conector.

Registro em log no MSK Connect

O MSK Connect pode gravar eventos de log que você pode usar para depurar seu conector. Ao criar um conector, você pode especificar zero ou mais dos seguintes destinos de log:

- Amazon CloudWatch Logs: você especifica o grupo de logs para o qual deseja que o MSK Connect envie os eventos de log do seu conector. Para obter informações sobre como criar um grupo de registros, consulte Criar um grupo de registros no Guia do usuário de CloudWatch registros.
- Amazon S3: você especifica o bucket do S3 para o qual deseja que o MSK Connect envie os eventos de log do seu conector. Para obter mais informações sobre como criar um bucket do S3, consulte Criar um bucket, no Guia do usuário do Amazon S3.
- Amazon Data Firehose: você especifica o stream de entrega para o qual deseja que o MSK Connect envie os eventos de log do seu conector. Para obter informações sobre como criar um stream de entrega, consulte Criação de um stream de entrega do Amazon Data Firehose no Guia do usuário do Firehose.

Para saber mais sobre como configurar o registro em log, consulte Habilitar o registro em log de determinados serviços da AWS no Guia do usuário do Amazon CloudWatch Logs .

O MSK Connect emite os seguintes tipos de eventos de log:

Nível	Descrição
`INFO`	Eventos de runtime de interesse na inicializ ação e no desligamento.

Nível	Descrição
`WARN`	Situações de runtime que não são erros, mas são indesejáveis ou inesperadas.
`FATAL`	Erros graves que causam encerramento prematuro.
`ERROR`	Condições inesperadas e erros de runtime que não são fatais.

Veja a seguir um exemplo de um evento de registro enviado para o CloudWatch Logs:

```
[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)
```

Como evitar que segredos apareçam nos logs do conector

> **Note**
>
> Valores confidenciais de configuração podem aparecer nos registros do conector se um plug-in não definir esses valores como secretos. O Kafka Connect trata valores de configuração indefinidos da mesma forma que qualquer outro valor de texto simples.

Se seu plug-in definir uma propriedade como secreta, o Kafka Connect editará o valor da propriedade nos registros do conector. Por exemplo, os registros de conectores a seguir demonstram que o valor será substituído por **`[hidden]`** se um plug-in definir `aws.secret.key` como um tipo `PASSWORD`.

```
   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)
```

Para evitar que segredos apareçam nos arquivos de log do conector, um desenvolvedor de plug-ins deve usar a constante de enumeração `ConfigDef.Type.PASSWORD` do Kafka Connect para definir propriedades confidenciais. Quando uma propriedade for do tipo `ConfigDef.Type.PASSWORD`, o Kafka Connect excluirá seu valor dos registros do conector, mesmo que o valor seja enviado como texto simples.

Monitoramento do MSK Connect

O monitoramento é uma parte importante da manutenção da confiabilidade, disponibilidade e desempenho do MSK Connect e de suas outras AWS soluções. A Amazon CloudWatch monitora seus AWS recursos e os aplicativos nos quais você executa AWS em tempo real. É possível coletar e rastrear métricas, criar painéis personalizados e definir alarmes que o notificam ou que realizam ações quando uma métrica especificada atinge um limite definido. Por exemplo, você pode CloudWatch monitorar o uso da CPU ou outras métricas do seu conector, para que você possa aumentar sua capacidade, se necessário. Para obter mais informações, consulte o Guia CloudWatch do usuário da Amazon.

A tabela a seguir mostra as métricas para as quais o MSK Connect envia CloudWatch sob a `ConnectorName` dimensão. O MSK Connect fornece essas métricas por padrão e sem custo adicional. CloudWatch mantém essas métricas por 15 meses, para que você possa acessar informações históricas e ter uma perspectiva melhor sobre o desempenho de seus conectores. Você também pode definir alarmes que observam determinados limites e enviam notificações ou realizam ações quando esses limites são atingidos. Para obter mais informações, consulte o Guia CloudWatch do usuário da Amazon.

Métricas do MSK Connect

Nome da métrica	Descrição
`BytesInPerSec`	O número total de bytes recebidos pelo conector.
`BytesOutPerSec`	O número total de bytes entregues pelo conector.

Nome da métrica	Descrição
`CpuUtilization`	O percentual de consumo de CPU por sistema e usuário.
`ErroredTaskCount`	O número de tarefas que apresentaram erro.
`MemoryUtilization`	O percentual da memória total em uma instância de agente, não apenas a memória de pilha da máquina virtual Java (JVM) atualment e em uso. Normalmente, a JVM não libera memória de volta para o sistema operacion al. Portanto, o tamanho da pilha da JVM (MemoryUtilization) geralmente começa com um tamanho mínimo de pilha que aumenta incrementalmente até um máximo estável de cerca de 80-90%. O uso da pilha da JVM pode aumentar ou diminuir conforme o uso efetivo da memória do conector muda.
`RebalanceCompletedTotal`	O número total de rebalanceamentos concluído s por esse conector.
`RebalanceTimeAvg`	O tempo médio em milissegundos gasto pelo conector no rebalanceamento.
`RebalanceTimeMax`	O tempo máximo em milissegundos gasto pelo conector no rebalanceamento.
`RebalanceTimeSinceLast`	O tempo em milissegundos desde que esse conector concluiu o rebalanceamento mais recente.
`RunningTaskCount`	O número de tarefas em execução no conector.
`SinkRecordReadRate`	O número médio de registros lidos por segundo do cluster do Apache Kafka ou do Amazon MSK.

Nome da métrica	Descrição
`SinkRecordSendRate`	O número médio de registros que são gerados pelas transformações e enviados ao destino por segundo. Esse número não inclui registros filtrados.
`SourceRecordPollRate`	O número médio de registros produzidos ou pesquisados por segundo.
`SourceRecordWriteRate`	O número médio de registros gerados pelas transformações e gravados no cluster do Apache Kafka ou do Amazon MSK por segundo.
`TaskStartupAttemptsTotal`	O número total de inicializações de tarefas que o conector tentou realizar. Você pode usar essa métrica para identificar anomalias nas tentativas de inicialização de tarefas.
`TaskStartupSuccessPercentage`	O percentual médio de tarefas bem-sucedidas iniciadas para o conector. Você pode usar essa métrica para identificar anomalias nas tentativa s de inicialização de tarefas.
`WorkerCount`	O número de operadores em execução no conector.

Exemplos

Esta seção inclui exemplos para ajudar você a configurar os recursos do Amazon MSK Connect, como conectores e provedores de configuração terceirizados comuns.

Tópicos

- Conector de coletor do Amazon S3
- Conector de origem Debezium com provedor de configuração

Conector de coletor do Amazon S3

Este exemplo mostra como usar o conector coletor Confluent Amazon S3 e como criar um conector coletor Amazon S3 AWS CLI no MSK Connect.

1. Copie e cole o JSON a seguir em um novo arquivo. Substitua as sequências de caracteres de espaço reservado por valores que correspondam à string de conexão dos servidores de bootstrap do seu cluster do Amazon MSK e aos IDs da sub-rede e do grupo de segurança do cluster. Para obter mais informações sobre como configurar um perfil de execução de serviços, consulte the section called “Perfis e políticas do 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. Execute o AWS CLI comando a seguir na pasta em que você salvou o arquivo JSON na etapa anterior.

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

Veja a seguir um exemplo da saída que você vai obter ao executar o comando com êxito.

```
{
    "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"
}
```

Conector de origem Debezium com provedor de configuração

Este exemplo mostra como usar o plug-in do conector Debezium para MySQL com um banco de dados Amazon Aurora compatível com MySQL como origem. Neste exemplo, também configuramos o AWS Secrets Manager Config Provider de código aberto para externalizar as credenciais do banco de dados no AWS Secrets Manager. Para saber mais sobre os provedores de configuração, consulte Externalizando informações confidenciais usando provedores de configuração.

> **Important**
>
> O plug-in do conector Debezium para MySQL é compatível com apenas uma tarefa e não funciona com o modo de capacidade de escalabilidade automática para o Amazon MSK Connect. Em vez disso, você deve usar o modo de capacidade provisionada e definir `workerCount` igual a um na configuração do conector. Para saber mais sobre os modos de capacidade do MSK Connect, consulte Capacidade do conector.

Antes de começar

Seu conector deve ser capaz de acessar a Internet para poder interagir com serviços como os AWS Secrets Manager que estão fora do seu Amazon Virtual Private Cloud. As etapas desta seção ajudam você a concluir as tarefas a seguir para habilitar o acesso à Internet.

- Configure uma sub-rede pública que hospede um gateway NAT e roteie o tráfego para um gateway da Internet em sua VPC.
- Crie uma rota padrão que direcione seu tráfego de sub-rede privada para seu gateway NAT.

Para ter mais informações, consulte Como habilitar o acesso à Internet para o Amazon MSK Connect.

Pré-requisitos

Antes de habilitar o acesso à Internet, você precisa dos seguintes itens:

- O ID do Amazon Virtual Private Cloud (VPC) associado ao seu cluster. Por exemplo, vpc-123456ab.
- Os IDs das sub-redes privadas em sua VPC. Por exemplo, subnet-a1b2c3de, subnet-f4g5h6ij etc. Você deve configurar seu conector com sub-redes privadas.

Para habilitar o acesso à Internet para seu conector

1. Abra o Amazon Virtual Private Cloud console em https://console.aws.amazon.com/vpc/.
2. Crie uma sub-rede pública para seu gateway NAT com um nome descritivo e anote o ID da sub-rede. Para obter instruções detalhadas, consulte Criar uma sub-rede na VPC.

3. Crie um gateway da Internet para que a VPC possa se comunicar com a Internet e anote o ID do gateway. Anexe o gateway da internet à sua VPC. Para obter mais instruções, consulte Criar e anexar um gateway da Internet à VPC.
4. Provisione um gateway NAT público para que os hosts em suas sub-redes privadas possam acessar sua sub-rede pública. Ao criar o gateway NAT, selecione a sub-rede pública que você criou anteriormente. Para obter instruções, consulte Create a NAT gateway (Criar um gateway NAT)
5. Configure suas tabelas de rotas. Para concluir essa configuração, você deve ter duas tabelas de rotas no total. Você já deve ter uma tabela de rotas principal criada automaticamente junto com sua VPC. Nesta etapa, você cria uma tabela de rotas adicional para sua sub-rede pública.

   a. Use as configurações a seguir para modificar a tabela de rotas principal da sua VPC para que suas sub-redes privadas roteiem o tráfego para seu gateway NAT. Para obter instruções, consulte Trabalhar com tabelas de rotas no Guia do usuário do Amazon Virtual Private Cloud.

      Tabela de rotas MSKC privado

Propriedade	Valor
Name tag	Recomendamos que você atribua uma tag de nome descritivo a essa tabela de rotas para ajudar na identificação dela. Por exemplo, MSKC privado.
Sub-redes associadas	Suas sub-redes privadas
Uma rota para habilitar o acesso à Internet para o MSK Connect	• Destino: 0.0.0.0/0 • Alvo: o ID do seu gateway NAT Por exemplo, nat-12a345bc6789efg1h.
Uma rota local para o tráfego interno	• Destino: 10.0.0.0/16 Esse valor pode ser diferente dependendo do bloco CIDR da sua VPC. • Alvo: local

   b. Siga as instruções em Criar uma tabela de rotas personalizada para criar uma tabela de rotas para sua sub-rede pública. Ao criar a tabela, insira um nome descritivo no campo

Tag de nome para ajudar você a identificar a qual sub-rede a tabela está associada. Por exemplo, MSKC público.

c. Configure sua tabela de rotas MSKC público usando as configurações a seguir.

Propriedade	Valor
Name tag	MSKC público ou um nome descritivo diferente que você escolher
Sub-redes associadas	Sua sub-rede pública com gateway NAT
Uma rota para habilitar o acesso à Internet para o MSK Connect	• Destino: 0.0.0.0/0 • Alvo: o ID do seu gateway da Internet Por exemplo, igw-1a234bc5.
Uma rota local para o tráfego interno	• Destino: 10.0.0.0/16 Esse valor pode ser diferente dependendo do bloco CIDR da sua VPC. • Alvo: local

Agora que habilitou o acesso à Internet para o Amazon MSK Connect, você está pronto para criar um conector.

Como criar um conector de origem do Debezium

1. Criar um plug-in personalizado

   a. Baixe o plug-in do conector MySQL para obter a versão estável mais recente no site do Debezium. Anote a versão do Debezium que você baixou (versão 2.x ou a antiga série 1.x). Você criará um conector com base na sua versão do Debezium mais adiante neste procedimento.

   b. Baixe e extraia o AWS Secrets Manager Config Provider.

   c. Coloque os seguintes arquivos no mesmo diretório:

   - A pasta `debezium-connector-mysql`.
   - A pasta `jcusten-border-kafka-config-provider-aws-0.1.1`.

d. Compacte em um arquivo ZIP o diretório que você criou na etapa anterior e, em seguida, carregue o arquivo ZIP em um bucket do S3. Para obter instruções, consulte Upload de objetos no Guia do usuário do Amazon S3.

e. Copie e cole o JSON a seguir em um arquivo. Por exemplo, `debezium-source-custom-plugin.json`. Substitua *< example-custom-plugin-name >* pelo nome que você deseja que o plug-in tenha, *< arn-of-your-s 3-bucket>* pelo ARN do bucket do S3 em que você fez o upload do arquivo ZIP e *<file-key-of-ZIP-object>* pela chave de arquivo do objeto ZIP que você carregou no S3.

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

f. Execute o AWS CLI comando a seguir na pasta em que você salvou o arquivo JSON para criar um plug-in.

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

Você deve ver uma saída semelhante ao seguinte exemplo.

```
{
    "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. Execute o comando a seguir para verificar o estado do plug-in. O estado do cluster deve mudar de `CREATING` para `ACTIVE`. Substitua o espaço reservado de ARN pelo ARN que você obteve na saída do comando anterior.

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

2. Configure AWS Secrets Manager e crie um segredo para suas credenciais de banco de dados

   a. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.

   b. Crie um novo segredo para armazenar as credenciais de login do banco de dados. Para obter instruções, consulte Criar um segredo no Guia do usuário do AWS Secrets Manager.

   c. Copie o ARN do seu segredo.

   d. Adicione as permissões do Secrets Manager do exemplo de política a seguir ao seu Perfil de execução do serviço. Substitua *<arn:aws:secretsmanager:us-east-1:123456789000:secret*: -1234> pelo ARN do seu segredo. 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>"
      ]
    }
  ]
}
```

   Para obter instruções sobre como adicionar permissões do IAM, consulte Adicionar e remover permissões de identidade do IAM no Guia do usuário do IAM.

3. Criar uma configuração personalizada de operador com informações sobre seu provedor de configuração

   a. Copie as seguintes propriedades de configuração do operador em um arquivo, substituindo as strings de espaço reservado por valores que correspondam ao seu cenário. Para saber

mais sobre as propriedades de configuração do AWS Secrets Manager Config Provider, consulte a SecretsManagerConfigProviderdocumentação do plug-in.

```
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. Execute o AWS CLI comando a seguir para criar sua configuração de trabalhador personalizada.

Substitua os valores a seguir:

- *< my-worker-config-name >* - um nome descritivo para sua configuração de trabalhador personalizada
- *< encoded-properties-file-content -string>* - uma versão codificada em base64 das propriedades de texto simples que você copiou na etapa anterior

```
aws kafkaconnect create-worker-configuration --name <my-worker-config-name> --
properties-file-content <encoded-properties-file-content-string>
```

4. Criar um conector

a. Copie o seguinte JSON que corresponde à sua versão do Debezium (2.x ou 1.x) e cole-o em um novo arquivo. Substitua as strings *<placeholder>* por valores que correspondam ao seu cenário. Para obter mais informações sobre como configurar um perfil de execução de serviços, consulte the section called “Perfis e políticas do IAM”.

Para especificar as credenciais do banco de dados, a configuração usa variáveis como `${secretManager:MySecret-1234:dbusername}` em vez de texto simples. Substitua *MySecret-1234* pelo nome do seu segredo e inclua o nome da chave que você deseja recuperar. Você também deve substituir *<arn-of-config-provider-worker-configuration>* pelo ARN da sua configuração personalizada de operador.

Debezium 2.x

Para as versões 2.x do Debezium, copie o seguinte JSON e cole-o em um novo arquivo. Substitua as strings *<placeholder>* por valores que correspondam ao seu cenário.

```
{
 "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

Para as versões 1.x do Debezium, copie o seguinte JSON e cole-o em um novo arquivo. Substitua as strings *<placeholder>* por valores que correspondam ao seu cenário.

```
{
 "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. Execute o AWS CLI comando a seguir na pasta em que você salvou o arquivo JSON na etapa anterior.

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

Veja a seguir um exemplo da saída que você vai obter ao executar o comando com êxito.

```
{
    "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"
}
```

Para ver um exemplo de conector Debezium com etapas detalhadas, consulte Introdução ao Amazon MSK Connect: transmita dados de e para seus clusters do Apache Kafka usando conectores gerenciados.

Práticas recomendadas

Use isso como referência para localizar rapidamente recomendações para maximizar o desempenho do Amazon MSK Connect.

Tópicos

Conexão de conectores

As práticas recomendadas a seguir podem melhorar o desempenho da sua conectividade com o Amazon MSK Connect.

Não sobreponha IPs para emparelhamento com Amazon VPC ou Transit Gateway

Se você estiver usando o emparelhamento da Amazon VPC ou o Transit Gateway com o Amazon MSK Connect, não configure seu conector para alcançar os recursos de VPC emparelhados com IPs nas faixas CIDR:

- “10.99.0.0/16”
- “192.168.0.0/16”
- “172.21.0.0/16”

Guia de migração do Amazon MSK Connect

Esta seção descreve como migrar seu aplicativo de conector Apache Kafka para o Amazon Managed Streaming for Apache Kafka Connect (Amazon MSK Connect).

Tópicos

Benefícios do uso do Amazon MSK Connect

O Apache Kafka é uma das plataformas de streaming de código aberto mais amplamente adotadas para ingerir e processar fluxos de dados em tempo real. Com o Apache Kafka, você pode desacoplar e escalar de forma independente seus aplicativos que produzem e consomem dados.

O Kafka Connect é um componente importante da criação e execução de aplicativos de streaming com o Apache Kafka. O Kafka Connect fornece uma maneira padronizada de mover dados entre o Kafka e sistemas externos. O Kafka Connect é altamente escalável e pode lidar com grandes

volumes de dados. O Kafka Connect fornece um poderoso conjunto de operações e ferramentas de API para configurar, implantar e monitorar conectores que movem dados entre tópicos do Kafka e sistemas externos. Você pode usar essas ferramentas para personalizar e ampliar a funcionalidade do Kafka Connect para atender às necessidades específicas do seu aplicativo de streaming.

Você pode encontrar desafios ao operar clusters do Apache Kafka Connect por conta própria ou ao tentar migrar aplicativos de código aberto do Apache Kafka Connect para o. AWS Esses desafios incluem o tempo necessário para configurar a infraestrutura e implantar aplicativos, obstáculos de engenharia ao configurar clusters autogerenciados do Apache Kafka Connect e sobrecarga operacional administrativa.

Para enfrentar esses desafios, recomendamos o uso do Amazon Managed Streaming for Apache Kafka Connect (Amazon MSK Connect) para migrar seus aplicativos Apache Kafka Connect de código aberto para o. AWS O Amazon MSK Connect simplifica o uso do Kafka Connect para transmitir dados de e para entre clusters do Apache Kafka e sistemas externos, como bancos de dados, índices de pesquisa e sistemas de arquivos.

Aqui estão alguns dos benefícios de migrar para o Amazon MSK Connect:

- Eliminação da sobrecarga operacional — o Amazon MSK Connect elimina a carga operacional associada à aplicação de patches, provisionamento e escalabilidade dos clusters do Apache Kafka Connect. O Amazon MSK Connect monitora continuamente a integridade dos seus clusters do Connect e automatiza a aplicação de patches e as atualizações de versão sem causar interrupções em suas cargas de trabalho.
- Reinício automático das tarefas do Connect — O Amazon MSK Connect pode recuperar automaticamente tarefas com falha para reduzir as interrupções na produção. As falhas nas tarefas podem ser causadas por erros temporários, como a violação do limite de conexão TCP do Kafka e o rebalanceamento de tarefas quando novos trabalhadores se juntam ao grupo de consumidores para conectores de coletores.
- Escalabilidade horizontal e vertical automática — O Amazon MSK Connect permite que o aplicativo de conectores seja escalado automaticamente para suportar maiores taxas de transferência. O Amazon MSK Connect gerencia a escalabilidade para você. Você só precisa especificar o número de trabalhadores no grupo de auto scaling e os limites de utilização. Você pode usar a operação da `UpdateConnector` API Amazon MSK Connect para aumentar ou reduzir verticalmente as vCPUs entre 1 e 8 vCPUs para suportar a taxa de transferência variável.
- Conectividade de rede privada — O Amazon MSK Connect se conecta de forma privada aos sistemas de origem e coletor usando nomes AWS PrivateLink DNS privados.

Migração para o Amazon MSK Connect

Esta seção descreve resumidamente os tópicos de gerenciamento de estado usados pelo Kafka Connect e pelo Amazon MSK Connect. Esta seção também aborda os procedimentos para migrar conectores de origem e coletor.

Tópicos

Tópicos internos usados pelo Kafka Connect

Um aplicativo Apache Kafka Connect que está sendo executado no modo distribuído armazena seu estado usando tópicos internos no cluster Kafka e na associação ao grupo. A seguir estão os valores de configuração que correspondem aos tópicos internos usados nos aplicativos do Kafka Connect:

- Tópico de configuração, especificado por meio de `config.storage.topic`

  No tópico de configuração, o Kafka Connect armazena a configuração de todos os conectores e tarefas que foram iniciados pelos usuários. Sempre que os usuários atualizam a configuração de um conector ou quando um conector solicita uma reconfiguração (por exemplo, o conector detecta que pode iniciar mais tarefas), um registro é emitido para esse tópico. Este tópico tem compactação ativada, portanto, ele sempre mantém o último estado de cada entidade.
- Tópico de compensações, especificado por meio de `offset.storage.topic`

  No tópico de compensações, o Kafka Connect armazena as compensações dos conectores de origem. Assim como o tópico de configuração, o tópico de compensações está habilitado para compactação. Este tópico é usado para escrever as posições de origem somente para conectores de origem que produzem dados para o Kafka a partir de sistemas externos. Os conectores coletores, que lêem dados do Kafka e os enviam para sistemas externos, armazenam suas compensações de consumo usando grupos regulares de consumidores do Kafka.
- Tópico de status, especificado por meio de `status.storage.topic`

  No tópico de status, o Kafka Connect armazena o estado atual dos conectores e das tarefas. Esse tópico é usado como o local central para os dados que são consultados pelos usuários

da API REST. Este tópico permite que os usuários consultem qualquer trabalhador e ainda obtenham o status de todos os plug-ins em execução. Assim como os tópicos de configuração e compensações, o tópico de status também está habilitado para compactação.

Além desses tópicos, o Kafka Connect faz uso extensivo da API de associação a grupos do Kafka. Os grupos são nomeados de acordo com o nome do conector. Por exemplo, para um conector chamado file-sink, o grupo é nomeado. connect-file-sink Cada consumidor do grupo fornece registros para uma única tarefa. Esses grupos e suas compensações podem ser recuperados usando ferramentas regulares de grupos de consumidores, como. `Kafka-consumer-group.sh` Para cada conector de coletor, o tempo de execução do Connect executa um grupo regular de consumidores que extrai registros do Kafka.

Gerenciamento estadual dos aplicativos Amazon MSK Connect

Por padrão, o Amazon MSK Connect cria três tópicos separados no cluster Kafka para cada conector Amazon MSK para armazenar a configuração, o deslocamento e o status do conector. Os nomes de tópicos padrão são estruturados da seguinte forma:

- *__msk_connect_configs_ nome do conector _ id do conector*
- *__msk_connect_status_ nome do conector _ id do conector*
- *__msk_connect_offsets_ nome do conector _ id do conector*

> **Note**
>
> Para fornecer a continuidade do deslocamento entre os conectores de origem, você pode usar um tópico de armazenamento offset de sua escolha, em vez do tópico padrão. Especificar um tópico de deslocamento de armazenamento ajuda você a realizar tarefas como criar um conector de origem que retoma a leitura desde o último deslocamento de um conector anterior. Para especificar um tópico de armazenamento offset, forneça um valor para a offset.storage.topicpropriedade na configuração do Amazon MSK Connect worker antes de criar o conector.

Migração de conectores de origem para o Amazon MSK Connect

Os conectores de origem são aplicativos Apache Kafka Connect que importam registros de sistemas externos para o Kafka. Esta seção descreve o processo de migração de aplicativos de

conectores de origem do Apache Kafka Connect que estão sendo executados localmente ou clusters autogerenciados do Kafka Connect que estão sendo executados no Amazon MSK Connect. AWS

O aplicativo conector de origem do Kafka Connect armazena compensações em um tópico nomeado com o valor definido para a propriedade config. `offset.storage.topic` A seguir estão exemplos de mensagens de deslocamento para um conector JDBC que está executando duas tarefas que importam dados de duas tabelas diferentes chamadas e. `movies` `shows` A linha mais recente importada dos filmes de tabela tem uma ID primária de`18343`. A linha mais recente importada da tabela shows tem uma ID primária de`732`.

```
["jdbcsource",{"protocol":"1","table":"sample.movies"}] {"incrementing":18343}
["jdbcsource",{"protocol":"1","table":"sample.shows"}] {"incrementing":732}
```

Para migrar conectores de origem para o Amazon MSK Connect, faça o seguinte:

1. Crie um plug-in personalizado do Amazon MSK Connect extraindo bibliotecas de conectores do seu cluster Kafka Connect local ou autogerenciado.
2. Crie propriedades de trabalho do Amazon MSK Connect e defina as propriedades `key.converter`, `value.converter`, e `offset.storage.topic` com os mesmos valores definidos para o conector Kafka que está sendo executado em seu cluster atual do Kafka Connect.
3. Pause o aplicativo do conector no cluster existente fazendo uma `PUT /connectors/`*`connector-name`*`/pause` solicitação no cluster existente do Kafka Connect.
4. Certifique-se de que todas as tarefas do aplicativo conector estejam completamente interrompidas. Você pode interromper as tarefas fazendo uma `GET /connectors/`*`connector-name`*`/status` solicitação no cluster existente do Kafka Connect ou consumindo as mensagens do nome do tópico definido para a propriedade. `status.storage.topic`
5. Obtenha a configuração do conector do cluster existente. Você pode obter a configuração do conector fazendo uma `GET /connectors/`*`connector-name`*`/config/` solicitação no cluster existente ou consumindo as mensagens do nome do tópico definido para a propriedade`config.storage.topic`.
6. Crie um novo Amazon MSK Connector com o mesmo nome de um cluster existente. Crie esse conector usando o plug-in personalizado do conector que você criou na etapa 1, as propriedades de trabalho que você criou na etapa 2 e a configuração do conector que você extraiu na etapa 5.
7. Quando o status do Amazon MSK Connector for`active`, visualize os registros para verificar se o conector começou a importar dados do sistema de origem.

8. Exclua o conector no cluster existente fazendo uma `DELETE /connectors/`*`connector-name`* solicitação.

Migração de conectores de coletor para o Amazon MSK Connect

Os conectores Sink são aplicativos Apache Kafka Connect que exportam dados do Kafka para sistemas externos. Esta seção descreve o processo de migração de aplicativos conectores de coletor do Apache Kafka Connect que estão sendo executados localmente ou clusters autogerenciados do Kafka Connect que estão sendo executados no Amazon MSK Connect. AWS

Os conectores de coletor do Kafka Connect usam a API de associação de grupos do Kafka e armazenam compensações nos mesmos `__consumer_offset` tópicos de um aplicativo de consumo típico. Esse comportamento simplifica a migração do conector do coletor de um cluster autogerenciado para o Amazon MSK Connect.

Para migrar conectores de coletor para o Amazon MSK Connect, faça o seguinte:

1. Crie um plug-in personalizado do Amazon MSK Connect extraindo bibliotecas de conectores do seu cluster Kafka Connect local ou autogerenciado.
2. Crie propriedades de trabalho do Amazon MSK Connect e defina as propriedades `key.converter` `value.converter` com os mesmos valores definidos para o conector Kafka que está sendo executado em seu cluster atual do Kafka Connect.
3. Pause o aplicativo do conector em seu cluster existente fazendo uma `PUT /connectors/`*`connector-name`*`/pause` solicitação no cluster existente do Kafka Connect.
4. Certifique-se de que todas as tarefas do aplicativo conector estejam completamente interrompidas. Você pode interromper as tarefas fazendo uma `GET /connectors/`*`connector-name`*`/status` solicitação no cluster existente do Kafka Connect ou consumindo as mensagens do nome do tópico definido para a propriedade. `status.storage.topic`
5. Obtenha a configuração do conector do cluster existente. Você pode obter a configuração do conector fazendo uma `GET /connectors/`*`connector-name`*`/config` solicitação no cluster existente ou consumindo as mensagens do nome do tópico definido para a propriedade`config.storage.topic`.
6. Crie um novo Amazon MSK Connector com o mesmo nome do cluster existente. Crie esse conector usando o plug-in personalizado do conector que você criou na etapa 1, as propriedades de trabalho que você criou na etapa 2 e a configuração do conector que você extraiu na etapa 5.

7. Quando o status do Amazon MSK Connector for `active`, visualize os registros para verificar se o conector começou a importar dados do sistema de origem.
8. Exclua o conector no cluster existente fazendo uma `DELETE /connectors/`*`connector-name`* solicitação.

Solução de problemas do Amazon MSK Connect

As informações a seguir podem ajudar você a solucionar problemas que você pode vir a enfrentar com MSK Connect. Você também pode publicar seu problema no AWS re:Post.

O conector não consegue acessar recursos hospedados na Internet pública

Consulte Como habilitar o acesso à Internet para o Amazon MSK Connect.

O número de tarefas em execução do conector não é igual ao número de tarefas especificado em tasks.max

Aqui estão alguns motivos pelos quais um conector pode usar menos tarefas do que o valor especificado na configuração tasks.max:

- Algumas implementações de conectores limitam o número de tarefas que podem ser usadas. Por exemplo, o conector Debezium para MySQL está limitado ao uso de uma única tarefa.
- Quando você usa o modo de capacidade com escalabilidade automática, o Amazon MSK Connect substitui a propriedade tasks.max de um conector por um valor proporcional ao número de operadores em execução no conector e ao número de MCUs por operador.
- Para conectores de coletor, o nível de paralelismo (número de tarefas) não pode ser maior que o número de partições de tópicos. Embora você possa definir tasks.max com um valor maior que esse, uma única partição nunca é processada por mais de uma única tarefa por vez.
- No Kafka Connect 2.7.x, o atribuidor de partição de consumidor padrão é `RangeAssignor`. O comportamento desse atribuidor é fornecer a primeira partição de cada tópico a um único consumidor, a segunda partição de cada tópico a um único consumidor etc. Isso significa que o número máximo de tarefas ativas usadas por um conector de coletor com `RangeAssignor` é igual ao número máximo de partições em qualquer tópico que esteja sendo consumido. Se isso não funcionar para seu caso de uso, você deve criar uma configuração de agente na qual a propriedade `consumer.partition.assignment.strategy` seja definida como um atribuidor de partição de consumidor mais adequado. Consulte Interface do Kafka 2.7 ConsumerPartitionAssignor: todas as classes de implementação conhecidas.

Replicador do MSK

O que é o replicador do Amazon MSK?

O Amazon MSK Replicator é um recurso do Amazon MSK que permite replicar dados de forma confiável em clusters do Amazon MSK em regiões diferentes ou na AWS mesma região. Com o replicador do MSK, você pode criar facilmente aplicações de streaming regionalmente resilientes para aumentar a disponibilidade e a continuidade dos negócios. O replicador do MSK fornece replicação assíncrona automática em clusters do MSK, eliminando a necessidade de criar código personalizado, gerenciar a infraestrutura ou configurar redes entre regiões.

O replicador do MSK escala automaticamente os recursos subjacentes, permitindo que você replique dados sob demanda sem precisar monitorar ou escalar a capacidade. O replicador do MSK também replica os metadados necessários do Kafka, incluindo configurações de tópicos, listas de controle de acesso (ACLs) e deslocamentos de grupos de consumidores. Se ocorrer um evento inesperado em uma região, você pode fazer o failover para a outra AWS região e retomar o processamento sem problemas.

O replicador do MSK é compatível com Cross-Region Replication (CRR – Replicação entre regiões) e Same-Region Replication (SRR – Replicação na mesma região). Na replicação entre regiões, os clusters MSK de origem e destino estão em regiões diferentes. AWS Na replicação na mesma região, os clusters MSK de origem e de destino estão na mesma região. AWS Você precisa criar clusters de origem e de destino do MSK antes de usá-los com o replicador do MSK.

> **Note**
>
> O MSK Replicator suporta as seguintes AWS regiões: Leste dos EUA (us-east-1, Norte da Virgínia); Leste dos EUA (us-east-2, Ohio); Oeste dos EUA (us-west-2, Oregon); Europa (eu-west-1, Irlanda); Europa (eu-central-1, Frankfurt); Ásia-Pacífico (ap-southeast-1, Cingapura); Ásia-Pacífico (ap-southeast-2, Sydney), Europa (eu-north-1, Estocolmo), Ásia-Pacífico (ap-south-1, Mumbai), Europa (eu-west-3, Paris), América do Sul (sa-east-1, São Paulo), Ásia-Pacífico (ap-west-3, Paris) northeast-2, Seul), Europa (eu-west-2, Londres), Ásia-Pacífico (ap-northeast-1, Tóquio), Oeste dos EUA (us-west-1, Norte da Califórnia), Canadá (ca-central-1, Central).

Veja alguns usos comuns do replicador do Amazon MSK.

- Crie aplicações de streaming multirregionais: crie aplicações de streaming altamente disponíveis e tolerantes a falhas para aumentar a resiliência sem configurar soluções personalizadas.
- Acesso a dados com menor latência: forneça acesso a dados com menor latência para consumidores em diferentes regiões geográficas.
- Distribua dados para seus parceiros: copie dados de um cluster do Apache Kafka para vários clusters do Apache Kafka, para que diferentes equipes/parceiros tenham as próprias cópias dos dados.
- Agregar dados para análise: copie dados de vários clusters do Apache Kafka em um cluster para gerar facilmente insights sobre dados agregados em tempo real.
- Escreva localmente, acesse seus dados globalmente: configure a replicação multiativa para propagar automaticamente as gravações realizadas em uma AWS região para outras regiões, fornecendo dados com menor latência e custo.

Funcionamento do replicador do Amazon MSK

Para começar a usar o MSK Replicator, você precisa criar um novo replicador na região do seu cluster de destino. AWS O MSK Replicator copia automaticamente todos os dados do cluster na AWS região primária chamada origem para o cluster na região de destino chamada destino. Os clusters de origem e de destino podem estar na mesma região ou em AWS regiões diferentes. Você precisará criar o cluster de destino se ele não existir.

Quando você cria um replicador, o MSK Replicator implanta todos os recursos necessários na AWS região do cluster de destino para otimizar a latência da replicação de dados. A latência de replicação varia com base em muitos fatores, incluindo a distância da rede entre as AWS regiões dos seus clusters MSK, a capacidade de taxa de transferência dos clusters de origem e de destino e o número de partições nos clusters de origem e de destino. O replicador do MSK escala automaticamente os recursos subjacentes, permitindo que você replique dados sob demanda sem precisar monitorar ou escalar a capacidade.

Replicação de dados

Por padrão, o MSK Replicator copia todos os dados de forma assíncrona do deslocamento mais recente nas partições de tópicos do cluster de origem para o cluster de destino. Se a configuração “Detectar e copiar novos tópicos” estiver ativada, o MSK Replicator detectará e copiará automaticamente novos tópicos ou partições de tópicos para o cluster de destino. No entanto, pode levar até 30 segundos para que o Replicator detecte e crie os novos tópicos ou partições de tópicos

no cluster de destino. Qualquer mensagem produzida no tópico de origem antes da criação do tópico no cluster de destino não será replicada. Como alternativa, você pode configurar seu replicador durante a criação para iniciar a replicação a partir do primeiro deslocamento nas partições de tópicos do cluster de origem, se quiser replicar as mensagens existentes em seus tópicos para o cluster de destino.

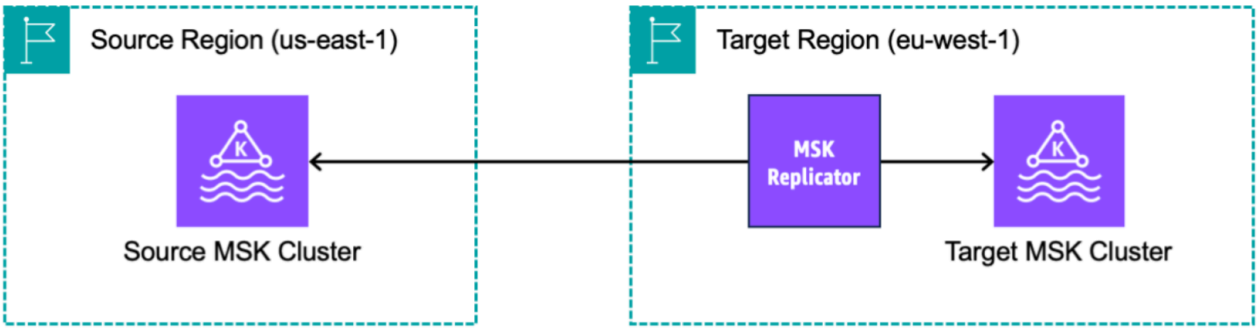
O replicador MSK não armazena seus dados. Os dados são consumidos do seu cluster de origem, armazenados em buffer na memória e gravados no cluster de destino. O buffer é limpo automaticamente quando os dados são gravados com sucesso ou falham após novas tentativas. Toda a comunicação e os dados entre o MSK Replicator e seus clusters são sempre criptografados em trânsito. Todas as chamadas da API do MSK Replicator`DescribeClusterV2`, como,`CreateTopic`, `DescribeTopicDynamicConfiguration` são capturadas em. AWS CloudTrail Seus registros do corretor MSK também refletirão o mesmo.

O MSK Replicator cria tópicos no cluster de destino com um fator de replicador de 3. Se necessário, você pode modificar o fator de replicação diretamente no cluster de destino.

Replicação de metadados

O MSK Replicator também suporta a cópia dos metadados do cluster de origem para o cluster de destino. Os metadados incluem configuração de tópicos, Listas de Controle de Acesso (ACLs) de leitura e compensações de grupos de consumidores. Assim como a replicação de dados, a replicação de metadados também acontece de forma assíncrona. Para um melhor desempenho, o MSK Replicator prioriza a replicação de dados sobre a replicação de metadados.

Como parte da sincronização de offsets de grupos de consumidores, o MSK Replicator otimiza para seus consumidores no cluster de origem, que estão lendo de uma posição mais próxima à ponta do stream (final da partição do tópico). Se seus grupos de consumidores estiverem atrasados no cluster de origem, você poderá observar um atraso maior para esses grupos de consumidores no destino em comparação com a origem. Isso significa que, após o failover para o cluster de destino, seus consumidores reprocessarão mais mensagens duplicadas. Para reduzir esse atraso, seus consumidores no cluster de origem precisariam se atualizar e começar a consumir a partir da ponta do stream (final da partição do tópico). À medida que seus consumidores se atualizarem, o MSK Replicator reduzirá automaticamente o atraso.



Requisitos e considerações sobre a criação de um replicador do Amazon MSK

Observe esses requisitos de cluster do MSK para executar um replicador do Amazon MSK.

Tópicos

- Permissões necessárias para criar um replicador do MSK
- Tipos e versões de cluster compatíveis
- Configuração de cluster do MSK Serverless
- Alterações na configuração de cluster

Permissões necessárias para criar um replicador do MSK

Veja um exemplo da política do IAM necessária para criar um replicador do MSK. A ação `kafka:TagResource` só é necessária se as tags forem fornecidas ao criar o replicador do 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": "*"
        }
    ]
}
```

Veja a seguir um exemplo de política do IAM para descrever o replicador. É necessário usar a ação `kafka:DescribeReplicator` ou a ação `kafka:ListTagsForResource`, mas não ambas.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "kafka:DescribeReplicator",
                "kafka:ListTagsForResource"
            ],
            "Resource": "*"
        }
    ]
}
```

Tipos e versões de cluster compatíveis

Estes são os requisitos para tipos de instância, versões do Kafka e configurações de rede compatíveis.

- O replicador do MSK oferece suporte a qualquer combinação de clusters provisionados do MSK e clusters do MSK com tecnologia sem servidor como clusters de origem e destino. No momento, o replicador do MSK não é compatível com outros tipos de clusters do Kafka.

- Os clusters do MSK com a tecnologia sem servidor exigem controle de acesso do IAM, não oferecem suporte à replicação de ACL do Apache Kafka e têm compatibilidade limitada com a replicação de configuração no tópico. Consulte MSK Serverless.
- O MSK Replicator é suportado somente em clusters que executam o Apache Kafka 2.7.0 ou superior, independentemente de seus clusters de origem e de destino estarem na mesma região ou em regiões diferentes. AWS
- O replicador do MSK é compatível com clusters usando tipos de instância m5.large ou maiores. Não há suporte para clusters t3.small.
- Se você estiver usando o replicador do MSK com um cluster provisionado pelo MSK, precisará de, no mínimo, três agentes nos clusters de origem e de destino. É possível replicar dados entre clusters em duas zonas de disponibilidade, mas você precisaria de um mínimo de quatro agentes nesses clusters.
- Os clusters MSK de origem e de destino devem estar na mesma AWS conta. Não há compatibilidade com a replicação entre clusters em contas diferentes.
- Se os clusters MSK de origem e de destino estiverem em AWS regiões diferentes (entre regiões), o MSK Replicator exigirá que o cluster de origem tenha a conectividade privada de várias VPCs ativada para seu método de controle de acesso IAM. Não é necessário ter multi-VPC para outros métodos de autenticação no cluster de origem. Várias VPCs não são necessárias se você estiver replicando dados entre clusters na mesma região. AWS Consulte the section called “Conectividade privada multi-VPC em uma única região”.

Configuração de cluster do MSK Serverless

- O MSK Serverless é compatível com a replicação destas configurações de tópicos para clusters de destino do MSK Serverless durante a criação do tópico: `cleanup.policy`, `compression.type`, `max.message.bytes`, `retention.bytes`, `retention.ms`.
- O MSK Serverless só é compatível com estas configurações de tópicos durante a sincronização da configuração de tópicos: `compression.type`, `max.message.bytes`, `retention.bytes`, `retention.ms`.
- O replicador usa 83 partições compactadas nos clusters de destino do MSK Serverless. Certifique-se de que os clusters de destino do MSK Serverless tenham um número suficiente de partições compactadas. Consulte Cota do MSK Serverless.

Alterações na configuração de cluster

- Recomenda-se que você não ative ou desative o armazenamento em camadas após a criação do replicador do MSK. Se seu cluster de destino não estiver em camadas, o MSK não copiará as configurações de armazenamento em camadas, independentemente de seu cluster de origem estar ou não com essa configuração. Se você ativar o armazenamento em camadas no cluster de destino após a criação do replicador, será necessário recriar o replicador. Se você quiser copiar dados de um cluster que não esteja em camadas para um cluster em camadas, você não deve copiar as configurações de tópico. Consulte Habilitar e desabilitar o armazenamento em camadas em um tópico existente.
- Não altere as configurações do cluster após a criação do replicador do MSK. As configurações do cluster são validadas durante a criação do replicador do MSK. Para evitar problemas com o replicador do MSK, não altere as configurações a seguir após a criação do replicador do MSK.
  - Altere o cluster do MSK para o tipo de instância t3.
  - Altere as permissões do perfil de execução do serviço.
  - Desabilite a conectividade privada multi-VPC do MSK.
  - Altere a política baseada em recursos anexada do cluster.
  - Altere as regras de grupos de segurança de cluster.

Conceitos básicos sobre como usar o replicador do Amazon MSK

Este tutorial mostra como configurar um cluster de origem e um de destino na mesma AWS região ou em AWS regiões diferentes. Posteriormente, você também pode usar esses clusters para criar um replicador do Amazon MSK.

Etapa 1: preparar o cluster de origem do Amazon MSK

Se você já tiver um cluster de origem do MSK criado para o replicador do MSK, certifique-se de que ele atenda aos requisitos descritos nesta seção. Caso contrário, siga estas etapas para criar um cluster de origem com a tecnologia sem servidor ou provisionado do MSK.

O processo de criação de um cluster de origem do replicador do MSK entre regiões e na mesma região é semelhante. As diferenças estão nas chamadas nos procedimentos a seguir.

1. Crie um cluster provisionado ou com tecnologia sem servidor do MSK com o controle de acesso do IAM ativado na região de origem. Seu cluster de origem deve ter, no mínimo, três agentes.

2. Para um replicador do MSK entre regiões, se a origem for um cluster provisionado, configure-o com a conectividade privada multi-VPC ativada para esquemas de controle de acesso do IAM. Observe que não há compatibilidade com o tipo de autenticação não autenticado quando o recurso multi-VPC estiver ativado. Você não precisa ativar a conectividade privada multi-VPC para outros esquemas de autenticação (mTLS ou SASL/SCRAM). É possível usar simultaneamente esquemas de autenticação mTLS ou SASL/SCRAM para seus outros clientes que se conectam ao seu cluster do MSK. Você pode configurar a conectividade privada multi-VPC nos detalhes do cluster no console, nas Configurações de rede ou com a API `UpdateConnectivity`. Consulte Proprietário do cluster ativa o recurso multi-VPC. Se seu cluster de origem for um cluster do MSK Serverless, você não precisará ativar a conectividade privada multi-VPC.

   Para um replicador do MSK na mesma região, o cluster de origem do MSK não exige conectividade privada multi-VPC e o cluster ainda pode ser acessado por outros clientes usando o tipo de autenticação não autenticada.

3. Para replicadores do MSK entre regiões, você deve anexar uma política de permissões baseada em recursos ao cluster de origem. Isso permite que o MSK se conecte a esse cluster para replicar dados. Você pode fazer isso usando os procedimentos da CLI ou AWS do console abaixo. Veja também as Políticas baseadas em recursos do Amazon MSK. Não há necessidade de executar essa etapa para replicadores do MSK na mesma região.

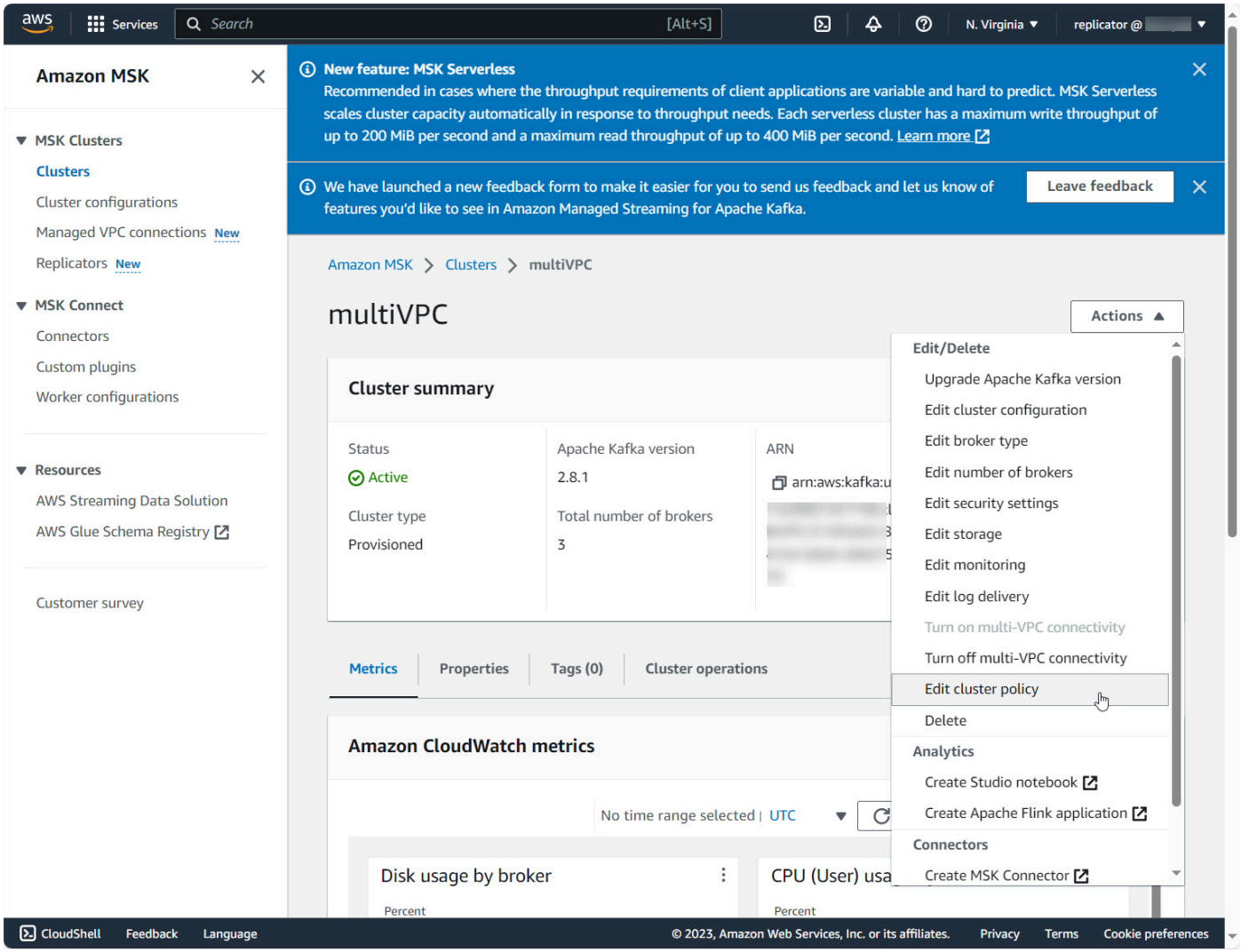
Console: create resource policy

Atualize a política de cluster de origem com o seguinte JSON. Substitua o espaço reservado pelo ARN do cluster de origem.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Service": [
                "kafka.amazonaws.com"
            ]
        },
        "Action": [
            "kafka:CreateVpcConnection",
            "kafka:GetBootstrapBrokers",
            "kafka:DescribeClusterV2"
```

```
        ],
        "Resource": "<sourceClusterARN>"
    }
  ]
}
```

Use a opção Editar política de cluster no menu Ações na página de detalhes do cluster.



CLI: create resource policy

Observação: se você usar o AWS console para criar um cluster de origem e escolher a opção de criar uma nova função do IAM, AWS anexe a política de confiança necessária à função. Se você quiser que o MSK use um perfil existente do IAM ou se você criar um perfil, anexe as seguintes políticas de confiança a esse perfil para que o replicador do MSK possa assumi-lo. Para obter informações sobre como modificar a relação de confiança de uma função, consulte Modificar uma função.

1. Obtenha a versão atual da política de cluster do MSK usando esse comando. Substitua os espaços reservados pelo ARN efetivo do cluster.

```
aws kafka get-cluster-policy –cluster-arn <Cluster ARN>
{
"CurrentVersion": "K1PA6795UKM GR7",
"Policy": "..."
}
```

2. Crie uma política baseada em recursos para permitir que o replicador do MSK acesse seu cluster de origem. Use a sintaxe a seguir como modelo, substituindo o espaço reservado pelo ARN efetivo do cluster de origem.

```
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>"
}
]
```

Etapa 2: preparar o cluster de destino do Amazon MSK

Crie um cluster de destino do MSK (provisionado ou com tecnologia sem servidor) com o controle de acesso do IAM ativado. O cluster de destino não exige que a conectividade privada multi-VPC esteja ativada. O cluster de destino pode estar na mesma AWS região ou em uma região diferente do cluster de origem. Os clusters de origem e de destino devem estar na mesma AWS conta. Seu cluster de destino deve ter, no mínimo, três agentes.

Etapa 3: criar um replicador do Amazon MSK

Antes de criar o replicador do Amazon MSK, certifique-se de ter Permissões necessárias para criar um replicador do MSK.

Tópicos

Crie um replicador usando o console da AWS na região do cluster de destino

1. Na AWS região em que seu cluster MSK de destino está localizado, abra o console do Amazon MSK em https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Escolha Replicadores para exibir a lista de replicadores na conta.
3. Escolha Criar replicador.
4. No painel Detalhes do replicador, dê um nome exclusivo ao novo replicador.

Escolher seu cluster de origem

O cluster de origem contém os dados que você deseja copiar para um cluster de destino do MSK.

1. No painel Cluster de origem, escolha a região da AWS do cluster de origem.

   Você pode consultar a região de um cluster acessando Clusters do MSK e examinando o ARN dos detalhes do cluster. O nome da região está incorporado na string do ARN. No exemplo de ARN a seguir, `ap-southeast-2` é a região do cluster.

   ```
   arn:aws:kafka:ap-southeast-2:123456789012:cluster/cluster-11/
   eec93c7f-4e8b-4baf-89fb-95de01ee639c-s1
   ```

2. Insira o ARN do seu cluster de origem ou navegue para escolher seu cluster de origem.
3. Escolha uma ou mais sub-redes para seu cluster de origem.

   O console exibe as sub-redes disponíveis na região do cluster de origem para você selecionar. Você deve selecionar, no mínimo, duas sub-redes. Para um replicador do MSK na mesma

região, as sub-redes que você seleciona para acessar o cluster de origem e as sub-redes para acessar o cluster de destino devem estar na mesma zona de disponibilidade.

4. Escolha grupos de segurança para que o replicador do MSK acesse seu cluster de origem.

   - Para replicação entre regiões (CRR), você não precisa fornecer grupos de segurança para seu cluster de origem.
   - Para replicação na mesma região (SRR), acesse o console do Amazon EC2 em https://console.aws.amazon.com/ec2/ e certifique-se de que os grupos de segurança que você fornecerá para o Replicator tenham regras de saída para permitir o tráfego para os grupos de segurança do seu cluster de origem. Além disso, certifique-se de que os grupos de segurança do seu cluster de origem tenham regras de entrada que permitam o tráfego dos grupos de segurança do Replicator fornecidos para a origem.

     Para adicionar regras de entrada ao grupo de segurança do seu cluster de origem:

     1. No AWS console, acesse os detalhes do cluster de origem selecionando o nome do cluster.
     2. Selecione a guia Propriedades e, em seguida, role para baixo até o painel Configurações de rede para selecionar o nome do Grupo de segurança aplicado.
     3. Acesse as regras de entrada e selecione Editar regras de entrada.
     4. Selecione Adicionar regra.
     5. Na coluna Tipo da nova regra, selecione TCP personalizado.
     6. Na coluna Intervalo de portas, digite`9098`. O MSK Replicator usa o controle de acesso do IAM para se conectar ao seu cluster que usa a porta 9098.
     7. Na coluna Origem, digite o nome do grupo de segurança que você fornecerá durante a criação do Replicador para o cluster de origem (pode ser o mesmo que o grupo de segurança do cluster de origem MSK) e selecione Salvar regras.

     Para adicionar regras de saída ao grupo de segurança do Replicator fornecido para a origem:

     1. No AWS console do Amazon EC2, acesse o grupo de segurança que você fornecerá durante a criação do Replicator para a fonte.
     2. Acesse as regras de saída e selecione Editar regras de saída.
     3. Selecione Adicionar regra.

4. Na coluna Tipo da nova regra, selecione TCP personalizado.
5. Na coluna Intervalo de portas, digite`9098`. O MSK Replicator usa o controle de acesso do IAM para se conectar ao seu cluster que usa a porta 9098.
6. Na coluna Origem, digite o nome do grupo de segurança do cluster de origem MSK e selecione Salvar regras.

> **Note**
>
> Como alternativa, se você não quiser restringir o tráfego usando seus grupos de segurança, poderá adicionar regras de entrada e saída que permitam Todo o Tráfego.
> 1. Selecione Adicionar regra.
> 2. Na coluna Tipo, escolha Todo o tráfego.
> 3. Na coluna Origem, digite `0.0.0.0/0` e selecione Salvar regras.

Escolher seu cluster de destino

O cluster de destino é o cluster do MSK provisionado ou com tecnologia sem servidor para o qual os dados de origem são copiados.

> **Note**
>
> O replicador do MSK cria novos tópicos no cluster de destino com um prefixo gerado automaticamente adicionado ao nome do tópico. Por exemplo, o replicador do MSK replica dados em "`topic`" com base no cluster de origem para um novo tópico chamado `<sourceKafkaClusterAlias>.topic` no cluster de destino. Isso serve para distinguir entre tópicos que contenham dados replicados do cluster de origem de outros tópicos no cluster de destino e para evitar que os dados sejam replicados circularmente entre os clusters. Você pode encontrar o prefixo que será adicionado aos nomes dos tópicos no cluster de destino no campo sourceKafkaClusterAlias usando a `DescribeReplicator` API ou a página de detalhes do Replicator no console MSK. O prefixo no cluster de destino é <sourceKafkaCluster Alias>.

1. No painel Cluster de destino, escolha a AWS região em que o cluster de destino está localizado.
2. Insira o ARN do seu cluster de destino ou navegue para escolher seu cluster de destino.

3. Escolha uma ou mais sub-redes para seu cluster de destino.

   O console exibe as sub-redes disponíveis na região do cluster de destino para você selecionar. Selecione ao menos duas sub-redes.

4. Escolha grupos de segurança para que o replicador do MSK acesse seu cluster de destino.

   Os grupos de segurança disponíveis na região do cluster de destino são exibidos para você selecionar. O grupo de segurança escolhido será associado a cada conexão. Para obter mais informações sobre o uso de grupos de segurança, consulte Controle o tráfego para seus AWS recursos usando grupos de segurança no Guia do usuário da Amazon VPC.

   - Para replicação entre regiões (CRR) e replicação na mesma região (SRR), acesse o console do Amazon EC2 em https://console.aws.amazon.com/ec2/ e certifique-se de que os grupos de segurança que você fornecerá ao Replicador tenham regras de saída para permitir o tráfego para os grupos de segurança do seu cluster de destino. Além disso, certifique-se de que os grupos de segurança do seu cluster de destino tenham regras de entrada que aceitem o tráfego proveniente dos grupos de segurança do replicador fornecidos para o destino.

     Para adicionar regras de entrada ao grupo de segurança do seu cluster de destino:

     1. No AWS console, acesse os detalhes do cluster de destino selecionando o nome do cluster.
     2. Selecione a guia Propriedades e, em seguida, role para baixo até o painel Configurações de rede para selecionar o nome do grupo de segurança aplicado.
     3. Acesse as regras de entrada e selecione Editar regras de entrada.
     4. Selecione Adicionar regra.
     5. Na coluna Tipo da nova regra, selecione TCP personalizado.
     6. Na coluna Intervalo de portas, digite`9098`. O MSK Replicator usa o controle de acesso do IAM para se conectar ao seu cluster que usa a porta 9098.
     7. Na coluna Origem, digite o nome do grupo de segurança que você fornecerá durante a criação do Replicador para o cluster de destino (pode ser o mesmo que o grupo de segurança do cluster de destino MSK) e selecione Salvar regras.

     Para adicionar regras de saída ao grupo de segurança do Replicator fornecido para o destino:

     1. No AWS console, acesse o grupo de segurança que você fornecerá durante a criação do Replicator para o destino.

2. Selecione a guia Propriedades e, em seguida, role para baixo até o painel Configurações de rede para selecionar o nome do grupo de segurança aplicado.
3. Acesse as regras de saída e selecione Editar regras de saída.
4. Selecione Adicionar regra.
5. Na coluna Tipo da nova regra, selecione TCP personalizado.
6. Na coluna Intervalo de portas, digite`9098`. O MSK Replicator usa o controle de acesso do IAM para se conectar ao seu cluster que usa a porta 9098.
7. Na coluna Origem, digite o nome do grupo de segurança do cluster de destino MSK e selecione Salvar regras.

> **Note**
>
> Como alternativa, se você não quiser restringir o tráfego usando seus grupos de segurança, poderá adicionar regras de entrada e saída que permitam Todo o Tráfego.
> 1. Selecione Adicionar regra.
> 2. Na coluna Tipo, escolha Todo o tráfego.
> 3. Na coluna Origem, digite `0.0.0.0/0` e selecione Salvar regras.

Definir configurações e permissões do replicador

1. No painel Configurações do replicador, especifique os tópicos que deseja replicar usando expressões regulares nas listas de permissão e proibição. Todos os tópicos são replicados por padrão.

   > **Note**
   >
   > O MSK Replicator replica somente até 750 tópicos em ordem ordenada. Se você precisar replicar mais tópicos, recomendamos criar um replicador separado. Acesse o Support Center do AWS console e crie um caso de suporte se precisar de suporte para mais de 750 tópicos por replicador. Você pode monitorar o número de tópicos que estão sendo replicados usando a métrica TopicCount "". Consulte Cota do Amazon MSK.

2. Por padrão, o MSK Replicator inicia a replicação a partir da compensação mais recente (mais recente) nos tópicos selecionados. Como alternativa, você pode iniciar a replicação a partir do deslocamento mais antigo (mais antigo) nos tópicos selecionados se quiser replicar os

dados existentes em seus tópicos. Depois que o replicador é criado, você não pode alterar essa configuração. Essa configuração corresponde ao `startingPosition`campo nas APIs de `CreateReplicator`solicitação e `DescribeReplicator`resposta.

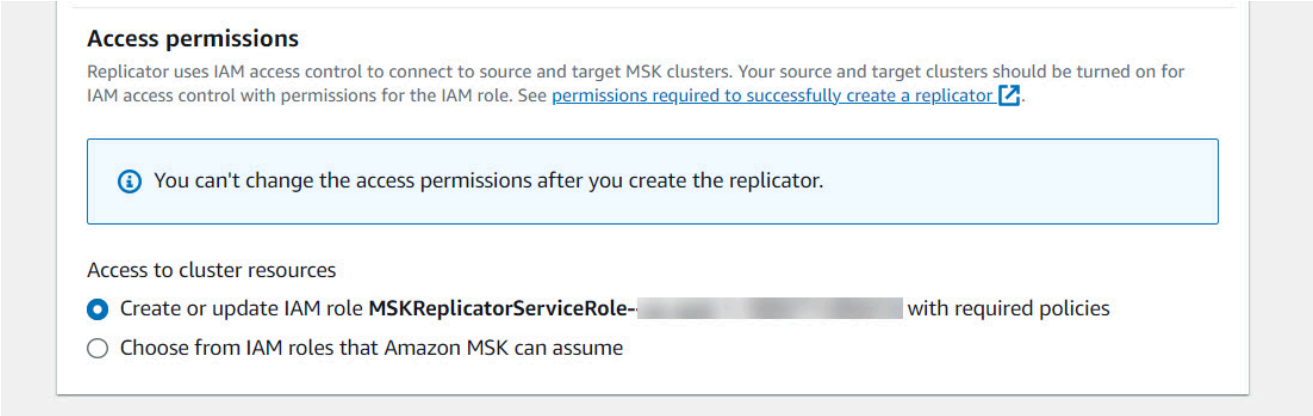
> **Note**
>
> O MSK Replicator atua como um novo consumidor para seu cluster de origem. Dependendo da quantidade de dados que você está replicando e da capacidade de consumo que você tem em seu cluster de origem, isso pode fazer com que outros consumidores em seu cluster de origem sejam limitados. Se você criar um conjunto de replicadores na posição inicial mais antiga, o MSK Replicator lerá uma explosão de dados no início, o que pode consumir toda a capacidade de consumo do cluster de origem. Depois que o replicador estiver atualizado, a taxa de consumo deverá diminuir para corresponder à taxa de transferência dos tópicos do cluster de origem. Se você estiver replicando da primeira posição, recomendamos que você gerencie a taxa de transferência do Replicator usando as cotas do Kafka para garantir que outros consumidores não sejam limitados.

3. Por padrão, o replicador do MSK copia todos os metadados, incluindo configurações de tópicos, listas de controle de acesso (ACLs) e deslocamentos de grupos de consumidores para um failover contínuo. Se você não estiver criando o replicador para failover, é possível optar por desativar uma ou mais dessas configurações disponíveis na seção Configurações adicionais.

> **Note**
>
> O replicador do MSK não replica ACLs de gravação, pois seus produtores não devem gravar diretamente no tópico replicado no cluster de destino. Seus produtores devem gravar no tópico local no cluster de destino após o failover. Para mais detalhes, consulte Executando um failover planejado para a região secundária AWS.

4. No painel Replicação do grupo de consumidores, especifique os grupos de consumidores que deseja replicar usando expressões regulares nas listas de permissão e proibição. Todos os grupos de consumidores são replicados por padrão.
5. No painel Compactação, você pode optar opcionalmente por compactar os dados gravados no cluster de destino. Se você for usar a compactação, recomendamos que use o mesmo método de compactação dos dados em seu cluster de origem.
6. No painel Permissões de acesso, execute uma das seguintes ações:

a. Selecione Criar ou atualizar a função do IAM com as políticas necessárias. O console do MSK anexará automaticamente as permissões e a política de confiança necessárias ao perfil de execução do serviço necessário para ler e gravar em seus clusters de origem e destino do MSK.



b. Forneça sua própria função do IAM selecionando Escolher entre as funções do IAM que o Amazon MSK pode assumir. Recomendamos que você anexe a política `AWSMSKReplicatorExecutionRole` gerenciada do IAM à sua função de execução do serviço, em vez de escrever sua própria política do IAM.

- Crie o perfil do IAM que o replicador usará para ler e gravar em seus clusters de origem e destino do MSK com o JSON abaixo como parte da política de confiança e o `AWSMSKReplicatorExecutionRole` anexo ao perfil. Na política de confiança, substitua o espaço reservado <yourAccountID> pelo ID efetivo da sua conta.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "kafka.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<yourAccountID>"
                }
            }
        }
```

```
    ]
}
```

7. No painel Tags do replicador, você pode, opcionalmente, atribuir tags ao recurso replicador do MSK. Para ter mais informações, consulte Atribuir tags a um cluster do Amazon MSK. Para um replicador do MSK entre regiões, as tags são sincronizadas automaticamente com a região remota quando o replicador é criado. Se você alterar as tags após a criação do replicador, a alteração não será sincronizada automaticamente com a região remota, então você precisará sincronizar manualmente as referências do replicador local e do replicador remoto.
8. Escolha Criar.

Se você quiser restringir a `kafka-cluster:WriteData` permissão, consulte a seção Criar políticas de autorização de Como funciona o controle de acesso do IAM para o Amazon MSK. Você precisará adicionar `kafka-cluster:WriteDataIdempotently` permissão ao cluster de origem e de destino.

A criação e transferência do replicador do MSK para o status RUNNING leva aproximadamente 30 minutos.

Se você criar um novo replicador do MSK para substituir um que você excluiu, o novo replicador iniciará a replicação a partir do último deslocamento.

Se o replicador do MSK tiver passado para o status FAILED, consulte a seção Solução de problemas do replicador do MSK.

Editar configurações do replicador do MSK

Você não pode alterar o cluster de origem, o cluster de destino ou a posição inicial do Replicador após a criação do MSK Replicator. No entanto, você pode editar outras configurações do Replicator, como tópicos e grupos de consumidores para replicar.

1. Faça login no AWS Management Console e abra o console do Amazon MSK em https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. No painel de navegação esquerdo, escolha Replicadores para exibir a lista de replicadores na conta e selecione o replicador do MSK que deseja editar.
3. Escolha a guia Properties (Propriedades).
4. Na seção Configurações do replicador, escolha Editar replicador.

5. Você pode editar as configurações do replicador do MSK alterando qualquer uma dessas configurações.

   - Especifique os tópicos que deseja replicar usando expressões regulares nas listas de permissão e proibição. Por padrão, o replicador do MSK copia todos os metadados, incluindo configurações de tópicos, listas de controle de acesso (ACLs) e deslocamentos de grupos de consumidores para um failover contínuo. Se você não estiver criando o replicador para failover, é possível optar por desativar uma ou mais dessas configurações disponíveis na seção Configurações adicionais.

     > **Note**
     >
     > O replicador do MSK não replica ACLs de gravação, pois seus produtores não devem gravar diretamente no tópico replicado no cluster de destino. Seus produtores devem gravar no tópico local no cluster de destino após o failover. Para mais detalhes, consulte Executando um failover planejado para a região secundária AWS.

   - Em Replicação do grupo de consumidores, é possível especificar os grupos de consumidores que deseja replicar usando expressões regulares nas listas de permissão e proibição. Todos os grupos de consumidores são replicados por padrão. Se as listas de permissão e proibição estiverem vazias, a replicação do grupo de consumidores será desativada.
   - No painel Tipo de compactação do destino, você pode optar por compactar ou não os dados gravados no cluster de destino. Se você for usar a compactação, recomendamos que use o mesmo método de compactação dos dados em seu cluster de origem.

6. Salve as alterações.

   A criação e transferência do replicador do MSK para o estado em execução leva aproximadamente 30 minutos. Se o replicador do MSK tiver passado para o status FAILED, consulte a seção de solução de problemas ???.

Excluir um replicador do MSK

Talvez seja necessário excluir um replicador do MSK se ele falhar na criação (status FAILED). Os clusters de origem e destino atribuídos a um replicador do MSK não podem ser alterados após a criação do replicador do MSK. Você pode excluir um replicador do MSK existente e criar um novo. Se você criar um novo replicador do MSK para substituir o excluído, o novo replicador iniciará a replicação com base na última compensação.

1. Na AWS região em que seu cluster de origem está localizado, faça login no AWS Management Console e abra o console do Amazon MSK em https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. No painel de navegação, selecione Replicadores.
3. Na lista de replicadores do MSK, selecione o que você deseja excluir e escolha Excluir.

Monitorar a replicação

Você pode usar https://console.aws.amazon.com/cloudwatch/ na região do cluster de destino para visualizar métricas para `ReplicationLatency`, `MessageLag` e `ReplicatorThroughput` no nível de tópico e no nível agregado para cada replicador do Amazon MSK. As métricas são visíveis abaixo ReplicatorNameno namespace “AWS/Kafka”. Você também pode ver as métricas `ReplicatorFailure`, `AuthError` e `ThrottleTime` para verificar se há problemas.

O console MSK exibe um subconjunto de CloudWatch métricas para cada replicador MSK. Na lista Replicadores do console, selecione o nome de um replicador e selecione a guia Monitoramento.

Métricas de replicador do MSK

As métricas a seguir descrevem as métricas de desempenho ou conexão do replicador do MSK.

AuthError as métricas não abrangem erros de autenticação em nível de tópico. Para monitorar os erros de autenticação em nível de tópico do MSK Replicator, monitore as métricas do Replicator e as ReplicationLatency métricas em nível de tópico do cluster de origem,. MessagesInPerSec Se um tópico ReplicationLatency cair para 0, mas o tópico ainda tiver dados sendo produzidos, isso indica que o replicador tem um problema de autenticação com o tópico. Verifique se o perfil do IAM de execução do serviço do replicador tem permissão suficiente para acessar o tópico.

Tipo de métrica	Métrica	Descrição	Dimensõ	Unidade	Granularidade métrica bruta	Estatística bruta de agregaçã métrica	
Performance	ReplicationLatency	O tempo necessário para	ReplicatorName	Milissegundos	Partition	Máximo	

Tipo de métrica	Métrica	Descrição	Dimensõ	Unidade	Granular dade métrica bruta	Estatísti ca bruta de agregaçã métrica	
		que os registros sejam replicado s da origem para o cluster de destino; a duração entre o tempo de produção do registro na origem e o tempo de replicação no destino. Se Replicati onLatency aumentar, verifique se os clusters têm partições suficientes para suportar a replicação. Pode ocorrer alta latência de replicaçã o quando a contagem de partições for muito baixa para	Replicato rName, Tópico	Milisegu ndos	Partition	Máximo	

Tipo de métrica	Métrica	Descrição	Dimensõ	Unidade	Granularidade métrica bruta	Estatística bruta de agregaçã métrica	
		um throughput alto.					

Tipo de métrica	Métrica	Descrição	Dimensõ	Unidade	Granular dade métrica bruta	Estatísti ca bruta de agregaçã métrica
Performa ce	MessageLag	Monitora a sincronização entre o MSK Replicator e o cluster de origem. MessageLa g indica o atraso entre as mensagens produzidas no cluster de origem e as mensagens consumidas pelo replicador. Não é o atraso entre o cluster de origem e o de destino. Mesmo que o cluster de origem esteja indisponí vel/interrompido, o replicado r terminará de gravar a mensagem consumida no cluster de destino. Depois	Replicato rName	Contage	Partition	Soma
			Replicato rName, Tópico	Contage	Partition	Soma

Tipo de métrica	Métrica	Descrição	Dimensõ	Unidade	Granularidade métrica bruta	Estatística bruta de agregaçã métrica	
		de uma interrupção, MessageLag mostra um aumento indicando o número de mensagens que o replicador está por trás do cluster de origem e isso pode ser monitorado até que o número de mensagens seja 0, mostrando que o replicador alcançou o cluster de origem.					

Tipo de métrica	Métrica	Descrição	Dimensõ	Unidade	Granular dade métrica bruta	Estatísti ca bruta de agregaçã métrica	
Performa ce	Replicato rThroughput	Número médio de bytes replicados por segundo. Se ReplicatorThroughput optar por um tópico, verifique KafkaClusterPingSuccessCount AuthError as métricas para garantir que o replicador possa se comunicar com os clusters, verifique as métricas do cluster para garantir que o cluster não esteja inativo.	ReplicatorName	BytesPer econd	Partition	Soma	
			ReplicatorName, Tópico	BytesPer econd	Partition	Soma	

Tipo de métrica	Métrica	Descrição	Dimensõ	Unidade	Granular dade métrica bruta	Estatísti ca bruta de agregaçã métrica
Depure	AuthError	O número de conexões com falha na autenticação por segundo. Se essa métrica estiver acima de 0, você poderá verificar se a política do perfil de execução do serviço para o replicado r é válida e garantir que não haja recusa de permissões definidas para as permissõe s do cluster. Com base na dimensão ClusterAlias, você pode identificar se o cluster de origem ou de destino está apresenta	Replicato rName, ClusterA ias	Contage	Operado	Soma

Tipo de métrica	Métrica	Descrição	Dimensõ	Unidade	Granularidade métrica bruta	Estatística bruta de agregaçã métrica	
		ndo erros de autenticação.					

Tipo de métrica	Métrica	Descrição	Dimensõ	Unidade	Granular dade métrica bruta	Estatísti ca bruta de agregaçã métrica	
Depure	ThrottleTime	O tempo médio em ms em que uma solicitaç ão passou por controle de utilização pelos agentes no cluster. Defina o controle de utilização para evitar que o replicador do MSK sobrecarr egue o cluster. Se essa métrica for 0, a latência de replicaçã o não for alta e o replicato rThroughput for o esperado, o controle de utilização estará funcionando conforme o esperado. Se essa métrica estiver acima de 0, você poderá	Replicato rName, ClusterA ias	Milisegu ndos	Operado	Máximo	

Tipo de métrica	Métrica	Descrição	Dimensõ	Unidade	Granularidade métrica bruta	Estatística bruta de agregaçã métrica	
		ajustar o controle de utilização adequadamente.					
Depure	ReplicatorFailure	O número de falhas que o replicador está enfrentando.	Replicato rName	Contage		Soma	

Tipo de métrica	Métrica	Descrição	Dimensõ	Unidade	Granular dade métrica bruta	Estatísti ca bruta de agregaçã métrica
Depure	KafkaClus terPingSu ccessCount	Indica a integrida de da conexão do replicador com o cluster do Kafka. Se esse valor for 1, a conexão está íntegra. Se o valor for 0 ou não houver nenhum ponto de dados, a conexão não está íntegra. Se o valor for 0, você poderá verificar as configurações de permissão de rede ou IAM para o cluster do Kafka. Com base na ClusterAlias dimensão, você pode identificar se essa métrica é para o cluster de origem ou de destino.	Replicato rName, ClusterA ias	Contage		Soma

Como usar a replicação para aumentar a resiliência de uma aplicação de streaming do Kafka em todas as regiões

Você pode usar o MSK Replicator para configurar topologias de cluster ativo-ativo ou ativo-passivo para aumentar a resiliência do seu aplicativo Apache Kafka em todas as regiões. AWS Em uma configuração ativa-ativa, os dois clusters do MSK estão atendendo ativamente leituras e gravações. Em uma configuração ativa-passiva, somente um cluster do MSK por vez estará atendendo ativamente dados de streaming, enquanto o outro cluster estará em espera.

Considerações para criar aplicações Apache Kafka em várias regiões

Seus consumidores devem ser capazes de reprocessar mensagens duplicadas sem impacto posterior. O MSK Replicator replica dados at-least-once que podem resultar em duplicatas no cluster em espera. Quando você muda para a AWS região secundária, seus consumidores podem processar os mesmos dados mais de uma vez. O replicador do MSK prioriza a cópia de dados em vez das compensações do consumidor para melhorar o desempenho. Após um failover, o consumidor pode começar a ler as compensações anteriores, resultando em processamento duplicado.

Produtores e consumidores também devem tolerar a perda mínima de dados. Como o MSK Replicator replica dados de forma assíncrona, quando a AWS região primária começa a apresentar falhas, não há garantia de que todos os dados sejam replicados para a região secundária. Você pode usar a latência de replicação para determinar o máximo de dados que não foram copiados para a região secundária.

Uso da topologia ativa-ativa vs. ativa-passiva de cluster

Uma topologia ativa-ativa de cluster oferece quase zero tempo de recuperação e a capacidade de sua aplicação de streaming operar simultaneamente em várias regiões da AWS . Quando um cluster em uma região está comprometido, as aplicações conectadas ao cluster na outra região continuam processando dados.

As configurações ativa-passiva são adequadas para aplicações que podem ser executadas em apenas uma região da AWS por vez ou quando você precisa de mais controle sobre a ordem de processamento de dados. As configurações ativa-passiva exigem mais tempo de recuperação do que as configurações ativa-ativa, pois você deve iniciar toda a configuração ativa-passiva, incluindo seus produtores e consumidores, na região secundária para retomar o streaming de dados após um failover.

Como criar uma configuração ativa-passiva de cluster do Kafka e nomenclatura replicada de tópicos

Para uma configuração ativa-passiva, recomendamos que você opere uma configuração semelhante de produtores, clusters MSK e consumidores (com o mesmo nome de grupo de consumidores) em duas regiões diferentes. AWS É importante que os dois clusters do MSK tenham capacidade idêntica de leitura e gravação para garantir a replicação confiável dos dados. Você precisa criar um replicador do MSK para copiar continuamente os dados do cluster primário para o cluster em espera. Você também precisa configurar seus produtores para gravar dados em tópicos em um cluster na mesma AWS região.

Para garantir que seus consumidores possam reiniciar o processamento de maneira confiável diretamente do cluster em espera, você precisa configurar seus consumidores para ler os dados dos tópicos usando um operador curinga “*”. Por exemplo, o MSK Replicator replica “topic1” do cluster primário para um novo tópico no cluster em espera chamado “< Alias>.topic1”. sourceKafkaCluster Por exemplo, você pode configurar seus produtores para gravar em “topic1” e seus consumidores para consumir usando “.*topic1” em ambas as regiões. Esse exemplo também incluiria um tópico como footopic1, portanto, ajuste o operador curinga de acordo com suas necessidades.

Quando fazer o failover para a região secundária AWS

Recomendamos que você monitore a latência de replicação na AWS região secundária usando o. CloudWatch Durante um evento de serviço na AWS região principal, a latência da replicação pode aumentar repentinamente. Se a latência continuar aumentando, use o AWS Service Health Dashboard para verificar se há eventos de serviço na AWS região principal. Se houver um evento, você pode fazer o failover para a AWS região secundária.

Executando um failover planejado para a região secundária AWS

Você pode realizar um failover planejado para testar a resiliência do seu aplicativo contra um evento inesperado em sua AWS região primária, que tem seu cluster MSK de origem. Um failover planejado não deve resultar em perda de dados.

1. Desligue todos os produtores e consumidores que se conectam ao seu cluster de origem.
2. Crie um novo replicador do MSK para replicar dados do seu cluster do MSK na região secundária para o seu cluster do MSK na região primária. Isso é necessário para copiar os dados que você gravará na região secundária de volta para a região primária, para que você possa fazer failback para a região primária após o término do evento inesperado.

3. Inicie produtores no cluster de destino na AWS região secundária.
4. Dependendo dos requisitos de ordenação de mensagens da aplicação, siga as etapas em uma das guias a seguir.

No message ordering

Se seu aplicativo não exigir a ordenação de mensagens, inicie consumidores na AWS região secundária que leiam os tópicos locais (por exemplo,`topic`) e replicados (por exemplo,`<sourceKafkaClusterAlias>.topic`) usando um operador curinga (por exemplo,. *tópico).

Message ordering

Se sua aplicação exigir a ordenação de mensagens, inicie os consumidores somente para os tópicos replicados no cluster de destino (p. ex., `<sourceKafkaClusterAlias>.topic`), mas não para os tópicos locais (p. ex., `topic`).

1. Aguarde até que todos os consumidores de tópicos replicados no cluster de destino do MSK concluam o processamento de todos os dados, para que o atraso do consumidor seja 0 e o número de registros processados também seja 0. Em seguida, interrompa os consumidores dos tópicos replicados no cluster de destino. Nesse ponto, todos os registros que foram replicados do cluster do MSK de origem para o cluster do MSK de destino foram consumidos.
2. Inicie consumidores para os tópicos locais (p. ex., `topic`) no cluster de destino do MSK.

Executando um failover não planejado para a região secundária AWS

Você pode realizar um failover não planejado quando há um evento de serviço na AWS região primária que tem seu cluster MSK de origem e você deseja redirecionar temporariamente seu tráfego para a AWS região secundária que tem seu cluster MSK de destino. Um failover não planejado pode resultar na perda de alguns dados.

1. Tente desligar todos os produtores e consumidores que se conectam ao cluster de origem do MSK na região primária. Isso pode falhar.
2. Inicie a conexão dos produtores com o cluster de destino do MSK na região secundária.
3. Dependendo dos requisitos de ordenação de mensagens da aplicação, siga as etapas em uma das guias a seguir.

No message ordering

Se seu aplicativo não exigir a ordenação de mensagens, inicie consumidores na AWS região de destino que leiam os tópicos locais (por exemplo`topic`) e replicados (por exemplo,`<sourceKafkaClusterAlias>.topic`) usando um operador curinga (por exemplo,). `.*topic`

Message ordering

1. Inicie os consumidores somente para os tópicos replicados no cluster de destino (p. ex., `<sourceKafkaClusterAlias>.topic`), mas não para os tópicos locais (p. ex., `topic`).
2. Aguarde até que todos os consumidores de tópicos replicados no cluster de destino do MSK concluam o processamento de todos os dados, para que o atraso do deslocamento seja 0 e o número de registros processados também seja 0. Em seguida, interrompa os consumidores dos tópicos replicados no cluster de destino. Nesse ponto, todos os registros que foram replicados do cluster do MSK de origem para o cluster do MSK de destino foram consumidos.
3. Inicie consumidores para os tópicos locais (p. ex., `topic`) no cluster de destino do MSK.

4. Depois que o evento de serviço terminar na região primária, crie um novo replicador MSK para replicar dados do seu cluster MSK na região secundária para o cluster MSK na região primária com a posição inicial do replicador definida como a mais antiga. Isso é necessário para copiar os dados que você gravará na região secundária de volta para a região primária, para que você possa fazer failback para a região primária após o término do evento de serviço. Se você não definir a posição inicial do Replicador como a mais antiga, todos os dados produzidos para o cluster na região secundária durante o evento de serviço na região primária não serão copiados de volta para o cluster na região primária.

Executando o failback para a região primária AWS

Você pode retornar à AWS região primária após o término do evento de serviço nessa região. O replicador do MSK ignora automaticamente os tópicos com o alias do cluster de origem como prefixo ao replicar dados de volta para a região primária durante o failback.

Se você seguiu as etapas de failover não planejadas, já deve ter criado o replicador de failback como parte da última etapa do failover da região primária para a secundária.

Se você não seguiu as etapas de failover não planejadas, depois que o evento de serviço terminar na região primária, crie um novo MSK Replicator para replicar dados do seu cluster MSK na região secundária para o cluster MSK na região primária, com a posição inicial do replicador definida como a mais antiga. Isso é necessário para copiar os dados que você gravará na região secundária de volta para a região primária, para que você possa fazer failback para a região primária após o término do evento de serviço. Se você não alterar a posição inicial do Replicador do valor padrão mais recente para o mais antigo, todos os dados produzidos para o cluster na região secundária durante o evento de serviço na região primária não serão copiados de volta para o cluster na região primária.

Você deve iniciar as etapas de failback somente depois que a replicação do cluster na região secundária para o cluster na região primária for recuperada e a MessageLag métrica CloudWatch estiver próxima de 0. Um failback planejado não deve resultar em nenhuma perda de dados.

1. Feche todos os produtores e consumidores que se conectam ao cluster do MSK na região secundária.
2. Para a topologia ativa-passiva, exclua o replicador que está replicando dados do cluster na região secundária para a região primária. Você não precisa excluir o replicador para a topologia ativa-ativa.
3. Inicie a conexão dos produtores com o cluster do MSK na região primária.
4. Dependendo dos requisitos de ordenação de mensagens da aplicação, siga as etapas em uma das guias a seguir.

   No message ordering

   Se seu aplicativo não exigir a ordenação de mensagens, inicie consumidores na AWS região primária que leiam os tópicos locais (por exemplo,`topic`) e replicados (por exemplo,`<sourceKafkaClusterAlias>.topic`) usando um operador curinga (por exemplo,). `.*topic` Os consumidores de tópicos locais (p. ex., tópico) retomarão com base no último deslocamento que consumiram antes do failover. Se houver algum dado não processado antes do failover, ele será processado agora. No caso de um failover planejado, esse registro não deverá existir.

   Message ordering

   1. Inicie os consumidores somente para os tópicos replicados na região primária (p. ex., `<sourceKafkaClusterAlias>.topic`), mas não para os tópicos locais (p. ex., `topic`).

2. Aguarde até que todos os consumidores de tópicos replicados na região primária do cluster concluam o processamento de todos os dados, para que o atraso do deslocamento seja 0 e o número de registros processados também seja 0. Em seguida, interrompa os consumidores dos tópicos replicados no cluster na região primária. Nesse ponto, todos os registros que foram produzidos na região secundária após o failover terão sido consumidos na região primária.
3. Inicie consumidores para os tópicos locais (p. ex., `topic`) no cluster na região primária.

5. Verifique se o replicador existente do cluster na região primária para o cluster na região secundária está no estado RUNNING e funcionando conforme o esperado usando as métricas `ReplicatorThroughput` de latência e.

Como criar uma configuração ativa-ativa usando o replicador do MSK

Siga estas etapas para configurar a topologia ativa-ativa entre o cluster A de origem do MSK e o cluster B de destino do MSK.

1. Crie um replicador do MSK com o cluster A do MSK como origem e o cluster B do MSK como destino.
2. Depois que o replicador do MSK acima for criado com sucesso, crie um replicador com o cluster B como origem e o cluster A como destino.
3. Crie dois conjuntos de produtores, cada um gravando dados ao mesmo tempo no tópico local (p. ex., “topic”) no cluster na mesma região do produtor.
4. Crie dois conjuntos de consumidores, cada um lendo dados usando uma assinatura curinga (como”. *tópico”) do cluster MSK na mesma AWS região do consumidor. Dessa forma, seus consumidores lerão automaticamente os dados produzidos localmente na região com base no tópico local (p. ex., `topic`), bem como os dados replicados de outra região no tópico com o prefixo`<sourceKafkaClusterAlias>.topic`. Esses dois conjuntos de consumidores devem ter IDs diferentes de grupo de consumidores para que os deslocamentos de grupos de consumidores não sejam sobrepostos quando o replicador do MSK os copiarem para o outro cluster.

Solução de problemas do replicador do MSK

Tópicos

As informações a seguir podem ajudar você a solucionar problemas que você pode vir a enfrentar com o replicador do MSK. Você também pode publicar seu problema no AWS re:Post.

O estado do replicador do MSK vai de CREATING para FAILED

Aqui estão algumas causas comuns de falha na criação do replicador do MSK.

1. Verifique se os grupos de segurança que você forneceu para a criação do replicador na seção do cluster de destino têm regras de saída para permitir o tráfego para os grupos de segurança do seu cluster de destino. Além disso, verifique se os grupos de segurança do seu cluster de destino têm regras de entrada que aceitem o tráfego proveniente dos grupos de segurança fornecidos para a criação do replicador na seção do cluster de destino. Consulte Escolher seu cluster de destino.
2. Se você estiver criando o replicador para replicação entre regiões, verifique se o cluster de origem tem conectividade multi-VPC ativada para o método de autenticação IAM Access Control. Consulte Conectividade privada multi-VPC do Amazon MSK em uma única região. Verifique também se a política de cluster está configurada no cluster de origem para que o replicador do MSK possa se conectar ao cluster de origem. Consulte Etapa 1: preparar o cluster de origem do Amazon MSK.
3. Verifique se a perfil do IAM que você forneceu durante a criação do replicador do MSK tem as permissões necessárias para ler e gravar nos clusters de origem e destino. Além disso, verifique se a perfil do IAM tem permissões para gravar em tópicos. Consulte Definir configurações e permissões do replicador
4. Verifique se suas ACLs de rede não estão bloqueando a conexão entre o replicador do MSK e seus clusters de origem e destino.
5. É possível que os clusters de origem ou de destino não estivessem totalmente disponíveis quando o replicador do MSK tentou se conectar a eles. Isso pode decorrer de níveis excessivos de carga, uso do disco ou da CPU, o que faz com que o replicador não consiga se conectar aos agentes. Corrija o problema com os agentes e repita a criação do replicador.

Após realizar as validações acima, crie o replicador do MSK novamente.

O replicador do MSK parece preso no estado CREATING

Às vezes a criação do replicador do MSK pode levar até 30 minutos. Aguarde 30 minutos e verifique o estado do replicador novamente.

O replicador do MSK não está replicando dados ou replicando apenas dados parciais

Siga estas etapas para solucionar problemas de replicação de dados.

1. Verifique se seu replicador não está enfrentando nenhum erro de autenticação usando a AuthError métrica fornecida pelo MSK Replicator em. CloudWatch Se essa métrica estiver acima de 0, verifique se a política do perfil do IAM que você forneceu para o replicador é válida e se não há recusa de permissões definidas para as permissões do cluster. Com base na dimensão ClusterAlias, você pode identificar se o cluster de origem ou de destino está apresentando erros de autenticação.
2. Verifique se seus clusters de origem e destino não estão enfrentando problemas. É possível que o replicador não consiga se conectar ao seu cluster de origem ou de destino. Isso pode acontecer devido a muitas conexões, disco com capacidade total ou alto uso da CPU.
3. Verifique se seus clusters de origem e destino podem ser acessados pelo MSK Replicator usando a métrica em. KafkaClusterPingSuccessCount CloudWatch Com base na dimensão ClusterAlias, você pode identificar se o cluster de origem ou de destino está apresentando erros de autenticação. Se essa métrica for 0 ou não tiver ponto de dados, a conexão não está íntegra. Você deve verificar as permissões de rede e do perfil do IAM que o replicador do MSK está usando para se conectar aos seus clusters.
4. Verifique se o replicador não está apresentando falhas devido à falta de permissões em nível de tópico usando a métrica em. ReplicatorFailure CloudWatch Se essa métrica estiver acima de 0, verifique o perfil do IAM que você forneceu para obter permissões no nível de tópico.
5. Verifique se a expressão regular que você forneceu na lista de permissões ao criar o replicador corresponde aos nomes dos tópicos que você deseja replicar. Além disso, verifique se os tópicos não estão sendo excluídos da replicação devido a uma expressão regular na lista de proibição.
6. Observe que pode levar até 30 segundos para que o Replicator detecte e crie os novos tópicos ou partições de tópicos no cluster de destino. Qualquer mensagem produzida no tópico de origem antes da criação do tópico no cluster de destino não será replicada se a posição inicial do replicador for a mais recente (padrão). Como alternativa, você pode iniciar a replicação a partir

do primeiro deslocamento nas partições de tópicos do cluster de origem se quiser replicar as mensagens existentes sobre seus tópicos no cluster de destino. Consulte Definir configurações e permissões do replicador.

Os deslocamentos de mensagens no cluster de destino são diferentes do cluster de origem

Como parte da replicação de dados, o MSK Replicator consome mensagens do cluster de origem e as produz para o cluster de destino. Isso pode fazer com que as mensagens tenham diferentes deslocamentos nos clusters de origem e de destino. No entanto, se você ativou a sincronização de offsets de grupos de consumidores durante a criação do Replicator, o MSK Replicator traduzirá automaticamente os offsets enquanto copia os metadados para que, após o failover para o cluster de destino, seus consumidores possam retomar o processamento de perto de onde pararam no cluster de origem.

O MSK Replicator não está sincronizando grupos de consumidores, offsets ou o grupo de consumidores não existe no cluster de destino

Siga estas etapas para solucionar problemas de replicação de metadados.

1. Verifique se sua replicação de dados está funcionando conforme o esperado. Se não, consulte O replicador do MSK não está replicando dados ou replicando apenas dados parciais.
2. Verifique se a expressão regular que você forneceu na lista de permissões ao criar o Replicador corresponde aos nomes dos grupos de consumidores que você deseja replicar. Além disso, verifique se os grupos de consumidores não estão sendo excluídos da replicação devido a uma expressão regular na lista de negação.
3. Verifique se o MSK Replicator criou o tópico no cluster de destino. Pode levar até 30 segundos para que o Replicator detecte e crie os novos tópicos ou partições de tópicos no cluster de destino. Qualquer mensagem produzida no tópico de origem antes da criação do tópico no cluster de destino não será replicada se a posição inicial do replicador for a mais recente (padrão). Se seu grupo de consumidores no cluster de origem tiver consumido somente as mensagens que não foram replicadas pelo MSK Replicator, o grupo de consumidores não será replicado para o cluster de destino. Depois que o tópico for criado com sucesso no cluster de destino, o MSK Replicator começará a replicar mensagens recém-gravadas no cluster de origem para o destino. Quando seu grupo de consumidores começar a ler essas mensagens da origem, o MSK Replicator replicará automaticamente o grupo de consumidores para o cluster de destino. Como alternativa, você

pode iniciar a replicação a partir do primeiro deslocamento nas partições de tópicos do cluster de origem se quiser replicar as mensagens existentes sobre seus tópicos no cluster de destino. Consulte Definir configurações e permissões do replicador.

> **Note**
>
> O MSK Replicator otimiza a sincronização offset de grupos de consumidores para seus consumidores no cluster de origem, que estão lendo de uma posição mais próxima ao final da partição do tópico. Se seus grupos de consumidores estiverem atrasados no cluster de origem, você poderá observar um atraso maior para esses grupos de consumidores no destino em comparação com a origem. Isso significa que, após o failover para o cluster de destino, seus consumidores reprocessarão mais mensagens duplicadas. Para reduzir esse atraso, seus consumidores no cluster de origem precisariam se atualizar e começar a consumir a partir da ponta do stream (final da partição do tópico). À medida que seus consumidores se atualizarem, o MSK Replicator reduzirá automaticamente o atraso.

A latência de replicação é alta ou continua aumentando

Aqui estão algumas causas comuns da alta latência de replicação.

1. Verifique se você tem o número certo de partições nos clusters de origem e destino do MSK. Ter poucas ou muitas partições pode afetar o desempenho. Para obter orientação sobre como escolher o número de partições, consulte Práticas recomendadas para usar o replicador do MSK. A tabela a seguir mostra o número mínimo recomendado de partições para obter o throughput desejado com o replicador do MSK.

   Throughput e número mínimo recomendado de partições

Throughput (MB/s)	Número mínimo necessário de partições
50	167
100	334
250	833
500	1666

Throughput (MB/s)	Número mínimo necessário de partições
1000	3333

2. Verifique se você tem capacidade suficiente de leitura e gravação em seus clusters de origem e destino do MSK para atender o tráfego de replicação. O replicador do MSK atua como consumidor do cluster de origem (saída) e como produtor do cluster de destino (entrada). Portanto, você deve provisionar a capacidade do cluster para atender ao tráfego de replicação, além de outros tráfegos em seus clusters. Consulte ??? para obter orientação sobre como dimensionar seus clusters do MSK.
3. A latência de replicação pode variar para clusters MSK em diferentes pares de AWS regiões de origem e destino, dependendo da distância geográfica entre os clusters. Por exemplo, a latência de replicação geralmente é menor ao replicar entre clusters nas regiões da Europa (Irlanda) e Europa (Londres) em comparação com a replicação entre clusters nas regiões da Europa (Irlanda) e Ásia-Pacífico (Sydney).
4. Verifique se o replicador não está sendo submetido ao controle de utilização devido às cotas excessivamente agressivas definidas em seus clusters de origem ou de destino. Você pode usar a ThrottleTime métrica fornecida pelo MSK Replicator CloudWatch para ver o tempo médio em milissegundos em que uma solicitação foi limitada por agentes em seu cluster de origem/destino. Se essa métrica estiver acima de 0, você deve ajustar as cotas do Kafka para reduzir o controle de utilização de modo que o replicador possa se atualizar. Consulte Como gerenciar o throughput do replicador do MSK usando cotas do Kafka para obter informações sobre o gerenciamento de cotas do Kafka para o replicador.
5. ReplicationLatency e MessageLag pode aumentar quando uma AWS região se degrada. Use o AWS Service Health Dashboard para verificar se há um evento de serviço do MSK na região do seu cluster primário do MSK. Se houver um evento de serviço, você poderá redirecionar temporariamente as leituras e gravações da aplicação para a outra região.

Práticas recomendadas para usar o replicador do MSK

Esta seção aborda práticas recomendadas e estratégias de implementação comuns para usar o replicador do MSK.

Tópicos

- Como gerenciar o throughput do replicador do MSK usando cotas do Kafka
- Definir o período de retenção do cluster

Como gerenciar o throughput do replicador do MSK usando cotas do Kafka

Como o replicador do MSK atua como consumidor do seu cluster de origem, a replicação pode fazer com que outros consumidores passem por controle de utilização em seu cluster de origem. A quantidade de controle de utilização depende da capacidade de leitura que você tem no cluster de origem e do throughput dos dados que você está replicando. Recomendamos que você provisione capacidade idêntica para seus clusters de origem e de destino e leve em conta o throughput de replicação ao calcular a capacidade necessária.

Você também pode definir cotas do Kafka para o replicador em seus clusters de origem e destino a fim de controlar a capacidade que o replicador do MSK pode usar. Recomenda-se usar uma cota de largura de banda da rede. Uma cota de largura de banda da rede define um limite de taxa de bytes, definido como bytes por segundo, para um ou mais clientes que compartilham uma cota. Essa cota é definida por agente.

Siga estas etapas para aplicar uma cota.

1. Recupere a string do servidor bootstrap para o cluster de origem. Consulte Como obter agentes de bootstrap para um cluster do Amazon MSK.
2. Recupere o Service execution role (SER – Perfil de execução de serviço) usado pelo replicador do MSK. Esse é o SER que você usou para uma solicitação `CreateReplicator`. Você também pode extrair o SER da DescribeReplicator resposta de um replicador existente.
3. Usando as ferramentas de CLI do Kafka, execute o comando a seguir no cluster de origem.

```
./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. Após executar o comando acima, verifique se a métrica `ReplicatorThroughput` não ultrapassa a cota que você definiu.

Observe que todos estarão sujeitos a essa cota se você reutilizar um perfil de execução de serviço entre vários replicadores do MSK. Se você quiser manter cotas separadas por replicador, use perfis de execução de serviço separados.

Para obter mais informações sobre o uso da autenticação do IAM no MSK com cotas, consulte Clusters Apache Kafka multilocação no Amazon MSK com controle de acesso do IAM e cotas do Kafka: parte 1.

> **Warning**
>
> Definir uma taxa de consumer_byte_rate extremamente baixa pode fazer com que seu replicador do MSK atue de maneiras inesperadas.

Definir o período de retenção do cluster

Você pode definir o período de retenção de log para clusters do MSK provisionados e com tecnologia sem servidor. O período recomendado de retenção é de 7 dias. Consulte Alterações na configuração de cluster ou Configuração de cluster do MSK Serverless.

Estados de cluster

A tabela a seguir descreve os estados possíveis de um cluster e descrevem seus significados. Ela também descreve quais ações você pode e não pode realizar quando um cluster estiver em um desses estados. Para descobrir o estado de um cluster, você pode acessar o AWS Management Console. Você também pode usar o comando describe-cluster-v2 ou a operação DescribeClusterV2 para descrever o cluster. A descrição de um cluster inclui seu estado.

Estado de cluster	Significado e ações possíveis
ACTIVE	Você pode produzir e consumir dados. Você também pode realizar AWS CLI operações e APIs do Amazon MSK no cluster.
CRIANDO	O Amazon MSK está configurando o cluster. Você deve esperar que o cluster alcance o estado ATIVO antes de poder usá-lo para produzir ou consumir dados ou para executar a API do Amazon MSK ou AWS CLI operações neles.
EXCLUINDO	O cluster está sendo excluído. Você não pode usá-lo para produzir ou consumir dados. Você também não pode executar a API do Amazon MSK ou AWS CLI operações nela.
COM FALHA	O processo de criação ou exclusão do cluster falhou. Você não pode usar o cluster para produzir ou consumir dados. Você pode excluir o cluster, mas não pode executar a API Amazon MSK nem AWS CLI atualizar operações nele.
HEALING	O Amazon MSK está executando uma operação interna, como a substituição de um agente não íntegro. Por exemplo, talvez o agente não esteja respondendo. Você ainda

Estado de cluster	Significado e ações possíveis
	pode usar o cluster para produzir e consumir dados. No entanto, você não pode realizar operações de API ou AWS CLI atualizar a API do Amazon MSK no cluster até que ele retorne ao estado ATIVO.
MAINTENANCE	O Amazon MSK está realizando operações de manutenção de rotina no cluster. Essas operações de manutenção incluem a aplicação de patches de segurança. Você ainda pode usar o cluster para produzir e consumir dados. No entanto, você não pode realizar operações de API ou AWS CLI atualizar a API do Amazon MSK no cluster até que ele retorne ao estado ATIVO.
REBOOTING_BROKER	O Amazon MSK está reiniciando um agente. Você ainda pode usar o cluster para produzir e consumir dados. No entanto, você não pode realizar operações de API ou AWS CLI atualizar a API do Amazon MSK no cluster até que ele retorne ao estado ATIVO.
ATUALIZANDO	Uma API ou AWS CLI operação do Amazon MSK iniciada pelo usuário está atualizando o cluster. Você ainda pode usar o cluster para produzir e consumir dados. No entanto, você não pode realizar nenhuma operação adicional de API ou AWS CLI atualização do Amazon MSK no cluster até que ele retorne ao estado ATIVO.

Segurança no Amazon Managed Streaming for Apache Kafka

A segurança na nuvem AWS é a maior prioridade. Como AWS cliente, você se beneficia de uma arquitetura de data center e rede criada para atender aos requisitos das organizações mais sensíveis à segurança.

A segurança é uma responsabilidade compartilhada entre você AWS e você. O modelo de responsabilidade compartilhada descreve isso como segurança da nuvem e segurança na nuvem:

- Segurança da nuvem — AWS é responsável por proteger a infraestrutura que executa AWS os serviços na AWS nuvem. AWS também fornece serviços que você pode usar com segurança. Auditores terceirizados testam e verificam regularmente a eficácia de nossa segurança como parte dos Programas de Conformidade Programas de AWS de . Para saber mais sobre os programas de conformidade aplicáveis ao Amazon Managed Streaming for Apache Kafka, consulte Serviços da Amazon Web Services no escopo por programa de conformidade.
- Segurança na nuvem — Sua responsabilidade é determinada pelo AWS serviço que você usa. Você também é responsável por outros fatores, incluindo a confidencialidade de seus dados, os requisitos da sua empresa e as leis e regulamentos aplicáveis.

Esta documentação ajuda você a entender como aplicar o modelo de responsabilidade compartilhada ao usar o Amazon MSK. Os tópicos a seguir mostram como configurar o Amazon MSK para atender aos seus objetivos de segurança e compatibilidade. Saiba também como usar outros serviços da Amazon Web Services que ajudam você a monitorar e proteger seus recursos do Amazon MSK.

Tópicos

- Resiliência no Amazon Managed Streaming for Apache Kafka
- Segurança de infraestrutura no Amazon Managed Streaming for Apache Kafka

Proteção de dados no Amazon Managed Streaming for Apache Kafka

O modelo de responsabilidade AWS compartilhada O modelo se aplica à proteção de dados no Amazon Managed Streaming for Apache Kafka. Conforme descrito neste modelo, AWS é responsável por proteger a infraestrutura global que executa todos os Nuvem AWS. Você é responsável por manter o controle sobre seu conteúdo hospedado nessa infraestrutura. Você também é responsável pelas tarefas de configuração e gerenciamento de segurança dos Serviços da AWS que usa. Para obter mais informações sobre a privacidade de dados, consulte as Perguntas Frequentes sobre Privacidade de Dados.. Para obter mais informações sobre a proteção de dados na Europa, consulte a postagem do blog AWS Shared Responsibility Model and GDPR no Blog de segurança da AWS .

Para fins de proteção de dados, recomendamos que você proteja Conta da AWS as credenciais e configure usuários individuais com AWS IAM Identity Center ou AWS Identity and Access Management (IAM). Dessa maneira, cada usuário receberá apenas as permissões necessárias para cumprir suas obrigações de trabalho. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA) com cada conta.
- Use SSL/TLS para se comunicar com os recursos. AWS Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Configure a API e o registro de atividades do usuário com AWS CloudTrail.
- Use soluções de AWS criptografia, juntamente com todos os controles de segurança padrão Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sigilosos armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-2 ao acessar AWS por meio de uma interface de linha de comando ou de uma API, use um endpoint FIPS. Para ter mais informações sobre endpoints do FIPS disponíveis, consulte Federal Information Processing Standard (FIPS) 140-2.

É altamente recomendável que nunca sejam colocadas informações de identificação confidenciais, como endereços de e-mail dos seus clientes, em marcações ou campos de formato livre, como um campo Nome. Isso inclui quando você trabalha com o Amazon MSK ou outro Serviços da AWS usando o console, a API ou os AWS SDKs. AWS CLI Quaisquer dados inseridos em tags ou campos de texto de formato livre usados para nomes podem ser usados para logs de faturamento ou de diagnóstico. Se você fornecer um URL para um servidor externo, recomendemos fortemente que não sejam incluídas informações de credenciais no URL para validar a solicitação a esse servidor.

Tópicos

Criptografia do Amazon MSK

O Amazon MSK fornece opções de criptografia de dados que você pode usar para atender a requisitos rigorosos de gerenciamento de dados. É necessário renovar a cada 13 meses os certificados que o Amazon MSK usa para criptografia. O Amazon MSK renova automaticamente esses certificados para todos os clusters. Ele define o estado do cluster como `MAINTENANCE` quando inicia a operação de atualização do certificado. Quando a atualização é concluída, o estado é definido novamente como `ACTIVE`. Enquanto um cluster está no estado `MAINTENANCE`, você pode continuar a produzir e consumir dados, mas não pode executar nenhuma operação de atualização nele.

Criptografia em repouso

O Amazon MSK se integra ao AWS Key Management Service (KMS) para oferecer uma criptografia transparente no lado do servidor. O Amazon MSK sempre criptografa seus dados em repouso. Ao criar um cluster do MSK, você pode especificar a AWS KMS key que deseja que o Amazon MSK use para criptografar seus dados em repouso. Se você não especificar uma chave do KMS, o Amazon MSK criará uma Chave gerenciada pela AWS para você e a usará em seu nome. Para obter mais informações sobre como usar as chaves KMS, consulte AWS KMS keys o AWS Key Management Service Guia do desenvolvedor.

Criptografia em trânsito

O Amazon MSK usa TLS 1.2. Por padrão, ele criptografa os dados em trânsito entre os agentes do seu cluster do MSK. É possível substituir esse padrão no momento de criação do cluster.

Para a comunicação entre clientes e agentes, é necessário especificar uma destas três configurações:

- Permitir somente dados criptografados por TLS. Essa é a configuração padrão.
- Permitir dados não criptografados e dados criptografados por TLS.
- Permitir apenas dados não criptografados.

Os corretores do Amazon MSK usam certificados públicos AWS Certificate Manager . Portanto, qualquer armazenamento confiável que confie no Amazon Trust Services também confia nos agentes do Amazon MSK.

Embora seja altamente recomendável habilitar a criptografia em trânsito, isso pode acrescentar sobrecarga à CPU e alguns milissegundos de latência. Contudo, a maioria dos casos de uso não é afetada por essas diferenças, e a magnitude do impacto depende da configuração do cluster, dos clientes e do perfil de uso.

Como começo a usar a criptografia?

Ao criar um cluster do MSK, você pode especificar configurações de criptografia no formato JSON. Veja um exemplo a seguir.

```
{
   "EncryptionAtRest": {
       "DataVolumeKMSKeyId": "arn:aws:kms:us-east-1:123456789012:key/abcdabcd-1234-
abcd-1234-abcd123e8e8e"
    },
   "EncryptionInTransit": {
        "InCluster": true,
        "ClientBroker": "TLS"
    }
}
```

Para `DataVolumeKMSKeyId`, é possível especificar uma chave gerenciada pelo cliente ou a Chave gerenciada pela AWS para o MSK na sua conta (`alias/aws/kafka`). Se você não especificar`EncryptionAtRest`, o Amazon MSK ainda criptografa seus dados em repouso sob o. Chave gerenciada pela AWS Para determinar qual chave o cluster está usando, envie uma solicitação `GET` ou invoque a operação de API `DescribeCluster`.

Para `EncryptionInTransit`, o valor padrão de `InCluster` é verdadeiro, mas será possível defini-lo como falso se não quiser que o Amazon MSK criptografe seus dados conforme eles passam pelos agentes.

Para especificar o modo de criptografia para dados em trânsito entre clientes e agentes, defina `ClientBroker` como um dos três valores: `TLS`, `TLS_PLAINTEXT` ou `PLAINTEXT`.

Como especificar configurações de criptografia ao criar um cluster

1. Salve o conteúdo do exemplo anterior em um arquivo e dê ao arquivo qualquer nome que desejar. Por exemplo, nomeie-o como `encryption-settings.json`.
2. Execute o comando `create-cluster` e use a opção `encryption-info` para apontar para o arquivo onde você salvou a configuração JSON. Veja um exemplo a seguir. Substitua *`{YOUR MSK VERSION}`* por uma versão que corresponda à versão do cliente Apache Kafka. Para obter informações sobre como encontrar a versão de cluster do MSK, consulte To find the version of your MSK cluster. Esteja ciente de que usar uma versão do cliente Apache Kafka que não seja igual à sua versão de cluster do MSK pode resultar em corrupção, perda e tempo de inatividade dos dados do 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
   ```

   Veja a seguir um exemplo de uma resposta bem-sucedida após a execução desse comando.

   ```
   {
       "ClusterArn": "arn:aws:kafka:us-east-1:123456789012:cluster/SecondTLSTest/
   abcdabcd-1234-abcd-1234-abcd123e8e8e",
       "ClusterName": "ExampleClusterName",
       "State": "CREATING"
   }
   ```

Como testar a criptografia por TLS

1. Crie uma máquina de cliente seguindo as orientações em the section called “Etapa 3: criar uma máquina cliente”.
2. Instale o Apache Kafka na máquina de cliente.

3. Neste exemplo, o armazenamento confiável da JVM para se comunicar com o cluster do MSK. Para fazer isso, crie primeiramente uma pasta chamada `/tmp` na máquina cliente. Depois, acesse a pasta `bin` da instalação do Apache Kafka e execute o comando a seguir. (Seu caminho da JVM pode ser diferente.)

```
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. Enquanto ainda estiver na pasta `bin` da instalação do Apache Kafka na máquina cliente, crie um arquivo de texto chamado `client.properties` com o conteúdo a seguir.

```
security.protocol=SSL
ssl.truststore.location=/tmp/kafka.client.truststore.jks
```

5. Execute o comando a seguir em uma máquina que tenha o AWS CLI instalado, substituindo *`clusterARN`* pelo ARN do seu cluster.

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

   Um resultado bem-sucedido tem a aparência a seguir. Salve este resultado porque você precisará dele na próxima etapa.

```
{
    "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. Execute o comando a seguir, *`BootstrapBrokerStringTls`*substituindo-o por um dos endpoints do broker que você obteve na etapa anterior.

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

7. Abra uma nova janela de comando e conecte-se à mesma máquina cliente. Depois, execute o comando a seguir para criar um consumidor de console.

```
<path-to-your-kafka-installation>/bin/kafka-console-consumer.sh --bootstrap-
server BootstrapBrokerStringTls --consumer.config client.properties --topic
 TLSTestTopic
```

8. Na janela do produtor, digite uma mensagem de texto seguida de um retorno e procure a mesma mensagem na janela do consumidor. O Amazon MSK criptografou essa mensagem em trânsito.

Para obter mais informações sobre como configurar clientes do Apache Kafka para trabalhar com dados criptografados, consulte Configurar clientes do Kafka.

Autenticação e autorização para API do Amazon MSK

AWS Identity and Access Management (IAM) é uma ferramenta AWS service (Serviço da AWS) que ajuda o administrador a controlar com segurança o acesso aos AWS recursos. Os administradores do IAM controlam quem pode ser autenticado (conectado) e autorizado (ter permissões) para utilizar os recursos do Amazon MSK. O IAM é um AWS service (Serviço da AWS) que você pode usar sem custo adicional.

Esta página descreve como você pode usar o IAM para controlar quem pode realizar operações do Amazon MSK em seu cluster. Para obter informações sobre como controlar quem pode realizar operações do Apache Kafka em seu cluster, consulte the section called “Autenticação e autorização para API do Apache Kafka”.

Tópicos

- Como o Amazon MSK funciona com o IAM
- Exemplos de política baseada em identidade do Amazon MSK
- Uso de perfis vinculados a serviço para o Amazon MSK
- AWS políticas gerenciadas para o Amazon MSK
- Solução de problemas de identidade e acesso da Amazon MSK

Como o Amazon MSK funciona com o IAM

Antes de usar o IAM para gerenciar o acesso ao Amazon MSK, você deve entender quais recursos do IAM estão disponíveis para uso com o Amazon MSK. Para obter uma visão de alto nível de

como o Amazon MSK e outros AWS serviços funcionam com o IAM, consulte AWS Serviços que funcionam com o IAM no Guia do usuário do IAM.

Tópicos

- Políticas baseadas em identidade do Amazon MSK
- Políticas baseadas em recurso do Amazon MSK
- AWS políticas gerenciadas
- Autorização baseada em tags do Amazon MSK
- Perfis do IAM para o Amazon MSK

Políticas baseadas em identidade do Amazon MSK

Com as políticas baseadas em identidade do IAM, é possível especificar ações ou recursos permitidos ou negados, assim como as condições sob as quais as ações são permitidas ou negadas. O Amazon MSK é compatível com ações, chaves de condição e recursos específicos. Para conhecer todos os elementos usados em uma política JSON, consulte Referência de elementos de política JSON do IAM no Guia do usuário do IAM.

Ações

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos, e em que condições.

O elemento `Action` de uma política JSON descreve as ações que você pode usar para permitir ou negar acesso em uma política. As ações de política geralmente têm o mesmo nome da operação de AWS API associada. Existem algumas exceções, como ações somente de permissão, que não têm uma operação de API correspondente. Algumas operações também exigem várias ações em uma política. Essas ações adicionais são chamadas de ações dependentes.

Incluem ações em uma política para conceder permissões para executar a operação associada.

As ações de política no Amazon MSK usam o seguinte prefixo antes da ação: `kafka:`. Por exemplo, para conceder permissão a alguém para descrever um cluster do MSK com a operação de API `DescribeCluster` do Amazon MSK, inclua a ação `kafka:DescribeCluster` na política. As instruções de política devem incluir um elemento `Action` ou `NotAction`. O Amazon MSK define seu próprio conjunto de ações que descrevem as tarefas que você pode executar com esse serviço.

Para especificar várias ações em uma única instrução, separe-as com vírgulas, como segue:

```
"Action": ["kafka:action1", "kafka:action2"]
```

Você também pode especificar várias ações usando caracteres curinga (*). Por exemplo, para especificar todas as ações que começam com a palavra `Describe`, inclua a seguinte ação:

```
"Action": "kafka:Describe*"
```

Para ver uma lista de ações do Amazon MSK, consulte Ações, recursos e chaves de condição do Amazon Managed Streaming for Apache Kafka no Guia do usuário do IAM.

Recursos

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos, e em que condições.

O elemento de política JSON `Resource` especifica o objeto ou os objetos aos quais a ação se aplica. As instruções devem incluir um elemento `Resource` ou `NotResource`. Como prática recomendada, especifique um recurso usando seu nome do recurso da Amazon (ARN). Isso pode ser feito para ações que oferecem compatibilidade com um tipo de recurso específico, conhecido como permissões em nível de recurso.

Para ações que não oferecem compatibilidade com permissões em nível de recurso, como operações de listagem, use um curinga (*) para indicar que a instrução se aplica a todos os recursos.

```
"Resource": "*"
```

O recurso de instância do Amazon MSK tem o seguinte ARN:

```
arn:${Partition}:kafka:${Region}:${Account}:cluster/${ClusterName}/${UUID}
```

Para obter mais informações sobre o formato dos ARNs, consulte Amazon Resource Names (ARNs) e AWS Service Namespaces.

Por exemplo, para especificar a instância `CustomerMessages` na instrução, use o seguinte ARN:

```
"Resource": "arn:aws:kafka:us-east-1:123456789012:cluster/CustomerMessages/abcd1234-
abcd-dcba-4321-a1b2abcd9f9f-2"
```

Para especificar todas as instâncias que pertencem a uma conta específica, use o caractere curinga (*):

```
"Resource": "arn:aws:kafka:us-east-1:123456789012:cluster/*"
```

Algumas ações do Amazon MSK, como as usadas para a criação de recursos, não podem ser executadas em um recurso específico. Nesses casos, você deve utilizar o caractere curinga (*).

```
"Resource": "*"
```

Para especificar vários recursos em uma única instrução, separe os ARNs com vírgulas.

```
"Resource": ["resource1", "resource2"]
```

Para ver uma lista dos tipos de recursos do Amazon MSK e seus ARNs, consulte Recursos definidos pelo Amazon Managed Streaming for Apache Kafka no Guia do usuário do IAM. Para saber com quais ações você pode especificar o ARN de cada recurso, consulte Ações definidas pelo Amazon Managed Streaming for Apache Kafka.

Chaves de condição

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos, e em que condições.

O elemento `Condition` (ou bloco `Condition`) permite que você especifique condições nas quais uma instrução estiver em vigor. O elemento `Condition` é opcional. É possível criar expressões condicionais que usem agentes de condição, como “igual a” ou “menor que”, para fazer a condição da política corresponder aos valores na solicitação.

Se você especificar vários elementos `Condition` em uma instrução ou várias chaves em um único `Condition` elemento, a AWS os avaliará usando uma operação lógica `AND`. Se você especificar vários valores para uma única chave de condição, AWS avalia a condição usando uma `OR` operação lógica. Todas as condições devem ser atendidas antes que as permissões da instrução sejam concedidas.

Você também pode usar variáveis de espaço reservado ao especificar condições. Por exemplo, é possível conceder a um usuário do IAM permissão para acessar um atributo somente se ele estiver marcado com seu nome de usuário do IAM. Para obter mais informações, consulte Elementos da política do IAM: variáveis e tags no Guia do usuário do IAM.

AWS suporta chaves de condição globais e chaves de condição específicas do serviço. Para ver todas as chaves de condição AWS globais, consulte as chaves de contexto de condição AWS global no Guia do usuário do IAM.

O Amazon MSK define seu próprio conjunto de chaves de condição e também é compatível com o uso de algumas chaves de condição globais. Para ver todas as chaves de condição AWS globais, consulte Chaves de contexto de condição AWS global no Guia do usuário do IAM.

Para ver uma lista das chaves de condição do Amazon MSK, consulte Chaves de condição para o Amazon Managed Streaming for Apache Kafka no Guia do usuário do IAM. Para saber com quais ações e recursos você pode usar uma chave de condição, consulte Ações definidas pelo Amazon Managed Streaming for Apache Kafka.

Exemplos

Para visualizar exemplos de políticas baseadas em identidade do Amazon MSK, consulte Exemplos de política baseada em identidade do Amazon MSK.

Políticas baseadas em recurso do Amazon MSK

O Amazon MSK é compatível com uma política de cluster (também conhecida como política baseada em recurso) para uso com clusters do Amazon MSK. Você pode usar uma política de cluster para definir quais entidades principais do IAM têm permissões entre contas para configurar a conectividade privada com seu cluster do Amazon MSK. Quando usada com a autenticação de cliente do IAM, você também pode usar a política de cluster para definir de modo granular as permissões do plano de dados do Kafka para os clientes conectados.

Para ver um exemplo de como configurar uma política de cluster, consulte Etapa 2: anexar uma política de cluster ao cluster do MSK.

AWS políticas gerenciadas

Autorização baseada em tags do Amazon MSK

É possível anexar tags a clusters do Amazon MSK. Para controlar o acesso baseado em tags, forneça informações sobre as tags no elemento de condição de uma política usando as `kafka:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* ou chaves de condição `aws:TagKeys`. Para obter mais informações sobre a atribuição de tags a recursos do Amazon MSK, consulte the section called “Atribuir tags a um cluster”.

Para visualizar um exemplo de política baseada em identidades para limitar o acesso a um cluster baseado nas tags desse cluster, consulte Como acessar clusters do Amazon MSK com base em tags.

Perfis do IAM para o Amazon MSK

Um perfil do IAM é uma entidade dentro da sua conta da Amazon Web Services que tem permissões específicas.

Usar credenciais temporárias com o Amazon MSK

É possível usar credenciais temporárias para fazer login com federação, assumir um perfil do IAM ou assumir um perfil entre contas. Você obtém credenciais de segurança temporárias chamando operações de AWS STS API, como AssumeRoleou GetFederationToken.

A Amazon MSK é compatível com o uso de credenciais temporárias.

Funções vinculadas a serviço

Os perfis vinculados a serviço permitem que os serviços da Amazon Web Services acessem recursos em outros serviços para concluir uma ação em seu nome. Os perfis vinculados a serviço aparecem em sua conta do IAM e são de propriedade do serviço. Um administrador do pode visualizar, mas não pode editar as permissões para funções vinculadas ao serviço.

O Amazon ECS MSK é compatível com perfis vinculados a serviço. Para obter detalhes sobre como criar ou gerenciar perfis vinculados a serviço do Amazon MSK, consulte the section called “Funções vinculadas a serviço”.

Exemplos de política baseada em identidade do Amazon MSK

Por padrão, usuários e perfis do IAM não têm permissão para executar ações de API do Amazon MSK. Um administrador deve criar as políticas do IAM que concedam aos usuários e aos perfis permissões para executar operações de API específicas nos recursos especificados que precisam. O administrador deve anexar essas políticas aos usuários ou grupos do IAM que exigem essas permissões.

Para saber como criar uma política baseada em identidade do IAM usando esses exemplos de documentos de política JSON, consulte Criar políticas na guia JSON no Guia do usuário do IAM.

Tópicos

- Melhores práticas de política

- Permitir que usuários visualizem suas próprias permissões
- Acessar um cluster do Amazon MSK
- Como acessar clusters do Amazon MSK com base em tags

Melhores práticas de política

As políticas baseadas em identidade determinam se alguém pode criar, acessar ou excluir recursos do Amazon MSK em sua conta. Essas ações podem incorrer em custos para seus Conta da AWS. Ao criar ou editar políticas baseadas em identidade, siga estas diretrizes e recomendações:

- Comece com as políticas AWS gerenciadas e avance para as permissões de privilégios mínimos — Para começar a conceder permissões aos seus usuários e cargas de trabalho, use as políticas AWS gerenciadas que concedem permissões para muitos casos de uso comuns. Eles estão disponíveis no seu Conta da AWS. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo AWS cliente que sejam específicas para seus casos de uso. Para obter mais informações, consulte Políticas gerenciadas pela AWS ou Políticas gerenciadas pela AWS para funções de trabalho no Guia do Usuário do IAM.
- Aplique permissões de privilégio mínimo –- ao definir permissões com as políticas do IAM, conceda apenas as permissões necessárias para executar uma tarefa. Você faz isso definindo as ações que podem ser executadas em atributos específicos sob condições específicas, também conhecidas como permissões de privilégio mínimo. Para obter mais informações sobre como usar o IAM para aplicar permissões, consulte Políticas e permissões no IAM no Guia do Usuário do IAM.
- Use condições nas políticas do IAM para restringir ainda mais o acesso –- você pode adicionar uma condição às políticas para limitar o acesso a ações e recursos. Por exemplo, você pode gravar uma condição de política para especificar que todas as solicitações devem ser enviadas usando SSL. Você também pode usar condições para conceder acesso às ações de serviço se elas forem usadas por meio de uma ação específica AWS service (Serviço da AWS), como AWS CloudFormation. Para obter mais informações, consulte Elementos da política JSON do IAM: Condição no Guia do usuário do IAM.
- Use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais –- o IAM Access Analyzer valida as políticas novas e existentes para que elas sigam a linguagem de política do IAM (JSON) e as práticas recomendadas do IAM. O IAM Access Analyzer oferece mais de 100 verificações de política e recomendações acionáveis para ajudá-lo a criar políticas seguras e funcionais. Para obter mais informações, consulte Validação de políticas do IAM Access Analyzer no Guia do Usuário do IAM.

- Exigir autenticação multifator (MFA) — Se você tiver um cenário que exija usuários do IAM ou um usuário root, ative Conta da AWS a MFA para obter segurança adicional. Para exigir MFA quando as operações de API forem chamadas, adicione condições de MFA às suas políticas. Para obter mais informações, consulte Configuração de acesso à API protegido por MFA no Guia do Usuário do IAM.

Para obter mais informações sobre as práticas recomendadas do IAM, consulte Práticas Recomendadas de Segurança no IAM no Guia do Usuário do IAM.

Permitir que usuários visualizem suas próprias permissões

Este exemplo mostra como criar uma política que permita que os usuários do IAM visualizem as políticas gerenciadas e em linha anexadas a sua identidade de usuário. Essa política inclui permissões para concluir essa ação no console ou programaticamente usando a API AWS CLI ou 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": "*"
        }
    ]
}
```

Acessar um cluster do Amazon MSK

Neste exemplo, você vai permitir que um usuário do IAM na sua conta da Amazon Web Services acesse um dos seus cluster, `purchaseQueriesCluster`. Esta política permite que o usuário descreva o cluster, obtenha seus agentes de bootstrap, liste seus nós de agente e o atualize.

```
{
   "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"
      }
   ]
}
```

Como acessar clusters do Amazon MSK com base em tags

Você pode usar condições em sua política baseada em identidade para controlar o acesso aos recursos do Amazon MSK com base em tags. Este exemplo mostra como você pode criar uma política que permita que o usuário descreva o cluster, obtenha seus agentes de bootstrap, liste seus nós de agente, atualize-o e exclua-o. No entanto, a permissão será concedida somente se a tag de cluster `Owner` tiver o valor do nome desse usuário.

```
{
  "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}"
        }
      }
    }
  ]
}
```

É possível anexar essa política aos usuários do IAM na sua conta. Se um usuário chamado `richard-roe` tentar atualizar um cluster do MSK, o cluster deverá estar marcado como `Owner=richard-roe` ou `owner=richard-roe`. Caso contrário, ele terá o acesso negado. A chave da tag de condição `Owner` corresponde a `Owner` e a `owner` porque os nomes das chaves de condição não fazem distinção entre maiúsculas e minúsculas. Para obter mais informações, consulte IAM JSON Policy Elements: Condition (Elementos da política JSON do IAM: Condição) no Guia do usuário do IAM.

Uso de perfis vinculados a serviço para o Amazon MSK

O Amazon MSK usa funções vinculadas a serviços AWS Identity and Access Management (IAM). Um perfil vinculado a serviço é um tipo especial de perfil do IAM vinculado diretamente ao Amazon MSK. As funções vinculadas ao serviço são predefinidas pelo Amazon MSK e incluem todas as permissões que o serviço exige para chamar outros AWS serviços em seu nome.

Um perfil vinculado a serviço facilita a configuração do Amazon MSK porque você não precisa adicionar as permissões necessárias manualmente. O Amazon MSK define as permissões dos perfis vinculados a serviço. A menos que definido de outra forma, somente o Amazon MSK pode assumir seus perfis. As permissões definidas incluem a política de confiança e a política de permissões, e essa política não pode ser anexada a nenhuma outra entidade do IAM.

Para obter informações sobre outros serviços compatíveis com perfis vinculados a serviço, consulte Serviços da Amazon Web Services compatíveis com o IAM e procure os serviços que exibem Sim na coluna Perfil vinculado a serviço. Escolha um Sim com um link para visualizar a documentação da função vinculada a esse serviço.

Tópicos

Permissões de perfil vinculado a serviço para o Amazon MSK

O Amazon MSK usa o perfil vinculado a serviço chamado AWSServiceRoleForKafka. O Amazon MSK usa esse perfil para acessar seus recursos e realizar operações como:

- `*NetworkInterface`: criar e gerenciar interfaces de rede na conta do cliente que tornem os agentes de cluster acessíveis aos clientes na VPC do cliente.
- `*VpcEndpoints`— gerencie endpoints de VPC na conta do cliente que tornam os agentes de cluster acessíveis aos clientes que usam a VPC do cliente. AWS PrivateLink O Amazon MSK usa permissões para `DescribeVpcEndpoints`, `ModifyVpcEndpoint` e `DeleteVpcEndpoints`.
- `secretsmanager`— gerencie as credenciais do cliente com AWS Secrets Manager.
- `GetCertificateAuthorityCertificate`: recuperar o certificado para sua autoridade de certificação privada.

Essa função vinculada ao serviço é anexada à seguinte política gerenciada: `KafkaServiceRolePolicy`. Para atualizações desta política, consulte KafkaServiceRolePolicy.

A função vinculada ao serviço AWSServiceRoleForKafka confia nos seguintes serviços para aceitar a função:

- `kafka.amazonaws.com`

A política de permissões do perfil permite que o Amazon MSK execute as seguintes ações nos recursos.

```
{
 "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_*"
     }
    }
   }
  ]
}
```

Você deve configurar permissões para que uma entidade do IAM (por exemplo, um usuário, grupo ou função) crie, edite ou exclua uma função vinculada a serviço. Para mais informações, consulte Permissões de perfil vinculado ao serviço no Guia do usuário do IAM.

Criação de um perfil vinculado a serviço para o Amazon MSK

Não é necessário criar uma função vinculada ao serviço manualmente. Quando você cria um cluster do Amazon MSK na AWS Management Console, na ou na AWS API AWS CLI, o Amazon MSK cria a função vinculada ao serviço para você.

Se excluir essa função vinculada ao serviço e precisar criá-la novamente, você poderá usar esse mesmo processo para recriar a função em sua conta. Quando você cria um cluster do Amazon MSK, o Amazon MSK cria um perfil vinculado a serviço para você novamente.

Edição de um perfil vinculado a serviço do Amazon MSK

O Amazon MSK não permite que você edite o perfil vinculado a serviço do AWSServiceRoleForKafka. Depois que criar um perfil vinculado ao serviço, você não poderá alterar o nome do perfil, pois várias entidades podem fazer referência a ele. No entanto, será possível editar a descrição do perfil usando o IAM. Para obter mais informações, consulte Editar uma função vinculada a serviço no Guia do usuário do IAM.

Regiões compatíveis com perfis vinculados a serviço do Amazon MSK

O Amazon MSK é compatível com perfis vinculados a serviço em todas as regiões nas quais o serviço esteja disponível. Para mais informações, consulte Regiões e endpoints da AWS.

AWS políticas gerenciadas para o Amazon MSK

Uma política AWS gerenciada é uma política autônoma criada e administrada por AWS. AWS as políticas gerenciadas são projetadas para fornecer permissões para muitos casos de uso comuns, para que você possa começar a atribuir permissões a usuários, grupos e funções.

Lembre-se de que as políticas AWS gerenciadas podem não conceder permissões de privilégio mínimo para seus casos de uso específicos porque estão disponíveis para uso de todos os AWS clientes. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo cliente da específicas para seus casos de uso.

Você não pode alterar as permissões definidas nas políticas AWS gerenciadas. Se AWS atualizar as permissões definidas em uma política AWS gerenciada, a atualização afetará todas as identidades principais (usuários, grupos e funções) às quais a política está anexada. AWS é mais provável que atualize uma política AWS gerenciada quando uma nova AWS service (Serviço da AWS) é lançada ou novas operações de API são disponibilizadas para serviços existentes.

Para obter mais informações, consulte Políticas gerenciadas pela AWS no Guia do usuário do IAM.

AWS política gerenciada: AmazonMSK FullAccess

Essa política concede permissões administrativas que permitem que a entidade principal tenha acesso total a todas as ações do Amazon MSK. As permissões nessa política são agrupadas da seguinte forma:

- As permissões do Amazon MSK permitem todas as ações do Amazon MSK.
- **`Amazon EC2`**permissões — nesta política, é necessário validar os recursos passados em uma solicitação de API. Isso serve para garantir que o Amazon MSK seja capaz de usar adequadamente os recursos com um cluster. O restante das permissões do Amazon EC2 nesta política permitem que o Amazon MSK crie AWS os recursos necessários para possibilitar a conexão com seus clusters.
- **`AWS KMS`**permissões — são usadas durante chamadas de API para validar os recursos passados em uma solicitação. Elas são necessárias para que o Amazon MSK consiga usar a chave transmitida com o cluster do Amazon MSK.
- **`CloudWatch Logs, Amazon S3, and Amazon Data Firehose`**permissões — são necessárias para que o Amazon MSK possa garantir que os destinos de entrega de logs sejam acessíveis e que sejam válidos para uso do log do agente.

- **IAM**permissões — são necessárias para que o Amazon MSK possa criar uma função vinculada ao serviço em sua conta e para permitir que você passe uma função de execução de serviço para o 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 política gerenciada: ReadOnly AmazonMSK Access

Essa política concede permissões de acesso somente leitura que permitem que os usuários visualizem informações no Amazon MSK. As entidades principais com essa política anexada não podem fazer nenhuma atualização ou excluir recursos existentes, nem criar novos recursos do Amazon MSK. Por exemplo, entidades principais com essas permissões podem visualizar a lista de clusters e configurações associadas à conta, mas não podem alterar a configuração ou as definições de nenhum cluster. As permissões nessa política são agrupadas da seguinte forma:

- **`Amazon MSK`**permissões — permitem que você liste os recursos do Amazon MSK, descreva-os e obtenha informações sobre eles.
- **`Amazon EC2`**permissões — são usadas para descrever a Amazon VPC, sub-redes, grupos de segurança e ENIs associados a um cluster.
- **`AWS KMS`**permissão — é usada para descrever a chave associada ao 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 política gerenciada: KafkaServiceRolePolicy

Você não pode se vincular KafkaServiceRolePolicy às suas entidades do IAM. Essa política é anexada a um perfil vinculado a serviço que permite que o Amazon MSK realize ações como gerenciar endpoints da VPC (conectores) em clusters do MSK, gerenciar interfaces de rede e gerenciar credenciais de cluster com o AWS Secrets Manager. Para ter mais informações, consulte the section called “Funções vinculadas a serviço”.

AWS política gerenciada: AWSMSKReplicatorExecutionRole

A `AWSMSKReplicatorExecutionRole` política concede permissões ao replicador Amazon MSK para replicar dados entre clusters MSK. As permissões nessa política são agrupadas da seguinte forma:

- **`cluster`**— Concede ao Amazon MSK Replicator permissões para se conectar ao cluster usando a autenticação do IAM. Também concede permissões para descrever e alterar o cluster.
- **`topic`**— Concede ao Amazon MSK Replicator permissões para descrever, criar e alterar um tópico e alterar a configuração dinâmica do tópico.
- **`consumer group`**— Concede ao Amazon MSK Replicator permissões para descrever e alterar grupos de consumidores, ler e gravar datas de um cluster MSK e excluir tópicos internos criados pelo replicador.

```
{
 "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/*/*"
     ]
    }
   ]
}
```

Atualizações do Amazon MSK para políticas AWS gerenciadas

Veja detalhes sobre as atualizações das políticas AWS gerenciadas do Amazon MSK desde que esse serviço começou a rastrear essas alterações.

Alteração	Descrição	Data
WriteDataIdempotently permissão adicionada a AWSMSKReplicatorExecutionRole — Atualização de uma política existente	O Amazon MSK adicionou WriteDataIdempotently permissão à AWSMSKReplicatorExecutionRole política para oferecer suporte à replicação de dados entre clusters MSK.	12 de março de 2024
AWSMSKReplicatorExecutionRole – Nova política	O Amazon MSK adicionou uma AWSMSKReplicatorExecutionRole política para dar suporte ao Amazon MSK Replicator.	4 de dezembro de 2023
AmazonMSK FullAccess — Atualização de uma política existente	O Amazon MSK adicionou permissões para compatibilidade com o replicador do Amazon MSK.	28 de setembro de 2023
KafkaServiceRolePolicy: atualizar para uma política existente	O Amazon MSK adicionou permissões para compatibilidade com conectividade privada multi-VPC.	8 de março de 2023
AmazonMSK FullAccess — Atualização de uma política existente	O Amazon MSK adicionou novas permissões do Amazon EC2 para possibilitar a conexão a um cluster.	30 de novembro de 2021
AmazonMSK FullAccess — Atualização de uma política existente	O Amazon MSK adicionou uma nova permissão para permitir a descrição das tabelas de rotas do Amazon EC2.	19 de novembro de 2021

Alteração	Descrição	Data
O Amazon MSK passou a monitorar alterações	A Amazon MSK começou a monitorar as mudanças em suas políticas AWS gerenciadas.	19 de novembro de 2021

Solução de problemas de identidade e acesso da Amazon MSK

Use as informações a seguir para ajudar a diagnosticar e corrigir problemas comuns que você pode encontrar ao trabalhar com o Amazon MSK e o IAM.

Tópicos

- Não tenho autorização para executar uma ação no Amazon MSK

Não tenho autorização para executar uma ação no Amazon MSK

Se isso AWS Management Console indicar que você não está autorizado a realizar uma ação, entre em contato com o administrador para obter ajuda. Caso seu administrador seja a pessoa que forneceu suas credenciais de início de sessão.

O exemplo de erro a seguir ocorre quando o usuário do IAM `mateojackson` tenta usar o console para excluir um cluster, mas não tem permissões `kafka:DeleteCluster`.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 kafka:DeleteCluster on resource: purchaseQueriesCluster
```

Neste caso, Mateo pede ao administrador para atualizar suas políticas para permitir a ele o acesso ao recurso `purchaseQueriesCluster` usando a ação `kafka:DeleteCluster`.

Autenticação e autorização para API do Apache Kafka

É possível usar o IAM para autenticar clientes e permitir ou proibir ações do Apache Kafka. Como alternativa, você pode usar TLS ou SASL/SCRAM para autenticar clientes, além de ACLs do Apache Kafka para permitir ou proibir ações.

Para obter informações sobre como controlar quem pode realizar operações do Amazon MSK em seu cluster, consulte the section called “Autenticação e autorização para API do Amazon MSK”.

Tópicos

- Controle de acesso do IAM
- Autenticação TLS mútua
- Autenticação de credenciais de login com Secrets Manager AWS
- ACLs do Apache Kafka

Controle de acesso do IAM

O controle de acesso do IAM para o Amazon MSK permite que você gerencie a autenticação e a autorização para seu cluster do MSK. Isso elimina a necessidade de usar um mecanismo para autenticação e outro para autorização. Por exemplo, quando um cliente tenta gravar em seu cluster, o Amazon MSK usa o IAM para verificar se esse cliente é uma identidade autenticada e também se ele está autorizado a produzir para seu cluster. O controle de acesso do IAM funciona para clientes Java e não Java, incluindo clientes Kafka escritos em Python, Go e .NET. JavaScript

O Amazon MSK registra eventos de acesso para que você possa auditá-los. Para ter mais informações, consulte the section called “CloudTrail eventos”.

Para viabilizar o controle de acesso do IAM, o Amazon MSK faz pequenas modificações no código-fonte do Apache Kafka. Essas modificações não causarão uma diferença perceptível na sua experiência com o Apache Kafka.

Important

O controle de acesso do IAM não se aplica aos ZooKeeper nós do Apache. Para obter informações sobre como você pode controlar o acesso a esses nós, consulte the section called “Controlando o acesso ao Apache ZooKeeper”.

Important

A configuração `allow.everyone.if.no.acl.found` do Apache Kafka não tem efeito se seu cluster usar o controle de acesso do IAM.

> **Important**
>
> Você pode invocar as APIs de ACL do Apache Kafka para um cluster do MSK que use o controle de acesso do IAM. No entanto, as ACLs do Apache Kafka não têm efeito na autorização para funções do IAM. Você deve usar políticas do IAM para controlar o acesso de perfis do IAM.

Funcionamento do controle de acesso do IAM para o Amazon MSK

Para usar o controle de acesso do IAM para o Amazon MSK, execute as etapas a seguir, descritas em mais detalhes no restante desta seção.

- the section called “Crie um cluster que use o controle de acesso do IAM”
- the section called “Configurar clientes para controle de acesso do IAM”
- the section called “Criar políticas de autorização”
- the section called “Obter os agente de bootstrap para controle de acesso do IAM”

Crie um cluster que use o controle de acesso do IAM

Esta seção explica como você pode usar a AWS Management Console, a API ou a AWS CLI para criar um cluster que usa o controle de acesso do IAM. Para obter informações sobre como ativar o controle de acesso do IAM para um cluster existente, consulte the section called “Atualizar a segurança”.

Use o AWS Management Console para criar um cluster que usa o controle de acesso do IAM

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. Selecione Criar cluster.
3. Escolha Criar cluster com configurações personalizadas.
4. Na seção Autenticação, escolha Controle de acesso do IAM.
5. Preencha o restante do fluxo de trabalho para criar um cluster.

Use a API ou a AWS CLI para criar um cluster que usa o controle de acesso do IAM

- Para criar um cluster com o controle de acesso IAM ativado, use a CreateClusterAPI ou o comando da CLI create-cluster e passe o seguinte JSON para o parâmetro:.

`ClientAuthentication`	`"ClientAuthentication": { "Sasl": { "Iam": { "Enabled": true } } }`

Configurar clientes para controle de acesso do IAM

Para permitir que os clientes se comuniquem com um cluster do MSK que use o controle de acesso do IAM, você pode usar um dos seguintes mecanismos:

- Configuração de cliente que não seja Java usando o mecanismo SASL_OAUTHBEARER
- Configuração de cliente Java usando o mecanismo SASL_OAUTHBEARER ou o mecanismo AWS_MSK_IAM

Usar o mecanismo SASL_OAUTHBEARER para configurar o IAM

1. Edite o arquivo de configuração client.properties usando como guia a sintaxe destacada no exemplo de cliente Python Kafka abaixo. As alterações das configurações são semelhantes em outros idiomas.

```
#!/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. Baixe a biblioteca auxiliar para o idioma de configuração escolhido e siga as instruções na seção Getting started na página inicial desta biblioteca de idiomas.
   - 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
   - Go: 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: o suporte de SASL_OAUTHBEARER para Java está disponível por meio do arquivo jar `aws-msk-iam-auth`

Usar o mecanismo AWS_MSK_IAM personalizado do MSK para configurar o IAM

1. Adicione o seguinte ao arquivo `client.properties`. Substitua *<PATH_TO_TRUST_STORE_FILE>* pelo caminho totalmente qualificado para o arquivo de armazenamento confiável no cliente.

> **Note**
>
> Se você não quiser usar um certificado específico, poderá remover `ssl.truststore.location=`*<PATH_TO_TRUST_STORE_FILE>* do seu arquivo `client.properties`. Se você não especificar um valor para `ssl.truststore.location`, o processo Java usará o certificado padrão.

```
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
```

Para usar um perfil nomeado que você criou para AWS credenciais, inclua `awsProfileName="your profile name";` no arquivo de configuração do cliente. Para obter informações sobre perfis nomeados, consulte Perfis nomeados na AWS CLI documentação.

2. Baixe o arquivo JAR aws-msk-iam-auth estável mais recente e coloque-o no caminho da classe. Se você usa o Maven, adicione a seguinte dependência, ajustando o número da versão conforme necessário:

```
<dependency>
    <groupId>software.amazon.msk</groupId>
    <artifactId>aws-msk-iam-auth</artifactId>
    <version>1.0.0</version>
</dependency>
```

O plug-in do cliente do Amazon MSK é de código aberto sob a licença do Apache 2.0.

Criar políticas de autorização

Anexe uma política de autorização ao perfil do IAM correspondente ao cliente. Em uma política de autorização, você especifica quais ações permitir ou proibir para o perfil. Se seu cliente estiver em uma instância do Amazon EC2, associe a política de autorização ao perfil do IAM para essa instância do Amazon EC2. Como alternativa, você pode configurar seu cliente para usar um perfil nomeado e, em seguida, associar a política de autorização ao perfil desse perfil nomeado. the section called "Configurar clientes para controle de acesso do IAM" descreve como configurar um cliente para usar um perfil nomeado.

Para obter informações sobre como criar uma política do IAM, consulte Criar políticas do IAM.

Veja a seguir um exemplo de política de autorização para um cluster chamado MyTestCluster. Para entender a semântica dos elementos `Action` e `Resource`, consulte the section called "Semântica de ações e recursos".

> **Important**
>
> As alterações que você faz em uma política do IAM são refletidas imediatamente nas APIs do IAM e na AWS CLI . No entanto, a implementação da alteração da política pode levar um

tempo considerável. Na maioria dos casos, as mudanças na política entram em vigor em menos de um minuto. Às vezes, as condições da rede podem aumentar o atraso.

```
{
    "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/*"
            ]
        }
    ]
}
```

Para saber como criar uma política com elementos de ação que correspondam aos casos de uso comuns do Apache Kafka, como produzir e consumir dados, consulte the section called “Casos de uso comuns”.

Para as versões 2.8.0 e superiores do Kafka, a permissão WriteDataIdempotently está obsoleta (KIP-679). `enable.idempotence = true` é usado por padrão. Portanto, para as versões 2.8.0 e superiores do Kafka, o IAM não oferece a mesma funcionalidade das ACLs do Kafka. Não é possível atribuir `WriteDataIdempotently` a um tópico apenas fornecendo acesso `WriteData` a esse tópico. Isso não afeta o caso quando `WriteData` é fornecido para TODOS os tópicos. Nesse caso, `WriteDataIdempotently` é permitido. Isso se deve às diferenças na implementação da lógica do IAM em relação à maneira como as ACLs do Kafka são implementadas.

Para contornar isso, recomendamos o uso de uma política semelhante ao exemplo abaixo:

```
{
    "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"
            ]
        }
```

```
    ]
}
```

Nesse caso, `WriteData` permite gravações em `TestTopic`, enquanto `WriteDataIdempotently` permite gravações idempotentes no cluster. É importante observar que `WriteDataIdempotently` é uma permissão no nível de cluster. Não é possível usá-la no nível de tópico. Se `WriteDataIdempotently` estiver restrita ao nível do tópico, essa política não funcionará.

Obter os agente de bootstrap para controle de acesso do IAM

Consulte the section called “Como obter os agentes de bootstrap”.

Semântica de ações e recursos

Esta seção explica a semântica dos elementos de ação e recurso que você pode usar em uma política de autorização do IAM. Para visualizar um exemplo de política, consulte the section called “Criar políticas de autorização”.

Ações

A tabela a seguir lista as ações que você pode incluir em uma política de autorização ao usar o controle de acesso do IAM para o Amazon MSK. Ao incluir uma ação da coluna Ação da tabela em sua política de autorização, você também deve incluir as ações correspondentes da coluna Ações obrigatórias.

Ação	Descrição	Ações necessárias	Recursos necessários do	Aplicável a clusters com a tecnologia sem servidor
`kafka-cluster:Connect`	Concede permissão para se conectar e se autenticar no cluster.	Nenhum	cluster	Sim
`kafka-cluster:Desc`	Concede permissão para descrever	`kafka-cluster:Connect`	cluster	Sim

Ação	Descrição	Ações necessárias	Recursos necessários do	Aplicável a clusters com a tecnologia sem servidor
`ribeCluster`	vários aspectos do cluster, equivalente à ACL DESCRIBE CLUSTER do Apache Kafka.			

`kafka-cluster:AlterCluster`	Concede permissão para alterar vários aspectos do cluster, equivalente à ACL ALTER CLUSTER do Apache Kafka.	`kafka-cluster:Connect`
		
		`kafka-cluster:DescribeCluster`
			cluster	Não

`kafka-cluster:DescribeClusterDynamicConfiguration`	Concede permissão para descrever a configuração dinâmica de um cluster, equivalente à ACL DESCRIBE_CONFIGS CLUSTER do Apache Kafka.	`kafka-cluster:Connect`	cluster	Não

Ação	Descrição	Ações necessárias	Recursos necessários do	Aplicável a clusters com a tecnologia sem servidor

`kafka-cluster:AlterClusterDynamicConfiguration`	Concede permissão para alterar a configuração dinâmica de um cluster, equivalente à ACL ALTER_CONFIGS CLUSTER do Apache Kafka.	`kafka-cluster:Connect`
		
		`kafka-cluster:DescribeClusterDynamicConfiguration`
			cluster	Não

`kafka-cluster:WriteDataIdempotently`	Concede permissão para gravar dados em um cluster de modo idempotente, equivalente à ACL IDEMPOTENT_WRITE CLUSTER do Apache Kafka.	`kafka-cluster:Connect`
		
		`kafka-cluster:WriteData`
			cluster	Sim

Ação	Descrição	Ações necessárias	Recursos necessários do	Aplicável a clusters com a tecnologia sem servidor
`kafka-cluster:CreateTopic`	Concede permissão para criar tópicos em um cluster, equivalente à ACL CREATE CLUSTER/TOPIC do Apache Kafka.	`kafka-cluster:Connect`	tópico	Sim
`kafka-cluster:DescribeTopic`	Concede permissão para descrever os tópicos de um cluster, equivalente à ACL DESCRIBE TOPIC do Apache Kafka.	`kafka-cluster:Connect`	tópico	Sim

`kafka-cluster:AlterTopic`	Concede permissão para alterar os tópicos de um cluster, equivalente à ACL ALTER TOPIC do Apache Kafka.	`kafka-cluster:Connect`
		
		`kafka-cluster:DescribeTopic`
			tópico	Sim

Ação	Descrição	Ações necessárias	Recursos necessários do	Aplicável a clusters com a tecnologia sem servidor
`kafka-cluster:DeleteTopic`	Concede permissão para excluir tópicos de um cluster, equivalente à ACL DELETE TOPIC do Apache Kafka.	`kafka-cluster:Connect` `kafka-cluster:DescribeTopic`	tópico	Sim
`kafka-cluster:DescribeTopicDynamicConfiguration`	Concede permissão para descrever a configuração dinâmica dos tópicos de um cluster, equivalente à ACL DESCRIBE_CONFIGS TOPIC do Apache Kafka.	`kafka-cluster:Connect`	tópico	Sim

Ação	Descrição	Ações necessárias	Recursos necessários do	Aplicável a clusters com a tecnologia sem servidor
`kafka-cluster:AlterTopicDynamicConfiguration`	Concede permissão para alterar a configuração dinâmica dos tópicos de um cluster, equivalente à ACL ALTER_CONFIGS TOPIC do Apache Kafka.	`kafka-cluster:Connect` `kafka-cluster:DescribeTopicDynamicConfiguration`	tópico	Sim

`kafka-cluster:ReadData`	Concede permissão para ler dados dos tópicos de um cluster, equivalente à ACL READ TOPIC do Apache Kafka.	`kafka-cluster:Connect`
		`kafka-cluster:DescribeTopic`
		`kafka-cluster:AlterGroup`
			tópico	Sim

Ação	Descrição	Ações necessárias	Recursos necessários do	Aplicável a clusters com a tecnologia sem servidor
`kafka-cluster:WriteData`	Concede permissão para gravar dados em tópicos de um cluster, equivalente a WRITE TOPIC ACL do Apache Kafka.	`kafka-cluster:Connect` `kafka-cluster:DescribeTopic`	tópico	Sim
`kafka-cluster:DescribeGroup`	Concede permissão para descrever os grupos de um cluster, equivalente à ACL DESCRIBE GROUP do Apache Kafka.	`kafka-cluster:Connect`	group	Sim
`kafka-cluster:AlterGroup`	Concede permissão para entrar em grupos de um cluster, equivalente à ACL READ GROUP do Apache Kafka.	`kafka-cluster:Connect` `kafka-cluster:DescribeGroup`	group	Sim

Ação	Descrição	Ações necessárias	Recursos necessários do	Aplicável a clusters com a tecnologia sem servidor
`kafka-cluster:DeleteGroup`	Concede permissão para excluir grupos de um cluster, equivalente à ACL DELETE GROUP do Apache Kafka.	`kafka-cluster:Connect` `kafka-cluster:DescribeGroup`	group	Sim
`kafka-cluster:DescribeTransactionalId`	Concede permissão para descrever os IDs transacionais de um cluster, equivalente à ACL DESCRIBE TRANSACTIONAL_ID do Apache Kafka.	`kafka-cluster:Connect`	transactional-id	Sim

`kafka-cluster:AlterTransactionalId`	Concede permissão para alterar IDs transacionais de um cluster, equivalente à ACL WRITE TRANSACTIONAL_ID do Apache Kafka.	`kafka-cluster:Connect`
		`kafka-cluster:DescribeTransactionalId`
		`kafka-cluster:WriteData`
			transactional-id	Sim

Você pode usar o curinga asterisco (*) quantas vezes quiser em uma ação após o sinal de dois pontos. Veja os exemplos a seguir.

- `kafka-cluster:*Topic` corresponde a `kafka-cluster:CreateTopic`, `kafka-cluster:DescribeTopic`, `kafka-cluster:AlterTopic` e `kafka-cluster:DeleteTopic`. Isso não inclui `kafka-cluster:DescribeTopicDynamicConfiguration` ou `kafka-cluster:AlterTopicDynamicConfiguration`.
- `kafka-cluster:*` corresponde a todas as permissões.

Recursos

A tabela a seguir mostra os quatro tipos de recurso que você pode usar em uma política de autorização ao usar o controle de acesso do IAM para o Amazon MSK. Você pode obter o Amazon Resource Name (ARN) do cluster no AWS Management Console ou usando a DescribeClusterAPI ou o comando AWS CLI describe-cluster. Em seguida, você pode usar o ARN do cluster para estruturar ARNs de ID de tópico, grupo e transação. Para especificar um recurso em uma política de autorização, use o ARN desse recurso.

Recurso	Formato ARN
Cluster	arn:aws:kafka:*região*:*id-conta*:cluster/*nome-cluster* /*uuid-cluster*
Tópico	arn:aws:kafka:*região*:*id-conta*:topic/*nome-cluster* /*uuid-clus ter* /*nome-tópico*
Grupo	arn:aws:kafka:*região*:*id-conta*:group/*nome-cluster* /*uuid-clus ter* /*nome-grupo*
ID transacio nal	arn:aws:kafka:*região*:*id-conta*:transactional-id/*nome-cluster* /*uuid-clus ter* /*id-transacional*

Você pode usar o curinga asterisco (*) quantas vezes quiser em qualquer lugar na parte do ARN que vem depois de `:cluster/`, `:topic/`, `:group/` e `:transactional-id/`. Veja a seguir alguns exemplos de como usar o curinga asterisco (*) para se referir a vários recursos:

- `arn:aws:kafka:us-east-1:0123456789012:topic/MyTestCluster/*`: todos os tópicos em qualquer cluster nomeado MyTestCluster, independentemente do UUID do cluster.

- `arn:aws:kafka:us-east-1:0123456789012:topic/MyTestCluster/abcd1234-0123-abcd-5678-1234abcd-1/*_test`: todos os tópicos cujo nome termina com "_test" no cluster cujo nome é MyTestCluster e cujo UUID é abcd1234-0123-abcd-5678-1234abcd-1.
- `arn:aws:kafka:us-east-1:0123456789012:transactional-id/MyTestCluster/*/5555abcd-1111-abcd-1234-abcd1234-1`: todas as transações cuja ID transacional é 5555abcd-1111-abcd-1234-abcd1234-1, em todas as encarnações de um cluster nomeado em sua conta. MyTestCluster Isso significa que, se você criar um cluster chamado MyTestCluster, excluí-lo e criar outro cluster com o mesmo nome, poderá usar esse ARN de recurso para representar a mesma ID de transação nos dois clusters. No entanto, o cluster excluído não estará acessível.

Casos de uso comuns

A primeira coluna na tabela a seguir mostra alguns casos de uso comuns. Para autorizar um cliente a executar um determinado caso de uso, inclua as ações necessárias para esse caso de uso na política de autorização do cliente e defina `Effect` como `Allow`.

Para obter informações sobre todas as ações que fazem parte do controle de acesso do IAM para o Amazon MSK, consulte the section called "Semântica de ações e recursos".

> **Note**
>
> As ações são negadas por padrão. Você deve permitir explicitamente todas as ações que deseja autorizar o cliente a executar.

Caso de uso	Ações necessárias
Administrador	`kafka-cluster:*`
Criar um tópico	`kafka-cluster:Connect` `kafka-cluster:CreateTopic`

Produzir dados	`kafka-cluster:Connect`
	`kafka-cluster:DescribeTopic`
	`kafka-cluster:WriteData`

Caso de uso	Ações necessárias

Consumir dados	`kafka-cluster:Connect`
	`kafka-cluster:DescribeTopic`
	`kafka-cluster:DescribeGroup`
	`kafka-cluster:AlterGroup`
	`kafka-cluster:ReadData`

Produzir dados de modo idempotente	`kafka-cluster:Connect`
	`kafka-cluster:DescribeTopic`
	`kafka-cluster:WriteData`
	`kafka-cluster:WriteDataIdempotently`

Produzir dados de modo transacional	`kafka-cluster:Connect`
	`kafka-cluster:DescribeTopic`
	`kafka-cluster:WriteData`
	`kafka-cluster:DescribeTransactionalId`
	`kafka-cluster:AlterTransactionalId`

Descrever a configuração de um cluster	`kafka-cluster:Connect` `kafka-cluster:DescribeClusterDynamicConfiguration`

Caso de uso	Ações necessárias

Atualizar a configuração de um cluster	`kafka-cluster:Connect`
	
	`kafka-cluster:DescribeClusterDynamicConfiguration`
	
	`kafka-cluster:AlterClusterDynamicConfiguration`

Descrever a configuração de um tópico	`kafka-cluster:Connect`
	
	`kafka-cluster:DescribeTopicDynamicConfiguration`

Atualizar a configuração de um tópico	`kafka-cluster:Connect`
	
	`kafka-cluster:DescribeTopicDynamicConfiguration`
	
	`kafka-cluster:AlterTopicDynamicConfiguration`

Alterar um tópico	`kafka-cluster:Connect`
	
	`kafka-cluster:DescribeTopic`
	
	`kafka-cluster:AlterTopic`

Autenticação TLS mútua

Você pode habilitar a autenticação de clientes com TLS para conexões de seus aplicativos com seus corretores Amazon MSK. Para usar a autenticação do cliente, é necessário ter um CA privada da AWS. Eles CA privada da AWS podem estar no Conta da AWS mesmo cluster ou em uma conta diferente. Para obter informações sobre CA privada da AWS s, consulte Criando e gerenciando um CA privada da AWS.

> **Note**
>
> No momento, a autenticação por TLS não está disponível nas regiões Pequim e Ningxia.

O Amazon MSK não é compatível com Certificate revocation lists (CRLs – Listas de revogação de certificados). Para controlar o acesso aos tópicos do cluster ou bloquear certificados comprometidos, use ACLs e grupos de segurança do Apache Kafka. AWS Para obter mais informações sobre como usar ACLs do Apache Kafka, consulte the section called “ACLs do Apache Kafka”.

Este tópico contém as seguintes seções:

Como criar um cluster que ofereça suporte à autenticação de cliente

Este procedimento mostra como habilitar a autenticação do cliente usando um CA privada da AWS.

> **Note**
>
> É altamente recomendável usar o independente CA privada da AWS para cada cluster MSK ao usar o TLS mútuo para controlar o acesso. Isso garantirá que os certificados TLS assinados por PCAs sejam autenticados somente com um cluster do MSK.

1. Crie um arquivo denominado `clientauthinfo.json` com o seguinte conteúdo: Substitua *`Private-CA-ARN`* pelo ARN do PCA.

   ```
   {
      "Tls": {
          "CertificateAuthorityArnList": ["Private-CA-ARN"]
       }
   }
   ```

2. Crie um arquivo chamado `brokernodegroupinfo.json`, conforme descrito em the section called “Criando um cluster usando o AWS CLI”.

3. A autenticação de cliente exige que você também ative a criptografia em trânsito entre clientes e agentes. Crie um arquivo denominado `encryptioninfo.json` com o seguinte conteúdo: Substitua *KMS-Key-ARN* pelo ARN da chave do KMS. É possível definir `ClientBroker` como `TLS` ou `TLS_PLAINTEXT`.

```
{
   "EncryptionAtRest": {
       "DataVolumeKMSKeyId": "KMS-Key-ARN"
    },
   "EncryptionInTransit": {
        "InCluster": true,
        "ClientBroker": "TLS"
    }
}
```

   Para obter mais informações sobre criptografia, consulte the section called “Criptografia”.

4. Em uma máquina em que você tenha o AWS CLI instalado, execute o comando a seguir para criar um cluster com a autenticação e a criptografia em trânsito habilitadas. Salve o ARN do cluster fornecido na resposta.

```
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
```

Como configurar um cliente para usar a autenticação

1. Crie uma instância do Amazon EC2 para ser usada como uma máquina cliente. Para simplificar, crie essa instância na mesma VPC usada para o cluster. Consulte the section called “Etapa 3: criar uma máquina cliente” para obter um exemplo de como criar uma máquina de cliente.
2. Criar um tópico. Para obter um exemplo, consulte as instruções em the section called “Etapa 4: criar um tópico”.
3. Em uma máquina em que você tem o AWS CLI instalado, execute o comando a seguir para obter os corretores de bootstrap do cluster. Substitua o *Cluster-ARN* pelo ARN do seu cluster.

```
aws kafka get-bootstrap-brokers --cluster-arn Cluster-ARN
```

Salve a string associada ao `BootstrapBrokerStringTls` na resposta.

4. Na máquina de cliente, execute o comando a seguir para usar o armazenamento de confiança da JVM para criar o armazenamento de confiança do cliente. Se o caminho da JVM for diferente, ajuste o comando de acordo.

```
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. Na máquina de cliente, execute o comando a seguir para criar uma chave privada para o cliente. Substitua *Distinguished-Name*, *Example-Alias*, *Your-Store-Pass* e *Your-Key-Pass* pelas strings de sua escolha.

```
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. Na máquina de cliente, execute o comando a seguir para criar uma solicitação de certificado com a chave privada criada na etapa anterior.

```
keytool -keystore kafka.client.keystore.jks -certreq -file client-cert-sign-request
 -alias Example-Alias -storepass Your-Store-Pass -keypass Your-Key-Pass
```

7. Abra o arquivo `client-cert-sign-request` e verifique se ele começa com `-----BEGIN CERTIFICATE REQUEST-----` e termina com `-----END CERTIFICATE REQUEST-----`. Se ele começar com `-----BEGIN NEW CERTIFICATE REQUEST-----`, exclua a palavra `NEW` (e o espaço único que vem após) do começo e do final do arquivo.

8. Em uma máquina em que você tenha o AWS CLI instalado, execute o comando a seguir para assinar sua solicitação de certificado. Substitua *Private-CA-ARN* pelo ARN do PCA. Será possível alterar o valor de validade se quiser. Aqui usamos 300 como exemplo.

```
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"
```

Salve o ARN do certificado fornecido na resposta.

> **Note**
>
> Para recuperar seu certificado de cliente, use o comando `acm-pca get-certificate` e especifique o ARN do certificado. Para obter mais informações, consulte get-certificate na Referência de comandos da AWS CLI .

9. Execute o comando a seguir para obter o certificado CA privada da AWS assinado para você. Substitua *Certificate-ARN* pelo ARN obtido na resposta ao comando anterior.

```
aws acm-pca get-certificate --certificate-authority-arn Private-CA-ARN --
certificate-arn Certificate-ARN
```

10. Do resultado JSON obtido com a execução do comando anterior, copie as strings associadas a `Certificate` e `CertificateChain`. Cole essas duas sequências em um novo arquivo chamado signed-certificate-from-acm. Cole a string associada a `Certificate` primeiro, seguida pela string associada a `CertificateChain`. Substitua os caracteres `\n` por novas linhas. Veja a seguir a estrutura do arquivo depois que você colar o certificado e a cadeia de certificados nele.

```
-----BEGIN CERTIFICATE-----
...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
...
-----END CERTIFICATE-----
```

11. Execute o comando a seguir na máquina cliente para adicionar esse certificado ao repositório de chaves para poder apresentá-lo ao falar com os agentes do 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. Crie um arquivo denominado `client.properties` com o seguinte conteúdo: Ajuste os locais do armazenamento de confiança e do repositório de chaves usando os caminhos onde salvou `kafka.client.truststore.jks`. Substitua sua versão do cliente Kafka pelos espaços reservados *{SUA VERSÃO DO KAFKA}*.

```
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
```

Como produzir e consumir mensagens usando a autenticação

1. Execute o comando a seguir para criar um tópico. O arquivo chamado `client.properties` é o que você criou no procedimento anterior.

   ```
   <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. Execute o comando a seguir para iniciar um produtor de console. O arquivo chamado `client.properties` é o que você criou no procedimento anterior.

   ```
   <path-to-your-kafka-installation>/bin/kafka-console-producer.sh --bootstrap-
   server BootstrapBroker-String --topic ExampleTopic --producer.config
    client.properties
   ```

3. Em uma nova janela de comando na máquina de cliente, execute o comando a seguir para iniciar um consumidor de console.

   ```
   <path-to-your-kafka-installation>/bin/kafka-console-consumer.sh --bootstrap-
   server BootstrapBroker-String --topic ExampleTopic --consumer.config
    client.properties
   ```

4. Digite mensagens na janela do produtor e observe-as aparecerem na janela do consumidor.

Autenticação de credenciais de login com Secrets Manager AWS

Você pode controlar o acesso aos seus clusters do Amazon MSK usando credenciais de login que são armazenadas e protegidas usando o Secrets Manager. AWS Armazenar as credenciais de usuário no Secrets Manager reduz a sobrecarga da autenticação do cluster, como auditoria,

atualização e rodízio de credenciais. O Secrets Manager também permite que você compartilhe credenciais de usuário entre clusters.

Este tópico contém as seguintes seções:

Como funciona

A autenticação de credenciais de acesso para o Amazon MSK usa a autenticação SASL/SCRAM (Simple Authentication and Security Layer/Salted Challenge Response Authentication Mechanism). Para configurar a autenticação de credenciais de acesso para um cluster, você cria um recurso secreto no AWS Secrets Manager e associa as credenciais de acesso a esse segredo.

O SASL/SCRAM está definido no RFC 5802. O SCRAM usa algoritmos de hash protegidos e não transmite credenciais em texto simples entre o cliente e o servidor.

> **Note**
>
> Quando você configura a autenticação SASL/SCRAM para seu cluster, o Amazon MSK ativa a criptografia TLS para todo o tráfego entre clientes e agentes.

Como configurar a autenticação SASL/SCRAM para um cluster do Amazon MSK

Para configurar um segredo no AWS Secrets Manager, siga o tutorial Criando e recuperando um segredo no Guia do usuário do AWS Secrets Manager.

Observe os seguintes requisitos ao criar um segredo para um cluster do Amazon MSK:

- Escolha Outros tipos de segredos (p. ex., chave de API) para o tipo de segredo.
- O nome do segredo deve começar com o prefixo AmazonMSK_.
- Você deve usar uma AWS KMS chave personalizada existente ou criar uma nova AWS KMS chave personalizada para seu segredo. O Secrets Manager usa a AWS KMS chave padrão para um segredo por padrão.

> **Important**
>
> Um segredo criado com a AWS KMS chave padrão não pode ser usado com um cluster Amazon MSK.

- Seus dados de credencial de acesso devem estar no formato a seguir para que seja possível inserir pares de valor/chave usando a opção Texto simples.

```
{
  "username": "alice",
  "password": "alice-secret"
}
```

- Registre o valor do ARN (nome do recurso da Amazon) do seu segredo.
-

> **Important**
>
> Você não pode associar um segredo do Secrets Manager a um cluster que exceda os limites descritos em the section called “ Dimensione seu cluster adequadamente: número de partições por agente”.

- Se você usar o AWS CLI para criar o segredo, especifique um ID de chave ou ARN para o `kms-key-id` parâmetro. Não especifique um alias.
- Para associar o segredo ao seu cluster, use o console Amazon MSK ou a BatchAssociateScramSecretoperação.

> **Important**
>
> Quando você associa um segredo a um cluster, o Amazon MSK anexa uma política de recursos ao segredo, permitindo que seu cluster acesse e leia os valores secretos que você definiu. Você não deve modificar essa política de recursos. Isso pode impedir que seu cluster acesse seu segredo.

O exemplo de entrada JSON a seguir para a operação `BatchAssociateScramSecret` associa um segredo a um 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"
  ]
}
```

Como estabelecer conexão com o seu cluster usando credenciais de acesso

Após criar um segredo e associá-lo ao cluster, você poderá conectar o cliente ao cluster. As etapas de exemplo a seguir demonstram como conectar um cliente a um cluster que usa autenticação SASL/SCRAM e como produzir e consumir com base em um tópico de exemplo.

1. *Execute o comando a seguir em uma máquina que tenha a AWS CLI instalada, substituindo clusterARN pelo ARN do seu cluster.*

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

2. Para criar um tópico de exemplo, execute o comando a seguir, substituindo *BootstrapServerString* por um dos endpoints do broker que você obteve na etapa anterior.

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

3. Em sua máquina cliente, crie um arquivo de configuração JAAS contendo as credenciais de usuário armazenadas em seu segredo. Por exemplo, para o usuário alice, crie um arquivo chamado `users_jaas.conf` com o conteúdo a seguir.

   ```
   KafkaClient {
      org.apache.kafka.common.security.scram.ScramLoginModule required
      username="alice"
      password="alice-secret";
   };
   ```

4. Use o comando a seguir para exportar seu arquivo de configuração JAAS como um parâmetro de ambiente `KAFKA_OPTS`.

   ```
   export KAFKA_OPTS=-Djava.security.auth.login.config=<path-to-jaas-file>/
   users_jaas.conf
   ```

5. Crie um arquivo chamado `kafka.client.truststore.jks` em um diretório `./tmp`.
6. Use o seguinte comando para copiar o arquivo de armazenamento de chaves do JDK da sua pasta `cacerts` da JVM para o arquivo `kafka.client.truststore.jks` que você criou na etapa anterior. Substitua *`JDKFolder`* pelo nome da pasta JDK na sua instância. Por exemplo, sua pasta do JDK pode ter o nome `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. No diretório `bin` da instalação do Apache Kafka, crie um arquivo de propriedades do cliente chamado `client_sasl.properties` com o conteúdo a seguir. Esse arquivo define o mecanismo e o protocolo SASL.

```
security.protocol=SASL_SSL
sasl.mechanism=SCRAM-SHA-512
ssl.truststore.location=<path-to-keystore-file>/kafka.client.truststore.jks
```

8. Recupere sua string de agentes de bootstrap com o comando a seguir. *`ClusterArn`*Substitua pelo Amazon Resource Name (ARN) do seu cluster:

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

   No JSON resultante do comando, salve o valor associado à string chamada `BootstrapBrokerStringSaslScram`.
9. Para produzir o tópico de exemplo que você criou, execute o seguinte comando em sua máquina cliente. Substitua o *`BootstrapBrokerStringSaslScram`* pelo valor que você recuperou na etapa anterior.

```
<path-to-your-kafka-installation>/bin/kafka-console-producer.sh --broker-
list BootstrapBrokerStringSaslScram --topic ExampleTopicName --producer.config
 client_sasl.properties
```

10. Para consumir do tópico que você criou, execute o comando a seguir em sua máquina cliente. Substitua o *`BootstrapBrokerStringSaslScram`* pelo valor que você obteve anteriormente.

```
<path-to-your-kafka-installation>/bin/kafka-console-consumer.sh --bootstrap-
server BootstrapBrokerStringSaslScram --topic ExampleTopicName --from-beginning --
consumer.config client_sasl.properties
```

Trabalhar com usuários

Criação de usuários: você cria usuários como pares de valor/chave em seu segredo. Ao usar a opção Texto simples no console do Secrets Manager, você deve especificar os dados da credencial de login no formato a seguir.

```
{
  "username": "alice",
  "password": "alice-secret"
}
```

Revogando o acesso do usuário: para revogar as credenciais de um usuário para acessar um cluster, recomendamos que primeiro você remova ou force uma ACL no cluster e depois desassocie o segredo. Isso se dá pelo seguinte:

- A remoção de um usuário não fecha as conexões existentes.
- A propagação de alterações em seu segredo levam até 10 minutos.

Para obter informações sobre como usar uma ACL com o Amazon MSK, consulte ACLs do Apache Kafka.

Para clusters usando o ZooKeeper modo, recomendamos que você restrinja o acesso aos seus ZooKeeper nós para impedir que os usuários modifiquem as ACLs. Para ter mais informações, consulte Controlando o acesso ao Apache ZooKeeper.

Limitações

Observe as seguintes limitações ao usar segredos SCRAM:

- O Amazon MSK só é compatível com a autenticação SCRAM-SHA-512.
- Um cluster do Amazon MSK pode ter até 1.000 usuários.
- Você deve usar um AWS KMS key com seu segredo. Você não pode usar um segredo que use a chave de criptografia padrão do Secrets Manager com o Amazon MSK. Para obter informações sobre a criação de uma chave do KMS, consulte Criação de chaves do KMS de criptografia simétrica.
- Não é possível usar uma chave assimétrica do KMS com o Secrets Manager.
- Você pode associar até 10 segredos a um cluster por vez usando a BatchAssociateScramSecretoperação.

- O nome dos segredos associados a um cluster do Amazon MSK deve ter o prefixo AmazonMSK_.
- Os segredos associados a um cluster do Amazon MSK devem estar na mesma conta e AWS região da Amazon Web Services do cluster.

ACLs do Apache Kafka

O Apache Kafka tem um autorizador conectável e vem com uma implementação autorizadora. out-of-box O Amazon MSK habilita esse autorizador no arquivo `server.properties` dos agentes.

As ACLs do Apache Kafka têm o formato “Principal P é [Permitida/Negada] Operação O do Host H em qualquer recurso R correspondente a RP”. ResourcePattern Se o RP não corresponder a um recurso R específico, R não terá ACLs associadas e, portanto, ninguém além de superusuários terá permissão para acessar R. Para alterar esse comportamento do Apache Kafka, defina a propriedade `allow.everyone.if.no.acl.found` como true. O Amazon MSK a define como true por padrão. Isso significa que, com clusters do Amazon MSK, se você não definir explicitamente as ACLs em um recurso, todos os principais poderão acessá-lo. Se você habilitar ACLs em um recurso, somente os principais autorizados poderão acessá-lo. Se você quiser restringir o acesso a um tópico e autorizar um cliente usando a autenticação mútua TLS, adicione ACLs usando a CLI de autorização do Apache Kafka. Para obter mais informações sobre adicionar, remover e listar ACLs, consulte Kafka Authorization Command Line Interface.

Além do cliente, também é necessário conceder a todos os agentes acesso aos seus tópicos para que os agentes possam replicar mensagens da partição primária. Se os agentes não tiverem acesso a um tópico, ocorrerá uma falha na replicação dele.

Como adicionar ou remover o acesso de leitura e gravação a um tópico

1. Adicione os agentes à tabela de ACL para permitir que eles leiam todos os tópicos que possuem ACLs. Para conceder acesso de leitura a um tópico para os seus agentes, execute o comando a seguir em uma máquina cliente capaz de se comunicar com o cluster do MSK.

   Substitua *`Distinguished-Name`* pelo DNS de qualquer um dos agentes de bootstrap do cluster e substitua a string antes do primeiro ponto desse nome distinto por um asterisco (`*`). Por exemplo, se um dos agentes de bootstrap do cluster tiver o DNS `b-6.mytestcluster.67281x.c4.kafka.us-east-1.amazonaws.com`, substitua *`Distinguished-Name`* no comando a seguir por `*.mytestcluster.67281x.c4.kafka.us-east-1.amazonaws.com`. Para ver

informações sobre como obter os agentes de bootstrap, consulte the section called “Como obter os agentes de 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. Para conceder acesso de leitura a um tópico, execute o comando a seguir na máquina de cliente. Se usar autenticação TLS mútua, use o mesmo *Distinguished-Name* usado ao criar a chave privada.

   ```
   <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
   ```

   Para remover o acesso de leitura, é possível executar o mesmo comando, substituindo `--add` por `--remove`.

3. Para conceder acesso de gravação a um tópico, execute o comando a seguir na máquina de cliente. Se usar autenticação TLS mútua, use o mesmo *Distinguished-Name* usado ao criar a chave privada.

   ```
   <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
   ```

   Para remover o acesso de gravação, é possível executar o mesmo comando, substituindo `--add` por `--remove`.

Alterar o grupo de segurança do cluster no Amazon MSK

Esta página explica como alterar o grupo de segurança de um cluster existente do MSK. Talvez seja necessário alterar o grupo de segurança de um cluster para fornecer acesso a um determinado conjunto de usuários ou limitar o acesso ao cluster. Para obter mais informações sobre grupos de segurança, consulte Grupos de segurança para sua VPC no Guia do usuário da Amazon VPC.

1. Use a ListNodesAPI ou o comando list-nodes no AWS CLI para obter uma lista dos corretores em seu cluster. Os resultados dessa operação incluem os IDs das interfaces de rede elástica (ENIs) associadas aos agentes.

2. Faça login no AWS Management Console e abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
3. Usando a lista suspensa no canto superior direito da tela, selecione a região na qual o cluster está implantado.
4. No painel esquerdo, em Rede e Segurança, escolha Interfaces de rede.
5. Selecione a primeira ENI que você obteve na primeira etapa. Escolha o menu Ações na parte superior da tela e escolha Alterar grupos de segurança. Atribua o novo grupo de segurança a essa ENI. Repita essa etapa para cada uma das ENIs que você obteve na primeira etapa.

> **Note**
>
> As alterações que você fizer no grupo de segurança de um cluster usando o console do Amazon EC2 não serão refletidas no console do MSK em Configurações de rede.

6. Configure as regras do novo grupo de segurança para garantir que seus clientes tenham acesso aos agentes. Para obter informações sobre como configurar regras de grupo de segurança, consulte Adicionar, remover e atualizar regras no guia do usuário da Amazon VPC.

> **Important**
>
> Se você alterar o grupo de segurança associado aos agentes de um cluster e depois adicionar novos agentes a esse cluster, o Amazon MSK associará os novos agentes ao grupo de segurança original que estava associado ao cluster quando o cluster foi criado. No entanto, para que um cluster funcione corretamente, todos os seus agentes devem estar associados ao mesmo grupo de segurança. Portanto, se você adicionar novos agentes após alterar o grupo de segurança, deverá seguir novamente o procedimento anterior e atualizar as ENIs dos novos agentes.

Controlando o acesso ao Apache ZooKeeper

Por motivos de segurança, você pode limitar o acesso aos ZooKeeper nós do Apache que fazem parte do seu cluster Amazon MSK. Para limitar o acesso aos nós, é possível atribuir um grupo de segurança separado para eles. Depois, é possível decidir quem tem acesso a esse grupo de segurança.

> **Important**
>
> Esta seção não se aplica a clusters executados no modo Kraft. Consulte the section called "Modo Kraft ".

Este tópico contém as seguintes seções:

Para colocar seus ZooKeeper nós do Apache em um grupo de segurança separado

1. Obtenha a string de ZooKeeper conexão do Apache para seu cluster. Para saber como, consulte the section called "ZooKeeper modo". A string de conexão contém os nomes DNS dos seus nós do Apache ZooKeeper .
2. Use uma ferramenta como `host` ou `ping` para converter os nomes de DNS obtidos na etapa anterior para endereços IP. Salve esses endereços IP porque você precisará deles posteriormente neste procedimento.
3. Faça login no AWS Management Console e abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
4. No painel de navegação, em NETWORK & SECURITY (REDE E SEGURANÇA), selecione Network Interfaces (Interfaces de rede).
5. No campo de pesquisa acima da tabela de interfaces de rede, digite o nome do cluster e digite return. Isso limita o número de interfaces de rede que aparecem na tabela às interfaces associadas ao cluster.
6. Marque a caixa de seleção no início da linha que corresponde à primeira interface de rede na lista.
7. No painel de detalhes na parte inferior da página, procure o Primary private IPv4 IP (IP IPv4 privado primário). Se esse endereço IP corresponder a um dos endereços IP que você obteve na primeira etapa desse procedimento, isso significa que essa interface de rede está atribuída a um ZooKeeper nó Apache que faz parte do seu cluster. Caso contrário, desmarque a caixa de seleção ao lado dessa interface de rede e selecione a próxima interface de rede na lista. A ordem em que você seleciona as interfaces de rede não importa. Nas próximas etapas, você

executará as mesmas operações em todas as interfaces de rede atribuídas aos ZooKeeper nós do Apache, uma por uma.

8. Ao selecionar uma interface de rede que corresponde a um ZooKeeper nó do Apache, escolha o menu Ações na parte superior da página e escolha Alterar grupos de segurança. Atribua um novo grupo de segurança a essa interface de rede. Para obter informações sobre como criar grupos de segurança, consulte Criar um grupo de segurança na documentação da Amazon VPC.
9. Repita a etapa anterior para atribuir o mesmo novo grupo de segurança a todas as interfaces de rede associadas aos ZooKeeper nós Apache do seu cluster.
10. Agora é possível escolher quem tem acesso a esse novo grupo de segurança. Para obter informações sobre como configurar regras de grupo de segurança, consulte Adicionar, remover e atualizar regras na documentação da Amazon VPC.

Usando a segurança TLS com o Apache ZooKeeper

Você pode usar a segurança TLS para criptografia em trânsito entre seus clientes e seus nós Apache ZooKeeper . Para implementar a segurança TLS com seus ZooKeeper nós Apache, faça o seguinte:

- Os clusters devem usar o Apache Kafka versão 2.5.1 ou posterior para usar a segurança TLS com o Apache. ZooKeeper
- Habilite a segurança TLS ao criar ou configurar seu cluster. Clusters criados com o Apache Kafka versão 2.5.1 ou posterior com TLS ativado usam automaticamente a segurança TLS com endpoints Apache. ZooKeeper Para obter mais informações sobre a configuração da segurança TLS, consulte Como começo a usar a criptografia?.
- Recupere os ZooKeeper endpoints TLS Apache usando a operação. DescribeCluster
- Crie um arquivo de ZooKeeper configuração do Apache para uso com as `kafka-acls.sh`ferramentas `kafka-configs.sh` e ou com o ZooKeeper shell. Com cada ferramenta, você usa o `--zk-tls-config-file` parâmetro para especificar sua ZooKeeper configuração do Apache.

  O exemplo a seguir mostra um arquivo de ZooKeeper configuração típico do Apache:

```
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
```

- Para outros comandos (como`kafka-topics`), você deve usar a variável de `KAFKA_OPTS` ambiente para configurar ZooKeeper os parâmetros do Apache. O exemplo a seguir mostra como configurar a variável de `KAFKA_OPTS` ambiente para passar ZooKeeper parâmetros do Apache para outros comandos:

  ```
  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"
  ```

  Após configurar a variável de ambiente `KAFKA_OPTS`, você pode usar os comandos da CLI normalmente. O exemplo a seguir cria um tópico do Apache Kafka usando a ZooKeeper configuração do Apache a partir da variável de ambiente: `KAFKA_OPTS`

  ```
  <path-to-your-kafka-installation>/bin/kafka-topics.sh --create --
  zookeeper ZooKeeperTLSConnectString --replication-factor 3 --partitions 1 --topic
   AWSKafkaTutorialTopic
  ```

> **Note**
>
> Os nomes dos parâmetros que você usa no seu arquivo de ZooKeeper configuração do Apache e aqueles que você usa na sua variável de `KAFKA_OPTS` ambiente não são consistentes. Preste atenção nos nomes que você usa com quais parâmetros no arquivo de configuração e na variável de ambiente `KAFKA_OPTS`.

Para obter mais informações sobre como acessar seus ZooKeeper nós Apache com TLS, consulte KIP-515: Habilitar o cliente ZK para usar a nova autenticação compatível com TLS.

Registro em log

Você pode entregar registros do agente Apache Kafka para um ou mais dos seguintes tipos de destino: Amazon CloudWatch Logs, Amazon S3, Amazon Data Firehose. Você também pode registrar chamadas de API do Amazon MSK com AWS CloudTrail.

Logs do agente

Os logs de agente permitem solucionar problemas de aplicações do Apache Kafka e analisar as comunicações delas com o seu cluster do MSK. Você pode configurar seu cluster MSK novo ou existente para fornecer registros de agente em nível de informação a um ou mais dos seguintes tipos de recursos de destino: um grupo de CloudWatch registros, um bucket do S3, um stream de entrega do Firehose. Por meio do Firehose, você pode então entregar os dados de registro do seu stream de entrega para OpenSearch o Service. Você deve criar um recurso de destino antes de configurar seu cluster para entregar registros do agente a esse recurso. O Amazon MSK não cria esses recursos de destino para você se eles ainda não existirem. Para obter informações sobre esses três tipos de recursos de destino e como criá-los, consulte a seguinte documentação:

- CloudWatch Registros da Amazon
- Amazon S3
- Amazon Data Firehose

Permissões obrigatórias

Para configurar um destino para os logs de agente do Amazon MSK, a identidade do IAM que você usa para as ações do Amazon MSK deve ter as permissões descritas na política AWS política gerenciada: AmazonMSK FullAccess.

Para transmitir logs de agente para um bucket do S3, também é necessário ter a permissão `s3:PutBucketPolicy`. Para obter informações sobre as políticas de bucket do S3, consulte Como adiciono uma política de bucket do S3? no Guia do usuário do Amazon S3. Para obter informações sobre as políticas do IAM em geral, consulte Gerenciamento de acesso no Guia do usuário do IAM.

Política de chave obrigatória do KMS para uso com buckets de SSE-KMS

Se você habilitou a criptografia do lado do servidor para seu bucket do S3 usando chaves AWS KMS gerenciadas (SSE-KMS) com uma chave gerenciada pelo cliente, adicione o seguinte à política de chaves da sua chave KMS para que o Amazon MSK possa gravar arquivos de agente no 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": "*"
}
```

Configurando registros do broker usando o AWS Management Console

Se estiver criando um cluster, procure o cabeçalho Broker log delivery (Entrega de log de agente) na seção Monitoring (Monitoramento). É possível especificar os destinos aos quais deseja que o Amazon MSK entregue os logs de agente.

Para um cluster existente, escolha o cluster na lista de clusters e selecione a guia Propriedades. Role para baixo até a seção Entrega de logs e escolha o botão Editar. É possível especificar os destinos aos quais deseja que o Amazon MSK entregue os logs de agente.

Configurando registros do broker usando o AWS CLI

Quando você usar o comando `create-cluster` ou `update-monitoring`, poderá especificar o parâmetro `logging-info` opcionalmente e passar uma estrutura JSON para ele como o exemplo a seguir. Nesse JSON, todos os três tipos de destino são opcionais.

```
{
  "BrokerLogs": {
    "S3": {
      "Bucket": "ExampleBucketName",
      "Prefix": "ExamplePrefix",
      "Enabled": true
    },
    "Firehose": {
      "DeliveryStream": "ExampleDeliveryStreamName",
      "Enabled": true
    },
    "CloudWatchLogs": {
      "Enabled": true,
      "LogGroup": "ExampleLogGroupName"
```

```
    }
  }
}
```

Configurar logs de agente usando a API

Você pode especificar a `loggingInfo` estrutura opcional no JSON que você passa para as UpdateMonitoringoperações CreateClusterou.

> **Note**
>
> Por padrão, quando o registro em log do agente estiver habilitado, o Amazon MSK registrará os logs no nível de `INFO` nos destinos especificados. No entanto, os usuários do Apache Kafka 2.4.X e versões posteriores podem definir dinamicamente o nível de log do agente para qualquer um dos níveis de log log4j. Para obter informações sobre como definir dinamicamente o nível de log do agente, consulte KIP-412: estender a API Admin para oferecer suporte aos níveis dinâmicos de log do aplicativo. Se você definir dinamicamente o nível de log como `DEBUG` ou`TRACE`, recomendamos usar o Amazon S3 ou o Firehose como destino do log. Se você usar CloudWatch Logs como um destino de log e ativar `DEBUG` ou `TRACE` nivelar dinamicamente o registro, o Amazon MSK poderá fornecer continuamente uma amostra de registros. Isso pode afetar significativamente o desempenho do agente e só deve ser usado quando o nível de log `INFO` não for suficientemente detalhado para determinar a causa raiz de um problema.

Registro de chamadas de API do AWS CloudTrail com

> **Note**
>
> AWS CloudTrail os registros estão disponíveis para o Amazon MSK somente quando você usaControle de acesso do IAM.

O Amazon MSK é integrado com AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, função ou AWS serviço no Amazon MSK. CloudTrail captura chamadas de API como eventos. As chamadas capturadas incluem as chamadas do console do Amazon MSK e as chamadas de código para as operações da API do Amazon MSK. Ele também captura ações do Apache Kafka, como criar e alterar tópicos e grupos.

Se você criar uma trilha, poderá habilitar a entrega contínua de CloudTrail eventos para um bucket do Amazon S3, incluindo eventos para o Amazon MSK. Se você não configurar uma trilha, ainda poderá ver os eventos mais recentes no CloudTrail console no Histórico de eventos. Usando as informações coletadas por CloudTrail, você pode determinar a solicitação que foi feita para a Amazon MSK ou a ação do Apache Kafka, o endereço IP do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais.

Para saber mais CloudTrail, inclusive como configurá-lo e ativá-lo, consulte o Guia AWS CloudTrail do usuário.

Informações do Amazon MSK em CloudTrail

CloudTrail é ativado na sua conta da Amazon Web Services quando você cria a conta. Quando a atividade de evento suportada ocorre em um cluster MSK, essa atividade é registrada em um CloudTrail evento junto com outros eventos de AWS serviço no histórico de eventos. Você pode visualizar, pesquisar e baixar os eventos recentes em sua conta da Amazon Web Services. Para obter mais informações, consulte Visualização de eventos com histórico de CloudTrail eventos.

Para obter um registro contínuo dos eventos na sua conta da Amazon Web Services, incluindo os eventos do Amazon MSK, crie uma trilha. Uma trilha permite CloudTrail entregar arquivos de log para um bucket do Amazon S3. Por padrão, ao criar uma trilha no console, a mesma é aplicada a todas as Regiões. A trilha registra logs de eventos de todas as Regiões na AWS divisória e entrega os arquivos do log para o bucket Amazon S3 especificado. Além disso, você pode configurar outros serviços da Amazon para analisar e agir com base nos dados de eventos coletados nos CloudTrail registros. Para obter mais informações, consulte as informações a seguir.

- Visão Geral para Criar uma Trilha
- CloudTrail Serviços e integrações compatíveis
- Configurando notificações do Amazon SNS para CloudTrail
- Recebendo arquivos de CloudTrail log de várias regiões e recebendo arquivos de CloudTrail log de várias contas

O Amazon MSK registra todas as operações do Amazon MSK como eventos em arquivos de CloudTrail log. Além disso, ele registra as seguintes ações do Apache Kafka.

- cluster de kafka: DescribeClusterDynamicConfiguration
- cluster de kafka: AlterClusterDynamicConfiguration
- cluster de kafka: CreateTopic

- cluster de kafka: DescribeTopicDynamicConfiguration
- cluster de kafka: AlterTopic
- cluster de kafka: AlterTopicDynamicConfiguration
- cluster de kafka: DeleteTopic

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar:

- Se a solicitação foi feita com credenciais de usuário root ou AWS Identity and Access Management (IAM).
- Se a solicitação foi feita com credenciais de segurança temporárias de um perfil ou de um usuário federado.
- Se a solicitação foi feita por outro AWS serviço.

Para obter mais informações, consulte o elemento CloudTrail UserIdentity.

Exemplo: entradas de arquivo de log do Amazon MSK

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log para um bucket do Amazon S3 que você especificar. CloudTrail os arquivos de log contêm uma ou mais entradas de log. Um evento representa uma única solicitação de qualquer fonte, e inclui informações sobre a ação solicitada, data e hora da ação, parâmetros de solicitação e assim por diante. CloudTrail os arquivos de log não são um rastreamento de pilha ordenado das chamadas públicas da API e das ações do Apache Kafka, portanto, eles não aparecem em nenhuma ordem específica.

O exemplo a seguir mostra entradas de CloudTrail registro que demonstram as ações do `DeleteCluster` Amazon MSK `DescribeCluster` e do 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"
    }
  ]
}
```

O exemplo a seguir mostra uma entrada de CloudTrail registro que demonstra a `kafka-cluster:CreateTopic` ação.

```
{
  "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"
}
```

Validação de conformidade do Amazon Managed Streaming for Apache Kafka

Auditores terceirizados avaliam a segurança e a conformidade do Amazon Managed Streaming for Apache Kafka como parte de programas de conformidade da AWS . Eles incluem PCI e HIPAA BAA.

Para obter uma lista de AWS serviços no escopo de programas de conformidade específicos, consulte Amazon Services in Scope by Compliance Program . Para obter informações gerais, consulte Programas de AWS conformidade Programas AWS de .

Você pode baixar relatórios de auditoria de terceiros usando AWS Artifact. Para obter mais informações, consulte Baixar relatórios em AWS Artifact .

Sua responsabilidade de conformidade ao usar o Amazon MSK é determinada pela confidencialidade dos seus dados, pelos objetivos de conformidade da sua empresa e pelas leis e regulamentações aplicáveis. AWS fornece os seguintes recursos para ajudar na conformidade:

- Guias de início rápido de segurança e compatibilidade: estes guias de implantação abordam as considerações de arquitetura e fornecem etapas para implantação de ambientes de linha de base focados em compatibilidade e segurança na AWS.
- Documento técnico sobre arquitetura para segurança e conformidade com a HIPAA — Este whitepaper descreve como as empresas podem usar para criar aplicativos compatíveis com a HIPAA. AWS
- AWS Recursos de https://aws.amazon.com/compliance/resources/ de conformidade — Essa coleção de pastas de trabalho e guias pode ser aplicada ao seu setor e local.
- Avaliação de recursos com regras no Guia do AWS Config desenvolvedor — O AWS Config serviço avalia o quão bem suas configurações de recursos estão em conformidade com as práticas internas, as diretrizes e os regulamentos do setor.
- AWS Security Hub— Esse AWS serviço fornece uma visão abrangente do seu estado de segurança interno, AWS que ajuda você a verificar sua conformidade com os padrões e as melhores práticas do setor de segurança.

Resiliência no Amazon Managed Streaming for Apache Kafka

A infraestrutura AWS global é construída em torno de AWS regiões e zonas de disponibilidade. AWS As regiões fornecem várias zonas de disponibilidade fisicamente separadas e isoladas, conectadas a redes de baixa latência, alta taxa de transferência e alta redundância. Com as zonas de disponibilidade, é possível projetar e operar aplicativos e bancos de dados que automaticamente executam o failover entre as zonas sem interrupção. As zonas de disponibilidade são mais altamente disponíveis, tolerantes a falhas e escaláveis que uma ou várias infraestruturas de data center tradicionais.

Para obter mais informações sobre AWS regiões e zonas de disponibilidade, consulte Infraestrutura AWS global.

Segurança de infraestrutura no Amazon Managed Streaming for Apache Kafka

Como um serviço gerenciado, o Amazon Managed Streaming for Apache Kafka é protegido AWS pelos procedimentos globais de segurança de rede descritos no whitepaper Amazon Web Services: Visão geral dos processos de segurança.

Você usa chamadas de API AWS publicadas para acessar o Amazon MSK pela rede. Os clientes devem oferecer compatibilidade com Transport Layer Security (TLS) 1.0 ou posterior. Recomendamos TLS 1.2 ou posterior. Os clientes também devem ter compatibilidade com conjuntos de criptografia com perfect forward secrecy (PFS) como Ephemeral Diffie-Hellman (DHE) ou Ephemeral Elliptic Curve Diffie-Hellman (ECDHE). A maioria dos sistemas modernos como Java 7 e versões posteriores oferece compatibilidade com esses modos.

Além disso, as solicitações devem ser assinadas usando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou você pode usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

Como se conectar a um cluster do Amazon MSK

Por padrão, os clientes só podem acessar um cluster do MSK se estiverem na mesma VPC do cluster. Toda comunicação entre seus clientes Kafka e seu cluster do MSK é privada por padrão e seus dados de streaming nunca cruzam a Internet. Para estabelecer conexão com seu cluster do MSK usando um cliente que esteja na mesma VPC do cluster, certifique-se de que o grupo de segurança do cluster tenha uma regra de entrada que aceite o tráfego do grupo de segurança do cliente. Para obter informações sobre como configurar essas regras, consulte Regras do grupo de segurança. Para obter um exemplo de como acessar um cluster de uma instância do Amazon EC2 que esteja na mesma VPC do cluster, consulte Conceitos básicos.

Para se conectar ao seu cluster MSK a partir de um cliente que está fora da VPC do cluster, consulte Acesso de AWS dentro, mas de fora da VPC do cluster.

Tópicos

- Acesso público
- Acesso de dentro AWS , mas de fora da VPC do cluster

Acesso público

O Amazon MSK oferece a opção de ativar o acesso público aos agentes dos clusters do MSK que executem o Apache Kafka 2.6.0 ou versões posteriores. Por motivos de segurança, você não pode ativar o acesso público ao criar um cluster do MSK. No entanto, você pode atualizar um cluster existente para torná-lo acessível ao público. Você pode criar um novo cluster e atualizá-lo para torná-lo acessível publicamente.

Você pode ativar o acesso público a um cluster MSK sem custo adicional, mas os custos padrão de transferência de AWS dados se aplicam à transferência de dados para dentro e para fora do cluster. Para obter informações sobre os preços, consulte Preço do Amazon EC2 sob demanda.

Para ativar o acesso público a um cluster, primeiro certifique-se de que o cluster atenda a todas as seguintes condições:

- As sub-redes associadas ao cluster devem ser públicas. Isso significa que as sub-redes devem ter uma tabela de rotas associada a um gateway da Internet conectado. Para obter mais informações sobre como criar e anexar um gateway da Internet, consulte Gateways da Internet no Guia do usuário da Amazon VPC.

- O controle de acesso não autenticado deve estar desativado e pelo menos um dos seguintes métodos de controle de acesso deve estar ativado: SASL/IAM, SASL/SCRAM, mTLS. Para obter informações sobre como atualizar o método de controle de acesso de um cluster, consulte the section called “Atualizar a segurança”.
- A criptografia dentro do cluster deve estar ativada. A configuração ativada é o padrão ao criar um cluster. Não é possível ativar a criptografia dentro do cluster para um cluster que tenha sido criado com ela desativada. Portanto, não é possível ativar o acesso público para um cluster que tenha sido criado com a criptografia no cluster desativada.
- O tráfego de texto simples entre agentes e clientes deve estar desativado. Para obter informações sobre como desativá-lo se estiver ativado, consulte the section called “Atualizar a segurança”.
- Se você estiver usando os métodos de controle de acesso SASL/SCRAM ou mTLS, deverá definir as ACLs do Apache Kafka para seu cluster. Após definir as ACLs do Apache Kafka para seu cluster, atualize a configuração do cluster para que a propriedade `allow.everyone.if.no.acl.found` seja falsa para o cluster. Para obter informações sobre como atualizar a configuração de um cluster, consulte the section called “Operações de configuração”. Se você estiver usando o controle de acesso do IAM e quiser aplicar políticas de autorização ou atualizar suas políticas de autorização, consulte the section called “Controle de acesso do IAM”. Para obter mais informações sobre ACLs do Apache Kafka, consulte the section called “ACLs do Apache Kafka”.

Depois de garantir que um cluster MSK atenda às condições listadas acima, você pode usar a API AWS Management Console AWS CLI, a ou a Amazon MSK para ativar o acesso público. Depois de ativar o acesso público a um cluster, você pode obter uma string pública de agentes de bootstrap para ele. Para obter informações sobre a obtenção de agentes de bootstrap para um cluster, consulte the section called “Como obter os agentes de bootstrap”.

> **Important**
>
> Além de ativar o acesso público, certifique-se de que os grupos de segurança do cluster tenham regras de TCP de entrada que permitam acesso público do seu endereço IP. Recomendamos tornar essas regras o mais restritivas possível. Para obter mais informações sobre grupos de segurança e regras de entrada, consulte Grupos de segurança para sua VPC no Guia do usuário da Amazon VPC. Para obter os números das portas, consulte the section called “Informações de porta”. Para obter instruções sobre como alterar o grupo de segurança de um cluster, consulte the section called “Alterar os grupos de segurança”.

> **Note**
>
> Se você usar as instruções a seguir para ativar o acesso público e ainda não conseguir acessar o cluster, consulte the section called “Não é possível acessar o cluster que está com o acesso público ativado”.

Ativar o acesso público usando o console

1. Faça login no AWS Management Console e abra o console do Amazon MSK em https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Na lista de clusters, selecione o cluster ao qual deseja ativar o acesso público.
3. Escolha a guia Propriedades e, em seguida, encontre a seção Configurações de rede.
4. Escolha Editar acesso público.

Ativando o acesso público usando o AWS CLI

1. Execute o AWS CLI comando a seguir, substituindo a *Current-Cluster-Version pelo ARN ClusterArne pela versão* atual do cluster. Para encontrar a versão atual do cluster, use a DescribeClusteroperação ou o comando AWS CLI describe-cluster. Uma versão de exemplo é `KTVPDKIKX0DER`.

   ```
   aws kafka update-connectivity --cluster-arn ClusterArn --current-
   version Current-Cluster-Version --connectivity-info '{"PublicAccess": {"Type":
    "SERVICE_PROVIDED_EIPS"}}'
   ```

   A saída desse comando `update-connectivity` é semelhante ao seguinte JSON de exemplo.

   ```
   {
       "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**
>
> Para desativar o acesso público, use um AWS CLI comando semelhante, mas com as seguintes informações de conectividade:
>
> ```
> '{"PublicAccess": {"Type": "DISABLED"}}'
> ```

2. Para obter o resultado da `update-connectivity` operação, execute o comando a seguir, substituindo *`ClusterOperationArn`* pelo ARN obtido na saída do `update-connectivity` comando.

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

A saída desse comando `describe-cluster-operation` é semelhante ao seguinte JSON de exemplo.

```
{
    "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"
                }
            }
```

```
        }
    }
}
```

Se `OperationState` tiver o valor `UPDATE_IN_PROGRESS`, aguarde um pouco e execute o comando `describe-cluster-operation` novamente.

Ativar o acesso público usando a API do Amazon MSK

- Para usar a API para ativar ou desativar o acesso público a um cluster, consulte UpdateConnectivity.

> **Note**
>
> Por motivos de segurança, o Amazon MSK não permite acesso público aos nós controladores Apache ZooKeeper ou Kraft.

Acesso de dentro AWS , mas de fora da VPC do cluster

Para se conectar a um cluster MSK de dentro AWS , mas de fora da Amazon VPC do cluster, existem as seguintes opções.

Emparelhamento do Amazon VPC

Para se conectar ao seu cluster do MSK de uma VPC diferente daquela do cluster, é possível criar uma conexão de emparelhamento entre duas VPCs. Para obter informações sobre o emparelhamento da VPC, consulte Guia de emparelhamento do Amazon VPC.

AWS Direct Connect

AWS Direct Connect conecta sua rede local a AWS mais de um cabo de fibra óptica Ethernet padrão de 1 gigabit ou 10 gigabit. Uma extremidade do cabo está conectada ao roteador e a outra ao AWS Direct Connect roteador. Com essa conexão estabelecida, você pode criar interfaces virtuais diretamente na AWS nuvem e na Amazon VPC, ignorando os provedores de serviços de Internet em seu caminho de rede. Para ter mais informações, consulte AWS Direct Connect.

AWS Transit Gateway

AWS Transit Gateway é um serviço que permite conectar suas VPCs e suas redes locais a um único gateway. Para obter informações sobre como usar o AWS Transit Gateway, consulte AWS Transit Gateway.

Conexões da VPN

É possível conectar a VPC do cluster do MSK a redes remotas e usuários que usam as opções de conectividade por VPN descritas no seguinte tópico: Conexões por VPN.

Proxies REST

É possível instalar um proxy REST em uma instância sendo executada na Amazon VPC do cluster. Os proxies REST permitem que os produtores e os consumidores se comuniquem com o cluster por meio de solicitações HTTP de API.

Conectividade multi-VPC em múltiplas regiões

O seguinte documento descreve as opções de conectividade de várias VPCs que residem em diferentes regiões: Conectividade de várias VPCs em múltiplas regiões.

Conectividade privada multi-VPC de região única

A conectividade privada de várias VPCs (alimentada por AWS PrivateLink) para clusters Amazon Managed Streaming for Apache Kafka (Amazon MSK) é um recurso que permite que você conecte mais rapidamente clientes Kafka hospedados em diferentes nuvens privadas virtuais (VPCs) e contas a um cluster Amazon MSK. AWS

Consulte Conectividade multi-VPC de região única para clientes entre contas.

A rede EC2-Classic foi descontinuada

O Amazon MSK não oferece mais suporte a instâncias do Amazon EC2 em execução com a rede Amazon EC2-Classic.

Consulte O EC2-Classic Networking está sendo descontinuado — veja como se preparar.

Conectividade privada multi-VPC do Amazon MSK em uma única região

A conectividade privada de várias VPCs (alimentada por AWS PrivateLink) para clusters Amazon Managed Streaming for Apache Kafka (Amazon MSK) é um recurso que permite que você conecte mais rapidamente clientes Kafka hospedados em diferentes nuvens privadas virtuais (VPCs) e contas a um cluster Amazon MSK. AWS

A conectividade privada multi-VPC é uma solução gerenciada que simplifica a infraestrutura de rede para conectividade multi-VPCs e entre contas. Os clientes podem se conectar ao cluster Amazon MSK PrivateLink sem deixar de manter todo o tráfego na AWS rede. A conectividade privada de várias VPCs para clusters do Amazon MSK está disponível em todas as regiões em AWS que o Amazon MSK está disponível.

Tópicos

O que é conectividade privada multi-VPC?

A conectividade privada de várias VPCs para o Amazon MSK é uma opção de conectividade que permite conectar clientes Apache Kafka hospedados em diferentes nuvens privadas virtuais (VPCs) e contas a um cluster MSK. AWS

O Amazon MSK simplifica o acesso entre contas com políticas de cluster. Essas políticas permitem que o proprietário do cluster conceda permissões para que outras AWS contas estabeleçam conectividade privada com o cluster MSK.

Benefícios da conectividade privada multi-VPC

A conectividade privada multi-VPC tem várias vantagens em relação a outras soluções de conectividade:

- Ele automatiza o gerenciamento operacional da solução de AWS PrivateLink conectividade.
- Ela permite a sobreposição de IPs em VPCs conectadas, eliminando a necessidade de manter IPs não sobrepostos, emparelhamento complexo e tabelas de roteamento associadas a outras soluções de conectividade de VPC.

Você usa uma política de cluster para seu cluster MSK para definir quais AWS contas têm permissões para configurar a conectividade privada entre contas com seu cluster MSK. O administrador de várias contas pode delegar permissões aos perfis ou usuários adequados. Quando usada com a autenticação de cliente do IAM, você também pode usar a política de cluster para definir as permissões do plano de dados do Kafka de modo granular para os clientes conectados.

Requisitos e limitações para conectividade privada multi-VPC

Observe estes requisitos de cluster do MSK para executar a conectividade privada multi-VPC:

- A conectividade privada multi-VPC é compatível apenas com o Apache Kafka 2.7.1 ou superior. Certifique-se de que todos os clientes que você use com o cluster do MSK estejam executando versões do Apache Kafka compatíveis com o cluster.
- A conectividade privada multi-VPC é compatível com os tipos de autenticação IAM, TLS e SASL/SCRAM. Clusters não autenticados não podem usar conectividade privada multi-VPC.
- Se você estiver usando os métodos de controle de acesso SASL/SCRAM ou mTLS, deverá definir as ACLs do Apache Kafka para seu cluster. Primeiro, defina as ACLs do Apache Kafka para seu cluster. Em seguida, atualize a configuração do cluster para que a propriedade `allow.everyone.if.no.acl.found` seja definida como falsa para o cluster. Para obter informações sobre como atualizar a configuração de um cluster, consulte the section called “Operações de configuração”. Se você estiver usando o controle de acesso do IAM e quiser aplicar políticas de autorização ou atualizar suas políticas de autorização, consulte the section called “Controle de acesso do IAM”. Para obter mais informações sobre ACLs do Apache Kafka, consulte the section called “ACLs do Apache Kafka”.
- A conectividade privada multi-VPC não é compatível com o tipo de instância t3.small.
- A conectividade privada de várias VPCs não é suportada em todas AWS as regiões, somente em AWS contas dentro da mesma região.
- O Amazon MSK não é compatível com conectividade privada multi-VPC com os nós do Zookeeper.

Conceitos básicos sobre como usar a conectividade privada multi-VPC

Tópicos

Este tutorial usa um caso de uso comum como exemplo de como você pode usar a conectividade de várias VPCs para conectar de forma privada um cliente Apache Kafka a um cluster MSK de dentro, AWS mas fora da VPC do cluster. Esse processo exige que o usuário entre contas crie uma conexão e uma configuração de VPC gerenciada pelo MSK para cada cliente, incluindo as permissões de cliente necessárias. O processo também exige que o proprietário do cluster MSK habilite a PrivateLink conectividade no cluster MSK e selecione esquemas de autenticação para controlar o acesso ao cluster.

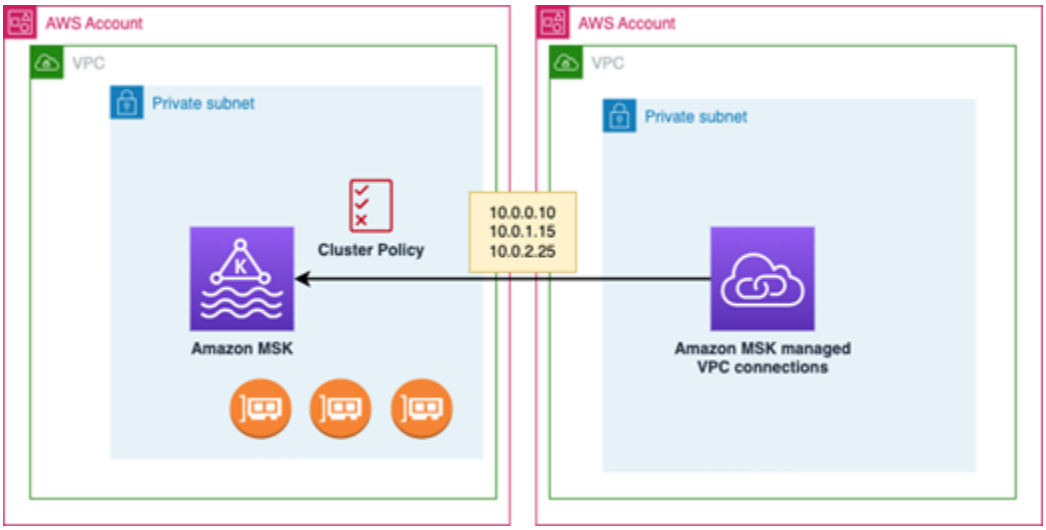
Em diferentes partes deste tutorial, escolhemos as opções aplicáveis a esse exemplo. Isso não significa que estas são as únicas opções disponíveis para configurar um cluster do MSK ou instâncias de cliente.

A configuração de rede para esse caso de uso é a seguinte:

- Um usuário com várias contas (cliente Kafka) e um cluster do MSK estão na mesma rede/região da AWS , mas em contas diferentes:
  - Cluster do MSK na conta A
  - Cliente Kafka na conta B
- O usuário entre contas se conectará de modo privado ao cluster do MSK usando o esquema de autenticação do IAM.

Este tutorial pressupõe que há um cluster do MSK provisionado criado com o Apache Kafka versão 2.7.1 ou superior. O cluster do MSK deve estar em um estado ACTIVE antes de iniciar o processo de configuração. Para evitar possíveis perdas de dados ou tempo de inatividade, os clientes que usarão uma conexão privada multi-VPC para se conectar ao cluster devem usar versões do Apache Kafka compatíveis com o cluster.

O diagrama a seguir ilustra a arquitetura da conectividade multi-VPC do Amazon MSK conectada a um cliente em uma conta diferente. AWS



Etapa 1: no cluster do MSK na conta A, ativar a conectividade multi-VPC para o esquema de autenticação do IAM no cluster

O proprietário do cluster do MSK precisa fazer as configurações no cluster do MSK após a criação do cluster e em um estado ACTIVE.

O proprietário do cluster ativa a conectividade privada multi-VPC no cluster ACTIVE para qualquer esquema de autenticação que estará ativo no cluster. Isso pode ser feito usando a UpdateSecurity API ou o console MSK. Os esquemas de autenticação do IAM, SASL/SCRAM e TLS são compatíveis com conectividade privada multi-VPC. A conectividade privada multi-VPC não pode ser habilitada para clusters não autenticados.

Para esse caso de uso, você configurará o cluster para usar o esquema de autenticação do IAM.

> **Note**
>
> Se você estiver configurando seu cluster do MSK para usar o esquema de autenticação SASL/SCRAM, a propriedade "`allow.everyone.if.no.acl.found=false`" das ACLs do Apache Kafka será obrigatória. Consulte ACLs do Apache Kafka.

Quando você atualiza as configurações de conectividade privada multi-VPC, o Amazon MSK inicia uma reinicialização contínua dos nós do agente que atualiza as configurações do agente. A conclusão dessa operação pode levar até 30 minutos ou mais. Você não pode fazer outras atualizações no cluster enquanto a conectividade estiver sendo atualizada.

Ativar o recurso multi-VPC para esquemas de autenticação selecionados no cluster na conta A usando o console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/ para a conta do cluster.
2. No painel de navegação, em Clusters do MSK, escolha Clusters para exibir a lista de clusters na conta.
3. Selecione o cluster a ser configurado para conectividade privada multi-VPC. O cluster deve estar em um estado ACTIVE.
4. Selecione a guia Propriedades do cluster e acesse as configurações de Rede.
5. Selecione o menu suspenso Editar e selecione Ativar conectividade multi-VPC.
6. Selecione um ou mais tipos de autenticação que você deseja ativar para esse cluster. Para esse caso de uso, selecione a autenticação baseada em perfil do IAM.
7. Selecione Save Changes (Salvar alterações).

Example - UpdateConnectivity API que ativa esquemas de autenticação de conectividade privada de várias VPCs em um cluster

Como alternativa ao console MSK, você pode usar a UpdateConnectivity API para ativar a conectividade privada de várias VPCs e configurar esquemas de autenticação em um cluster ATIVO. O exemplo a seguir mostra o esquema de autenticação do IAM ativado para o cluster.

```
{
  "currentVersion": "K3T4TT2Z381HKD",
  "connectivityInfo": {
    "vpcConnectivity": {
      "clientAuthentication": {
        "sasl": {
          "iam": {
            "enabled": TRUE
            }
        }
      }
    }
  }
}
```

O Amazon MSK cria a infraestrutura de rede necessária para conectividade privada. O Amazon MSK também cria um novo conjunto de endpoints do agente de bootstrap para cada tipo de autenticação que requer conectividade privada. Observe que o esquema de autenticação em texto simples não oferece suporte à conectividade privada multi-VPC.

Etapa 2: anexar uma política de cluster ao cluster do MSK

O proprietário do cluster pode anexar uma política de cluster (também conhecida como política baseada em recurso) ao cluster do MSK no qual você ativará a conectividade privada multi-VPC. A política de cluster permite que os clientes acessem o cluster usando outra conta. Antes de editar a política de cluster, você precisa dos IDs de conta para as contas que devem ter permissão para acessar o cluster do MSK. Consulte Como o Amazon MSK funciona com o IAM.

O proprietário do cluster deve anexar uma política de cluster ao cluster do MSK que autorize o usuário entre contas na conta B a obter agentes de bootstrap para o cluster e a autorizar as seguintes ações no cluster do MSK na conta A:

- CreateVpcConexão
- GetBootstrapCorretores
- DescribeCluster
- DescribeClusterV2

Example

Para referência, veja a seguir um exemplo do JSON para uma política básica de cluster, semelhante à política padrão apresentada no editor de políticas do IAM do console do 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"
        }
    ]
}
```

Anexar uma política de cluster ao cluster do MSK

1. No console do Amazon MSK, em Clusters do MSK, escolha Clusters.
2. Role para baixo até Configurações de segurança e selecione Editar política de cluster.
3. No console, na tela Editar política de cluster, selecione Política básica para conectividade multi-VPC.
4. No campo ID da conta, insira o ID da conta para cada conta que deve ter permissão para acessar esse cluster. Conforme você digita o ID, ele é copiado automaticamente para a sintaxe JSON da política exibida. Em nosso exemplo de política de cluster, o ID da conta é `123456789012`.
5. Selecione Save Changes (Salvar alterações).

Para obter informações sobre as APIs de políticas de cluster, consulte Políticas baseadas em recurso do Amazon MSK.

Etapa 3: ações de usuários entre contas para configurar conexões de VPC gerenciadas pelo cliente

Para configurar a conectividade privada multi-VPC entre um cliente em uma conta diferente do cluster do MSK, o usuário entre contas cria uma conexão VPC gerenciada para o cliente. É possível conectar vários clientes ao cluster do MSK repetindo esse procedimento. Para fins desse caso de uso, você configurará apenas um cliente.

Os clientes podem usar os esquemas de autenticação compatíveis IAM, SASL/SCRAM ou TLS. Cada conexão de VPC gerenciada só pode ter um esquema de autenticação associado a ela. O esquema de autenticação do cliente deve ser configurado no cluster do MSK ao qual o cliente se conectará.

Para esse caso de uso, configure o esquema de autenticação do cliente para que o cliente na conta B use o esquema de autenticação do IAM.

Pré-requisitos

Esse processo requer os seguintes itens:

- A política de cluster criada anteriormente que concede ao cliente na conta B permissão para realizar ações no cluster do MSK na conta A.
- Uma política de identidade anexada ao cliente na Conta B que concede permissões para `kafka:CreateVpcConnection`ec2:CreateTags, `ec2:CreateVPCEndpoint` e `ec2:DescribeVpcAttribute` ação.

Example

Para referência, este é um exemplo do JSON para uma política básica de identidade de cliente.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kafka:CreateVpcConnection",
        "ec2:CreateTags",
        "ec2:CreateVPCEndpoint",
        "ec2:DescribeVpcAttribute"
      ],
      "Resource": "*"
    }
  ]
}
```

Para criar uma conexão de VPC gerenciada para um cliente na conta B

1. Do administrador do cluster, obtenha o ARN do cluster do MSK na conta A ao qual você deseja que o cliente na conta B se conecte. Anote o ARN do cluster para usar posteriormente.
2. No console do MSK da conta B do cliente, escolha Conexões VPC gerenciadas e, em seguida, escolha Criar conexão.
3. No painel Configurações de conexão, cole o ARN do cluster no campo de texto ARN do cluster e escolha Verificar.

4. Selecione o Tipo de autenticação para o cliente na conta B. Para esse caso de uso, escolha IAM ao criar a conexão VPC do cliente.
5. Escolha a VPC para o cliente.
6. Escolha pelo menos duas zonas de disponibilidade e sub-redes associadas. Você pode obter os IDs da zona de disponibilidade nos detalhes do cluster do AWS Management Console ou usando a DescribeClusterAPI ou o comando da AWS CLI describe-cluster. As IDs de zona que você especifica para a sub-rede do cliente devem corresponder às da sub-rede do cluster. Se os valores de uma sub-rede estiverem ausentes, primeiro crie uma sub-rede com o mesmo ID de zona do seu cluster do MSK.
7. Escolha um Grupo de segurança para essa conexão VPC. Você pode usar o grupo de segurança padrão. Para mais informações sobre a configuração de grupos de segurança, consulte Controlar o tráfego para recursos usando grupos de segurança.
8. Selecione Criar conexão.
9. Para obter a lista das novas strings de agente de bootstrap no console do MSK do usuário entre contas (Detalhes do cluster > Conexão VPC gerenciada), consulte as strings de agente de bootstrap exibidas em “Cadeia de conexão do cluster”. Na Conta B do cliente, a lista de corretores de bootstrap pode ser visualizada chamando a API GetBootstrapBrokers ou visualizando a lista de corretores de bootstrap nos detalhes do cluster do console.
10. Atualize os grupos de segurança associados às conexões de VPC da seguinte forma:

    a. Defina regras de entrada para a PrivateLink VPC para permitir todo o tráfego do intervalo de IP da rede da Conta B.

    b. [Opcional] Defina as regras de conectividade de saída para o cluster do MSK. Escolha o grupo de segurança no console da VPC, Editar regras de saída e adicione uma regra para o Tráfego TCP personalizado para os intervalos de portas 14001-14100. O balanceador de carga de rede multi-VPC está escutando nos intervalos de portas 14001-14100. Consulte Network Load Balancers.
11. Configure o cliente na conta B para usar os novos agentes de bootstrap para conectividade privada multi-VPC para se conectar ao cluster do MSK na conta A. Consulte Produzir e consumir dados.

Após a conclusão da autorização, o Amazon MSK criará uma conexão VPC gerenciada para cada VPC e esquema de autenticação especificados. O grupo de segurança escolhido será associado a cada conexão. Essa conexão VPC gerenciada é configurada pelo Amazon MSK para se conectar

de maneira privada aos agentes. Você pode usar o novo conjunto de agentes de bootstrap para se conectar de maneira privada ao cluster do Amazon MSK.

Atualizar os esquemas de autorização em um cluster

A conectividade privada multi-VPC é compatível com vários esquemas de autorização: SASL/SCRAM, IAM e TLS. O proprietário do cluster pode ativar/desativar a conectividade privada para um ou mais esquemas de autenticação. O cluster precisa estar no estado ACTIVE para realizar essa ação.

Para ativar um esquema de autenticação usando o console do Amazon MSK

1. Abra o console do Amazon MSK em AWS Management Console para o cluster que deseja editar.
2. No painel de navegação, em Clusters do MSK, escolha Clusters para exibir a lista de clusters na conta.
3. Selecione o cluster que deseja editar. O cluster deve estar em um estado ACTIVE.
4. Selecione a guia Propriedades do cluster e acesse Configurações de rede.
5. Selecione o menu suspenso Editar e selecione Ativar conectividade multi-VPC para ativar o novo esquema de autorização.
6. Selecione um ou mais tipos de autenticação que você deseja ativar para esse cluster.
7. Selecione Ativar seleção.

Ao ativar um novo esquema de autenticação, você também deverá criar novas conexões VPC gerenciadas para o novo esquema de autenticação e atualizar seus clientes para usar os agentes de bootstrap específicos do novo esquema de autenticação.

Para desativar um esquema de autenticação usando o console do Amazon MSK

> **Note**
>
> Quando você desativa a conectividade privada multi-VPC para esquemas de autenticação, toda a infraestrutura relacionada à conectividade é excluída, incluindo as conexões VPC gerenciadas.

Quando você desativa a conectividade privada multi-VPC para esquemas de autenticação, as conexões VPC existentes no lado do cliente mudam para INACTIVE, e a infraestrutura do Privatelink no lado do cluster é removida, incluindo as conexões VPC gerenciadas. O usuário de várias contas só pode excluir a conexão VPC inativa. Se a conectividade privada for ativada novamente no cluster, o usuário entre contas precisará criar uma nova conexão com o cluster.

1. Abra o console do Amazon MSK em AWS Management Console.
2. No painel de navegação, em Clusters do MSK, escolha Clusters para exibir a lista de clusters na conta.
3. Selecione o cluster que deseja editar. O cluster deve estar em um estado ACTIVE.
4. Selecione a guia Propriedades do cluster e acesse Configurações de rede.
5. Selecione o menu suspenso Editar e selecione Desativar conectividade multi-VPC (para desativar um esquema de autorização).
6. Selecione um ou mais tipos de autenticação que você deseja desativar para esse cluster.
7. Selecione Desativar seleção.

Example Para ativar/desativar um esquema de autenticação com a API

Como alternativa ao console MSK, você pode usar a UpdateConnectivity API para ativar a conectividade privada de várias VPCs e configurar esquemas de autenticação em um cluster ATIVO. O exemplo a seguir mostra os esquemas de autenticação do IAM e SASL/SCRAM ativados para o cluster.

Ao ativar um novo esquema de autenticação, você também deverá criar novas conexões VPC gerenciadas para o novo esquema de autenticação e atualizar seus clientes para usar os agentes de bootstrap específicos do novo esquema de autenticação.

Quando você desativa a conectividade privada multi-VPC para esquemas de autenticação, as conexões VPC existentes no lado do cliente mudam para INACTIVE, e a infraestrutura do Privatelink no lado do cluster é removida, incluindo as conexões VPC gerenciadas. O usuário de várias contas só pode excluir a conexão VPC inativa. Se a conectividade privada for ativada novamente no cluster, o usuário entre contas precisará criar uma nova conexão com o 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"
}
```

Rejeitar uma conexão VPC gerenciada com um cluster do Amazon MSK

Você pode rejeitar uma conexão VPC do cliente no console do Amazon MSK na conta de administrador do cluster. Para ser rejeitada, a conexão VPC do cliente deve estar no estado AVAILABLE. Talvez você queira rejeitar uma conexão VPC gerenciada de um cliente que não esteja mais autorizado a se conectar ao seu cluster. Para evitar que novas conexões VPC gerenciadas se conectem a um cliente, negue o acesso ao cliente na política de cluster. Uma conexão rejeitada ainda gera custos até ser excluída pelo proprietário da conexão. Consulte Excluir uma conexão VPC gerenciada com um cluster do Amazon MSK.

Para rejeitar uma conexão VPC do cliente usando o console do MSK

1. Abra o console do Amazon MSK em AWS Management Console.
2. No painel de navegação, selecione Clusters e localize a lista Configurações de rede > Conexões VPC do cliente.

3. Selecione a conexão que deseja rejeitar e selecione Rejeitar conexão VPC do cliente.
4. Confirme que deseja rejeitar a conexão VPC do cliente selecionada.

Para rejeitar uma conexão VPC gerenciada usando a API, use a API `RejectClientVpcConnection`.

Excluir uma conexão VPC gerenciada com um cluster do Amazon MSK

O usuário entre contas pode excluir uma conexão VPC gerenciada para um cluster do MSK no console da conta do cliente. Como o usuário proprietário do cluster não é proprietário da conexão VPC gerenciada, a conexão não pode ser excluída na conta de administrador do cluster. Após a exclusão de uma conexão VPC, ela não terá mais custos.

Para excluir uma conexão VPC gerenciada usando o console do MSK

1. Na conta do cliente, abra o console do Amazon MSK em AWS Management Console.
2. No painel de navegação, selecione Conexões VPC gerenciadas.
3. Na lista de conexões, selecione a conexão que deseja excluir.
4. Confirme que deseja excluir a conexão VPC.

Para excluir uma conexão VPC gerenciada usando a API, use a API `DeleteVpcConnection`.

Permissões para conectividade privada multi-VPC

Esta seção resume as permissões necessárias para clientes e clusters que usam o recurso de conectividade privada multi-VPC. A conectividade privada multi-VPC exige que o administrador do cliente crie permissões em cada cliente que terá uma conexão VPC gerenciada com o cluster do MSK. Também exige que o administrador do cluster MSK habilite a PrivateLink conectividade no cluster MSK e selecione esquemas de autenticação para controlar o acesso ao cluster.

Tipo de autenticação de cluster e permissões de acesso a tópicos

Ative o recurso de conectividade privada multi-VPC para esquemas de autenticação habilitados para seu cluster do MSK. Consulte Requisitos e limitações para conectividade privada multi-VPC. Se você estiver configurando seu cluster do MSK para usar o esquema de autenticação SASL/SCRAM, a propriedade "`allow.everyone.if.no.acl.found=false`" das ACLs do Apache Kafka será obrigatória. Após definir as ACLs do Apache Kafka para seu cluster, atualize a configuração do cluster para que a propriedade `allow.everyone.if.no.acl.found` seja falsa para o cluster.

Para obter informações sobre como atualizar a configuração de um cluster, consulte Operações de configuração do Amazon MSK.

Permissões de política de cluster entre contas

Se um cliente Kafka estiver em uma AWS conta diferente do cluster MSK, anexe uma política baseada em cluster ao cluster MSK que autorize o usuário raiz do cliente a conectividade entre contas. Você pode editar a política de cluster multi-VPC usando o editor de políticas do IAM no console do MSK (Configurações de segurança do cluster > Editar política de cluster) ou usar as seguintes APIs para gerenciar a política de cluster:

PutClusterPolítica

Anexa a política de cluster ao cluster. Você pode usar essa API para criar ou atualizar a política de cluster do MSK especificada. Se você estiver atualizando a política, o campo currentVersion será obrigatório na carga da solicitação.

GetClusterPolítica

Recupera o texto JSON do documento de política de cluster anexado ao cluster.

DeleteClusterPolítica

Exclui a política de cluster.

Veja a seguir um exemplo do JSON para uma política básica de cluster, semelhante à política padrão apresentada no editor de políticas do IAM do console do 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"
        }
    ]
}
```

Permissões de cliente para conectividade privada multi-VPC com um cluster do MSK

Para configurar a conectividade privada multi-VPC entre um cliente Kafka e um cluster do MSK, o cliente precisa ter uma política de identidade anexada que conceda permissões para as ações `kafka:CreateVpcConnection`, `ec2:CreateTags` e `ec2:CreateVPCEndpoint` no cliente. Para referência, este é um exemplo do JSON para uma política básica de identidade de cliente.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kafka:CreateVpcConnection",
        "ec2:CreateTags",
        "ec2:CreateVPCEndpoint"
      ],
      "Resource": "*"
    }
  ]
}
```

Informações de porta

Use os seguintes números de porta para que o Amazon MSK possa se comunicar com máquinas clientes:

- Para se comunicar com agentes em texto simples, os agentes usam a porta 9092.
- Para se comunicar com corretores com criptografia TLS, use a porta 9094 para acesso interno AWS e a porta 9194 para acesso público.
- Para se comunicar com corretores com SASL/SCRAM, use a porta 9096 para acesso interno AWS e a porta 9196 para acesso público.

- Para se comunicar com corretores em um cluster configurado para usothe section called “Controle de acesso do IAM”, use a porta 9098 para acesso interno AWS e a porta 9198 para acesso público.
- Para se comunicar com o Apache ZooKeeper usando a criptografia TLS, use a porta 2182. ZooKeeper Os nós Apache usam a porta 2181 por padrão.

Migração para um cluster do Amazon MSK

O replicador do Amazon MSK pode ser usado para a migração do cluster do MSK. Consulte O que é o replicador do Amazon MSK?. Como alternativa, você pode usar o Apache MirrorMaker 2.0 para migrar de um cluster não MSK para um cluster Amazon MSK. Para obter um exemplo de como fazer isso, consulte Migrar um cluster Apache Kafka local para o Amazon MSK usando. MirrorMaker Para obter informações sobre como usar MirrorMaker, consulte Espelhamento de dados entre clusters na documentação do Apache Kafka. Recomendamos a configuração MirrorMaker em uma configuração altamente disponível.

Um resumo das etapas a serem seguidas ao usar MirrorMaker para migrar para um cluster MSK

1. Criar o cluster de destino do MSK
2. Comece MirrorMaker com uma instância do Amazon EC2 dentro da mesma Amazon VPC do cluster de destino.
3. Inspecione o MirrorMaker atraso.
4. Depois de MirrorMaker se atualizar, redirecione produtores e consumidores para o novo cluster usando os corretores de bootstrap do cluster MSK.
5. Desligar MirrorMaker.

Migração do cluster do Apache Kafka para o Amazon MSK

Suponha que você tenha um cluster do Apache Kafka chamado `CLUSTER_ONPREM`. Esse cluster é preenchido com tópicos e dados. Se quiser migrar esse cluster para um cluster recém-criado do Amazon MSK chamado `CLUSTER_AWSMSK`, esse procedimento fornecerá uma visualização de alto nível das etapas que você deverá seguir.

Para migrar o cluster existente do Apache Kafka para o Amazon MSK

1. No `CLUSTER_AWSMSK`, crie todos os tópicos que deseja migrar.

   Você não pode usar MirrorMaker essa etapa porque ela não recria automaticamente os tópicos que você deseja migrar com o nível de replicação correto. Você pode criar os tópicos no Amazon MSK com os mesmos fatores de replicação e números de partições que eles tinham em `CLUSTER_ONPREM`. Você também pode criar os tópicos com diferentes fatores de replicação e números de partições.

2. Comece MirrorMaker com uma instância que tenha acesso de leitura `CLUSTER_ONPREM` e gravação `CLUSTER_AWSMSK` a.
3. Execute o seguinte comando para espelhar todos os tópicos:

```
<path-to-your-kafka-installation>/bin/kafka-mirror-maker.sh --consumer.config
 config/mirrormaker-consumer.properties --producer.config config/mirrormaker-
producer.properties --whitelist '.*'
```

   Nesse comando, `config/mirrormaker-consumer.properties` aponta para um agente de bootstrap no `CLUSTER_ONPREM`; por exemplo, `bootstrap.servers=localhost:9092`. E `config/mirrormaker-producer.properties` aponta para um corretor de bootstrap em CLUSTER_AWSMSK; por exemplo,. `bootstrap.servers=10.0.0.237:9092,10.0.2.196:9092,10.0.1.233:9092`
4. Continue MirrorMaker executando em segundo plano e continue usando`CLUSTER_ONPREM`. MirrorMaker espelha todos os novos dados.
5. Verifique o progresso do espelhamento inspecionando o intervalo entre o último deslocamento de cada tópico e o deslocamento atual do qual está sendo consumido. MirrorMaker

   Lembre-se de que MirrorMaker é simplesmente usar um consumidor e um produtor. Portanto, você pode verificar o atraso usando a ferramenta `kafka-consumer-groups.sh`. Para localizar o nome do grupo de consumidores, procure o `group.id` no arquivo `mirrormaker-consumer.properties` e use seu valor. Se essa chave não existir no arquivo, você poderá criá-la. Por exemplo, defina `group.id=mirrormaker-consumer-group`.
6. Depois de MirrorMaker terminar de espelhar todos os tópicos, pare todos os produtores e consumidores e, em seguida, pare. MirrorMaker Redirecione os produtores e consumidores para o cluster `CLUSTER_AWSMSK` alterando seus valores dos agentes de bootstrap dos produtores e consumidores. Reinicie todos os produtores e consumidores no `CLUSTER_AWSMSK`.

Migração de um cluster do Amazon MSK para outro

Você pode usar o Apache MirrorMaker 2.0 para migrar de um cluster não MSK para um cluster MSK. Por exemplo, você pode migrar de uma versão do Apache Kafka para outra. Para obter um exemplo de como fazer isso, consulte Migrar um cluster Apache Kafka local para o Amazon MSK usando. MirrorMaker Como alternativa, o replicador do Amazon MSK pode ser usado para a migração do cluster do MSK. Para obter mais informações sobre o replicador do Amazon MSK, consulte Replicador do MSK.

MirrorMaker 1.0 melhores práticas

Essa lista de melhores práticas se aplica à MirrorMaker versão 1.0.

- Execute MirrorMaker no cluster de destino. Dessa forma, se ocorrer um problema de rede, as mensagens ainda estarão disponíveis no cluster de origem. Se você executa MirrorMaker no cluster de origem e os eventos são armazenados em buffer no produtor e há um problema de rede, os eventos podem ser perdidos.
- Se a criptografia for necessária em trânsito, execute-a no cluster de origem.
- Para os consumidores, defina auto.commit.enabled=false
- Para os produtores, defina
  - max.in.flight.requests.per.connection=1
  - retries=Int.Max_Value
  - acks=all
  - max.block.ms = Long.Max_Value
- Para obter um throughput alto do produtor:
  - Mensagens de buffer e lotes de mensagens de preenchimento: ajuste buffer.memory, batch.size, linger.ms
  - Ajuste os buffers de soquete: receive.buffer.bytes, send.buffer.bytes
- Para evitar a perda de dados, desative a confirmação automática na origem, para que ela MirrorMaker possa controlar as confirmações, o que normalmente acontece depois de receber o pacote do cluster de destino. Se o produtor tiver acks=all e o cluster de destino tiver min.insync.replicas definido como mais de 1, as mensagens persistirão em mais de um agente no destino antes que o consumidor confirme a compensação na origem. MirrorMaker
- Se a ordem for importante, você poderá definir novas tentativas como 0. Como alternativa, para um ambiente de produção, defina conexões máximas em trânsito como 1 para garantir que os lotes enviados não sejam confirmados fora de ordem se um lote falhar no meio. Dessa forma, cada lote enviado é repetido até que o próximo lote seja enviado. Se max.block.ms não estiver definido como o valor máximo, e se o buffer do produtor estiver cheio, poderá haver perda de dados (dependendo de algumas das outras configurações). Isso pode bloquear e retropressionar o consumidor.
- Para obter throughput alto
  - Aumente o buffer.memory.
  - Aumente o tamanho do lote.

  - Ajuste linger.ms para permitir que os lotes sejam preenchidos. Isso também permite uma melhor compactação, menos uso de largura de banda de rede e menos armazenamento no cluster. Isso resulta em maior retenção.
  - Monitore o uso da CPU e da memória.
- Para obter throughput alto do consumidor
  - Aumente o número de threads/consumidores por MirrorMaker processo — num.streams.
  - Aumente o número de MirrorMaker processos nas máquinas antes de aumentar os segmentos para permitir a alta disponibilidade.
  - Aumente o número de MirrorMaker processos primeiro na mesma máquina e depois em máquinas diferentes (com o mesmo ID de grupo).
  - Isole tópicos que tenham uma taxa de transferência muito alta e use instâncias separadas MirrorMaker .
- Para gerenciamento e configuração
  - Ferramentas de gerenciamento de uso AWS CloudFormation e configuração, como Chef e Ansible.
  - Use montagens do Amazon EFS para manter todos os arquivos de configuração acessíveis em todas as instâncias do Amazon EC2.
  - Use contêineres para facilitar o escalonamento e o gerenciamento de MirrorMaker instâncias.
- Normalmente, é preciso mais de um consumidor para saturar um produtor. MirrorMaker Portanto, configure vários consumidores. Primeiro, defina-os em diferentes máquinas para fornecer alta disponibilidade. Depois, ajuste a escala das máquinas individuais até ter um consumidor para cada partição, com consumidores distribuídos igualmente entre máquinas.
- Para obter consumo e entrega de throughput alto, ajuste os buffers de recebimento e envio porque seus padrões podem ser muito baixos. Para obter o máximo desempenho, certifique-se de que o número total de streams (num.streams) corresponda a todas as partições de tópicos que MirrorMaker estão tentando copiar para o cluster de destino.

MirrorMaker 2.* vantagens

- Usa a estrutura e o ecossistema do Apache Kafka Connect.
- Detecta novos tópicos e partições.
- Sincroniza automaticamente a configuração de tópicos entre clusters.
- Oferece suporte a pares de cluster “ativo/ativo”, além de qualquer número de clusters ativos.

- Fornece novas métricas, incluindo latência end-to-end de replicação em vários data centers e clusters.
- Emite deslocamentos necessários para migrar consumidores entre clusters e oferece as ferramentas para a conversão do deslocamento.
- Suporta um arquivo de configuração de alto nível para especificar vários clusters e fluxos de replicação em um só lugar, em comparação com propriedades de produtor/consumidor de baixo nível para cada processo 1.*. MirrorMaker

Como monitorar um cluster do Amazon MSK

Há várias maneiras pelas quais o Amazon MSK ajuda você a monitorar o status de um cluster do Amazon MSK.

- O Amazon MSK ajuda você a monitorar a capacidade de armazenamento em disco ao enviar automaticamente alertas de capacidade de armazenamento quando um cluster está prestes a atingir o limite de capacidade de armazenamento. Os alertas também fornecem recomendações sobre as melhores etapas a serem seguidas para resolver os problemas detectados. Isso ajuda você a identificar e resolver rapidamente os problemas de capacidade de disco antes que eles se tornem críticos. O Amazon MSK envia automaticamente esses alertas para o console do Amazon MSK, para a AWS Health Dashboard Amazon EventBridge e para os contatos de e-mail da sua AWS conta. Para obter mais informações sobre alertas de capacidade de armazenamento, consulte Alertas de capacidade de armazenamento do Amazon MSK.
- O Amazon MSK reúne métricas do Apache Kafka e as envia para a Amazon, CloudWatch onde você pode visualizá-las. Para obter mais informações sobre as métricas do Apache Kafka, incluindo as que surgem com o Amazon MSK, consulte Monitoramento na documentação do Apache Kafka.
- Também é possível monitorar o cluster do MSK com o Prometheus, uma aplicação de código aberto para monitoramento. Para obter informações sobre o Prometheus, consulte Visão geral na documentação do Prometheus. Para saber como monitorar o cluster com o Prometheus, consulte the section called “Monitoramento aberto com o Prometheus”.

Tópicos

Métricas do Amazon MSK para monitoramento com CloudWatch

O Amazon MSK se integra à Amazon CloudWatch para que você possa coletar, visualizar e analisar CloudWatch métricas para seu cluster do Amazon MSK. As métricas que você configura para seu

cluster MSK são automaticamente coletadas e enviadas para CloudWatch. Você pode definir o nível de monitoramento de um cluster do MSK como um dos seguintes: `DEFAULT`, `PER_BROKER`, `PER_TOPIC_PER_BROKER` ou `PER_TOPIC_PER_PARTITION`. As tabelas nas seções a seguir mostram todas as métricas disponíveis em cada nível de monitoramento.

> **Note**
>
> Os nomes de algumas métricas do Amazon MSK para CloudWatch monitoramento foram alterados na versão 3.6.0 e superior. Use os novos nomes para monitorar essas métricas. Para métricas com nomes alterados, a tabela abaixo mostra o nome usado nas versões 3.6.0 e posteriores, seguido pelo nome na versão 2.8.2.tiered.

As métricas no nível `DEFAULT` são gratuitas. Os preços de outras métricas estão descritos na página de CloudWatchpreços da Amazon.

Monitoramento no nível **`DEFAULT`**

As métricas descritas na tabela a seguir estão disponíveis no nível de monitoramento `DEFAULT`. Elas são gratuitas.

Métricas disponíveis no nível de monitoramento **`DEFAULT`**

Nome	Quando visível	Dimensõ	Descrição
`ActiveController Count`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster	Somente um controlador por cluster deve estar ativo em qualquer momento.
`BurstBalance`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O saldo restante dos créditos de intermitência de entrada/saída para volumes do EBS no cluster. Use-o para investigar a latência ou a diminuição do throughput. `BurstBalance` não é relatado para volumes do EBS quando o desempenho de linha de base de um volume for maior que o desempenho

Nome	Quando visível	Dimensõ	Descrição
			máximo de intermitência. Para obter mais informações, consulte Créditos de E/S e desempenho de intermitência.
`BytesInPerSec`	Depois de criar um tópico.	Nome do cluster, ID do agente, tópico	O número de bytes por segundo recebidos dos clientes. Essa métrica está disponível por agente e também por tópico.
`BytesOutPerSec`	Depois de criar um tópico.	Nome do cluster, ID do agente, tópico	O número de bytes por segundo enviados aos clientes. Essa métrica está disponível por agente e também por tópico.
`ClientConnectionCount`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente, autenticação de cliente	O número de conexões de cliente autenticadas e ativas.
`ConnectionCount`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O número de conexões ativas autenticadas, não autenticadas e entre agentes.

Nome	Quando visível	Dimensõ	Descrição
`CPUCredit Balance`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O número de créditos ganhos de CPU que um agente acumulou desde que foi iniciado. Os créditos são acumulados no saldo de créditos após terem sido ganhos e são removidos do saldo de créditos quando são gastos. A falta de saldo de créditos de CPU pode afetar negativamente o desempenho do cluster. Você pode adotar medidas para reduzir a carga da CPU. Por exemplo, você pode reduzir o número de solicitações de clientes ou atualizar o tipo de agente para um tipo de agente M5.
`CpuIdle`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	A porcentagem de tempo ocioso da CPU.
`CpuIoWait`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O percentual de tempo ocioso da CPU durante uma operação de disco pendente.
`CpuSystem`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	A porcentagem de CPU no espaço do kernel.

Nome	Quando visível	Dimensõ	Descrição
`CpuUser`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	A porcentagem de CPU no espaço do usuário.
`GlobalPartitionCount`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster	O número de partições em todos os tópicos no cluster, excluindo réplicas. Como `GlobalPartitionCount` não inclui réplicas, a soma dos `PartitionCount` valores pode ser maior do que GlobalPartitionCount se o fator de replicação de um tópico for maior que 1.
`GlobalTopicCount`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster	Número total de tópicos em todos os agentes no cluster.
`EstimatedMaxTimeLag`	Depois que o grupo de consumidores consome de um tópico.	Grupo de consumi res, tópico	Estimativa de tempo (em segundos) para drenar `MaxOffsetLag` .
`KafkaAppLogsDiskUsed`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	A porcentagem de espaço em disco usada para logs de aplicativos.
`KafkaDataLogsDiskUsed` (dimensão `Cluster Name, Broker ID`)	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	A porcentagem de espaço em disco usada para logs de dados.

Nome	Quando visível	Dimensõ	Descrição
`KafkaData LogsDiskUsed` (dimensão `Cluster Name`)	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster	A porcentagem de espaço em disco usada para logs de dados.
`LeaderCount`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O número total de líderes de partições por agente, sem incluir réplicas.
`MaxOffsetLag`	Depois que o grupo de consumidores consome de um tópico.	Grupo de consumi res, tópico	O atraso máximo de deslocame nto entre todas as partições em um tópico.
`MemoryBuffered`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O tamanho, em bytes, da memória armazenada em buffer para o agente.
`MemoryCached`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O tamanho, em bytes, da memória armazenada em cache para o agente.
`MemoryFree`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O tamanho, em bytes, de memória que é gratuita e disponível para o agente.

Nome	Quando visível	Dimensõ	Descrição
`HeapMemoryAfterGC`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O percentual da memória total da pilha em uso após a coleta de resíduos.
`MemoryUsed`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O tamanho, em bytes, de memória que está em uso pelo agente.
`MessagesInPerSec`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O número de mensagens recebidas por segundo do agente.
`NetworkRxDropped`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O número de pacotes de recebimento descartados.
`NetworkRxErrors`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O número de erros de recepção da rede para o agente.
`NetworkRxPackets`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O número de pacotes recebidos pelo agente.

Nome	Quando visível	Dimensõ	Descrição
`NetworkTx Dropped`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O número de pacotes de transmissão descartados.
`NetworkTx Errors`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O número de erros de transmissão da rede para o agente.
`NetworkTx Packets`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O número de pacotes transmitidos pelo agente.
`OfflinePa rtitionsCount`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster	Número total de partições que estão offline no cluster.
`PartitionCount`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O número total de partições de tópico por agente, incluindo réplicas.
`ProduceTo talTimeMsMean`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O tempo médio de produção em milissegundos.

Nome	Quando visível	Dimensõ	Descrição
`RequestBytesMean`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O número médio de bytes de solicitações do agente.
`RequestTime`	Após a limitação da solicitação ser aplicada.	Nome do cluster, ID do agente	O tempo médio gasto em milissegundos em threads de rede e de E/S do agente para processar solicitações.
`RootDiskUsed`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	A porcentagem do disco raiz usado pelo agente.
`SumOffsetLag`	Depois que o grupo de consumidores consome de um tópico.	Grupo de consumi res, tópico	O atraso de deslocamento agregado para todas as partições em um tópico.
`SwapFree`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O tamanho, em bytes, de memória de swap que está disponível para o agente.
`SwapUsed`	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O tamanho em bytes de memória de swap que está em uso para o agente.

Nome	Quando visível	Dimensõ	Descrição
TrafficShaping	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	Métricas de alto nível que indicam o número de pacotes modelados (descartados ou enfileirados) devido ao excesso de alocações de rede. É possível obter detalhes mais aprofundados com as métricas de PER_BROKER.
UnderMinIsrPartitionCount	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O número de partições em minIsr do agente.
UnderReplicatedPartitions	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	O número de partições sub-replicadas do agente.
ZooKeeperRequestLatencyMsMean	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	Para cluster ZooKeeper baseado. A latência média em milissegundos para ZooKeeper solicitações do Apache do broker.
ZooKeeperSessionState	Depois que o cluster passa para o estado ACTIVE.	Nome do cluster, ID do agente	Para cluster ZooKeeper baseado. Status da conexão da ZooKeeper sessão do broker, que pode ser um dos seguintes: NOT_CONNECTED: '0.0', ASSOCIATING: '0.1', CONNECTING: '0.5', CONNECTEDREADONLY: '0.8', CONNECTED: '1.0', CLOSED: '5.0', AUTH_FAILED: '10.0'.

Monitoramento no nível **PER_BROKER**

Ao definir o nível de monitoramento como PER_BROKER, você obtém as métricas descritas na tabela a seguir, além de todas as métricas de nível DEFAULT. Você paga pelas métricas na tabela a seguir, enquanto as métricas de nível DEFAULT continuam gratuitas. As métricas nesta tabela têm as seguintes dimensões: nome do cluster, ID do agente.

Métricas adicionais disponíveis a partir do nível de monitoramento **PER_BROKER**

Nome	Quando visível	Descrição
BwInAllowanceExceeded	Depois que o cluster passa para o estado ACTIVE.	Número de pacotes formados porque a largura de banda agregada de entrada excedeu o máximo para o agente.
BwOutAllowanceExceeded	Depois que o cluster passa para o estado ACTIVE.	Número de pacotes formados porque a largura de banda agregada de saída excedeu o máximo para o agente.
ConnTrackAllowanceExceeded	Depois que o cluster passa para o estado ACTIVE.	Número de pacotes formados porque o monitoramento de conexão excedeu o máximo para o agente. O monitoramento de conexão está relacionado a grupos de segurança que monitoram cada conexão estabelecida a fim de garantir que os pacotes de retorno sejam entregues conforme esperado.
ConnectionCloseRate	Depois que o cluster passa para o estado ACTIVE.	O número de conexões fechadas por segundo por receptor. Esse número é agregado por receptor e filtrado para os receptores do cliente.
ConnectionCreationRate	Depois que o cluster passa para o estado ACTIVE.	O número de novas conexões estabelecidas por segundo por receptor. Esse número é agregado por receptor e filtrado para os receptores do cliente.

Nome	Quando visível	Descrição
`CpuCreditUsage`	Depois que o cluster passa para o estado ACTIVE.	O número de créditos de CPU gastos pelo agente. A falta de saldo de créditos de CPU pode afetar negativamente o desempenho do cluster. Você pode adotar medidas para reduzir a carga da CPU. Por exemplo, você pode reduzir o número de solicitações de clientes ou atualizar o tipo de agente para um tipo de agente M5.
`FetchConsumerLocalTimeMsMean`	Depois de haver um produtor/consumidor.	O tempo médio em milissegundos que a solicitação do consumidor é processada no líder.
`FetchConsumerRequestQueueTimeMsMean`	Depois de haver um produtor/consumidor.	O tempo médio em milissegundos que a solicitação do consumidor aguarda na fila de solicitações.
`FetchConsumerResponseQueueTimeMsMean`	Depois de haver um produtor/consumidor.	O tempo médio em milissegundos que a solicitação do consumidor aguarda na fila de resposta.
`FetchConsumerResponseSendTimeMsMean`	Depois de haver um produtor/consumidor.	O tempo médio, em milissegundos, para que o consumidor envie uma resposta.
`FetchConsumerTotalTimeMsMean`	Depois de haver um produtor/consumidor.	O tempo total médio em milissegundos que os consumidores gastam obtendo dados do agente.
`FetchFollowerLocalTimeMsMean`	Depois de haver um produtor/consumidor.	O tempo médio em milissegundos que a solicitação do seguidor é processada no líder.

Nome	Quando visível	Descrição
FetchFollowerRequestQueueTimeMsMean	Depois de haver um produtor/consumidor.	O tempo médio em milissegundos que a solicitação de seguidor aguarda na fila de solicitações.
FetchFollowerResponseQueueTimeMsMean	Depois de haver um produtor/consumidor.	O tempo médio em milissegundos que a solicitação de seguidor aguarda na fila de resposta.
FetchFollowerResponseSendTimeMsMean	Depois de haver um produtor/consumidor.	O tempo médio em milissegundos para o seguidor enviar uma resposta.
FetchFollowerTotalTimeMsMean	Depois de haver um produtor/consumidor.	O tempo total médio em milissegundos que os seguidores gastam obtendo e dados do agente.
FetchMessageConversionsPerSec	Depois de criar um tópico.	O número de conversões de mensagens de busca por segundo do agente.
FetchThrottleByteRate	Depois que a limitação da largura de banda é aplicada.	O número de bytes limitados por segundo.
FetchThrottleQueueSize	Depois que a limitação da largura de banda é aplicada.	O número de mensagens na fila de limitação.
FetchThrottleTime	Depois que a limitação da largura de banda é aplicada.	O tempo médio de limitações de busca em milissegundos.
IAMNumberOfConnectionRequests	Depois que o cluster passa para o estado ACTIVE.	O número de solicitações de autenticação do IAM por segundo.

Nome	Quando visível	Descrição
IAMTooManyConnections	Depois que o cluster passa para o estado ACTIVE.	O número de conexões tentadas além de 100. 0 significa que o número de conexões está dentro do limite. Se >0, o limite do acelerador está sendo excedido e você precisa reduzir o número de conexões.
NetworkProcessorAvgIdlePercent	Depois que o cluster passa para o estado ACTIVE.	A porcentagem média do tempo em que os processadores de rede estão ociosos.
PpsAllowanceExceeded	Depois que o cluster passa para o estado ACTIVE.	O número de pacotes formados porque o PPS bidirecional excedeu o máximo para o agente.
ProduceLocalTimeMsMean	Depois que o cluster passa para o estado ACTIVE.	O tempo médio em milissegundos que a solicitação leva para ser processada no líder.
ProduceMessageConversionsPerSec	Depois de criar um tópico.	O número de conversões de mensagens de produção por segundo do agente.
ProduceMessageConversionsTimeMsMean	Depois que o cluster passa para o estado ACTIVE.	O tempo médio em milissegundos gasto em conversões de formato de mensagem.
ProduceRequestQueueTimeMsMean	Depois que o cluster passa para o estado ACTIVE.	O tempo médio em milissegundos que as mensagens de solicitação gastam na fila.
ProduceResponseQueueTimeMsMean	Depois que o cluster passa para o estado ACTIVE.	O tempo médio em milissegundos que as mensagens de resposta gastam na fila.

Nome	Quando visível	Descrição
`ProduceResponseSendTimeMsMean`	Depois que o cluster passa para o estado ACTIVE.	O tempo médio em milissegundos gasto no envio de mensagens de resposta.
`ProduceThrottleByteRate`	Depois que a limitação da largura de banda é aplicada.	O número de bytes limitados por segundo.
`ProduceThrottleQueueSize`	Depois que a limitação da largura de banda é aplicada.	O número de mensagens na fila de limitação.
`ProduceThrottleTime`	Depois que a limitação da largura de banda é aplicada.	O tempo médio de limitação da produção em milissegundos.
`ProduceTotalTimeMsMean`	Depois que o cluster passa para o estado ACTIVE.	O tempo médio de produção em milissegundos.
`RemoteFetchBytesPerSec (RemoteBytesInPerSec in v2.8.2.tiered)`	Depois de haver um produtor/consumidor.	O número total de bytes transferidos do armazenamento em camadas como resposta às buscas do consumidor. Essa métrica inclui todas as partições de tópicos que contribuem para o tráfego de transferência de dados downstream. Categoria: taxas de tráfego e erro. Essa é uma métrica KIP-405.

Nome	Quando visível	Descrição
`RemoteCopyBytesPerSec (RemoteBytesOutPerSec in v2.8.2.tiered)`	Depois de haver um produtor/consumidor.	O número total de bytes transferidos para o armazenamento em camadas, incluindo dados de segmentos de log, índices e outros arquivos auxiliares. Essa métrica inclui todas as partições de tópicos que contribuem para o tráfego de transferência de dados upstream. Categoria: taxas de tráfego e erro. Essa é uma métrica KIP-405.
`RemoteLogManagerTasksAvgIdlePercent`	Depois que o cluster passa para o estado ACTIVE.	O percentual médio do tempo que o gerenciador remoto de logs ficou ocioso. O gerenciador remoto de logs transfere dados do agente para o armazenamento em camadas. Categoria: atividade interna. Essa é uma métrica KIP-405.
`RemoteLogReaderAvgIdlePercent`	Depois que o cluster passa para o estado ACTIVE.	O percentual médio do tempo que o leitor remoto de logs ficou ocioso. O leitor remoto de logs transfere dados do armazenamento remoto para o agente em resposta às buscas do consumidor. Categoria: atividade interna. Essa é uma métrica KIP-405.
`RemoteLogReaderTaskQueueSize`	Depois que o cluster passa para o estado ACTIVE.	O número de tarefas responsáveis por leituras do armazenamento em camadas que estão aguardando para serem agendadas. Categoria: atividade interna. Essa é uma métrica KIP-405.

Nome	Quando visível	Descrição
`RemoteFetchErrorsPerSec (RemoteReadErrorPerSec in v2.8.2.tiered)`	Depois que o cluster passa para o estado ACTIVE.	A taxa total de erros em resposta às solicitações de leitura que o agente especificado enviou ao armazenamento em camadas para recuperar dados em resposta às buscas do consumidor. Essa métrica inclui todas as partições de tópicos que contribuem para o tráfego de transferência de dados downstream. Categoria: taxas de tráfego e erro. Essa é uma métrica KIP-405.
`RemoteFetchRequestsPerSec (RemoteReadRequestsPerSec in v2.8.2.tiered)`	Depois que o cluster passa para o estado ACTIVE.	O número total de solicitações de leitura que o agente especificado enviou ao armazenamento em camadas para recuperar dados em resposta às buscas do consumidor. Essa métrica inclui todas as partições de tópicos que contribuem para o tráfego de transferência de dados downstream. Categoria: taxas de tráfego e erro. Essa é uma métrica KIP-405.
`RemoteCopyErrorsPerSec (RemoteWriteErrorPerSec in v2.8.2.tiered)`	Depois que o cluster passa para o estado ACTIVE.	A taxa total de erros em resposta às solicitações de gravação que o agente especificado enviou ao armazenamento em camadas para transferir dados upstream. Essa métrica inclui todas as partições de tópicos que contribuem para o tráfego de transferência de dados upstream. Categoria: taxas de tráfego e erro. Essa é uma métrica KIP-405.

Nome	Quando visível	Descrição
`ReplicationBytesInPerSec`	Depois de criar um tópico.	O número de bytes por segundo recebidos dos outros agentes.
`ReplicationBytesOutPerSec`	Depois de criar um tópico.	O número de bytes por segundo enviados para outros agentes.
`RequestExemptFromThrottleTime`	Após a limitação da solicitação ser aplicada.	O tempo médio gasto em milissegundos em threads de rede e de E/S do agente para processar solicitações isentas de limitação.
`RequestHandlerAvgIdlePercent`	Depois que o cluster passa para o estado ACTIVE.	A porcentagem média do tempo em que os threads do manipulador de solicitações estão ociosos.
`RequestThrottleQueueSize`	Após a limitação da solicitação ser aplicada.	O número de mensagens na fila de limitação.
`RequestThrottleTime`	Após a limitação da solicitação ser aplicada.	O tempo médio da limitação de solicitações em milissegundos.
`TcpConnections`	Depois que o cluster passa para o estado ACTIVE.	Mostra o número de segmentos TCP de entrada e saída com o sinalizador SYN definido.

Nome	Quando visível	Descrição
`RemoteCopyLagBytes (TotalTierBytesLag in v2.8.2.tiered)`	Depois de criar um tópico.	O número total de bytes dos dados que são elegíveis para classificação hierárquica no agente, mas que ainda não foram transferidos para o armazenamento em camadas. Essas métricas mostram a eficiência da transferência de dados upstream. Conforme o atraso aumenta, a quantidade de dados que não persiste no armazenamento em camadas aumenta. Categoria: atraso de arquivamento. Essa não é uma métrica KIP-405.
`TrafficBytes`	Depois que o cluster passa para o estado ACTIVE.	Mostra o tráfego de rede em bytes gerais entre clientes (produtores e consumidores) e agentes. O tráfego entre agentes não é relatado.
`VolumeQueueLength`	Depois que o cluster passa para o estado ACTIVE.	O número de solicitações de operação de leitura e gravação aguardando conclusão em um período especific ado.
`VolumeReadBytes`	Depois que o cluster passa para o estado ACTIVE.	O número de bytes lidos durante um período especificado.
`VolumeReadOps`	Depois que o cluster passa para o estado ACTIVE.	O número de operações de leitura durante um período especificado.
`VolumeTotalReadTime`	Depois que o cluster passa para o estado ACTIVE.	O número total de segundos gastos por todas as operações de leitura que foram concluídas durante um período especificado.

Nome	Quando visível	Descrição
`VolumeTotalWriteTime`	Depois que o cluster passa para o estado ACTIVE.	O número total de segundos gastos por todas as operações de gravação que foram concluídas durante um período especificado.
`VolumeWriteBytes`	Depois que o cluster passa para o estado ACTIVE.	O número de bytes gravados durante um período especificado.
`VolumeWriteOps`	Depois que o cluster passa para o estado ACTIVE.	O número de operações de gravação durante um período especificado.

Monitoramento no nível **`PER_TOPIC_PER_BROKER`**

Ao definir o nível de monitoramento como `PER_TOPIC_PER_BROKER`, você obtém as métricas descritas na tabela a seguir, além de todas as métricas dos níveis `PER_BROKER` e DEFAULT. Somente as métricas de nível `DEFAULT` são gratuitas. As métricas nesta tabela têm as seguintes dimensões: nome do cluster, ID do agente, tópico.

> **Important**
>
> Para um cluster do Amazon MSK que use o Apache Kafka 2.4.1 ou uma versão mais recente, as métricas na tabela a seguir só aparecerão depois que os valores ficarem diferentes de zero pela primeira vez. Por exemplo, para ver `BytesInPerSec`, um ou mais produtores devem primeiro enviar dados para o cluster.

Métricas adicionais disponíveis a partir do nível de monitoramento **`PER_TOPIC_PER_BROKER`**

Nome	Quando visível	Descrição
`FetchMessageConversionsPerSec`	Depois de criar um tópico.	O número de mensagens obtidas convertidas por segundo.

Nome	Quando visível	Descrição
`MessagesInPerSec`	Depois de criar um tópico.	O número de mensagens recebidas por segundo.
`ProduceMessageConversionsPerSec`	Depois de criar um tópico.	O número de conversões por segundo de mensagens produzidas.
`RemoteFetchBytesPerSec (RemoteBytesInPerSec in v2.8.2.tiered)`	Após criar um tópico e o tópico estiver produzindo/consumindo.	O número de bytes transferidos do armazenamento em camadas em resposta às buscas do consumidor para o tópico e o agente especificados. Essa métrica inclui todas as partições do tópico que contribuem para o tráfego de transferência de dados downstream no agente especificado. Categoria: taxas de tráfego e erro. Essa é uma métrica KIP-405.
`RemoteCopyBytesPerSec (RemoteBytesOutPerSec in v2.8.2.tiered)`	Após criar um tópico e o tópico estiver produzindo/consumindo.	O número de bytes transferidos para o armazenamento em camadas, para o tópico e o agente especificados. Essa métrica inclui todas as partições do tópico que contribuem para o tráfego de transferência de dados upstream no agente especificado. Categoria: taxas de tráfego e erro. Essa é uma métrica KIP-405.
`RemoteFetchErrorsPerSec (RemoteReadErrorPerSec in v2.8.2.tiered)`	Após criar um tópico e o tópico estiver produzindo/consumindo.	A taxa de erros em resposta às solicitações de leitura que o agente especificado envia ao armazenamento em camadas para recuperar dados em resposta às buscas do consumidor sobre o tópico especificado. Essa métrica inclui todas as partições do tópico que contribuem para o tráfego de transferência de dados downstream no agente especificado. Categoria: taxas de tráfego e erro. Essa é uma métrica KIP-405.

Nome	Quando visível	Descrição
`RemoteFetchRequestsPerSec (RemoteReadRequestsPerSec in v2.8.2.tiered)`	Após criar um tópico e o tópico estiver produzindo/consumindo.	O número de solicitações de leitura que o agente especificado envia ao armazenamento em camadas para recuperar dados em resposta às buscas do consumidor sobre o tópico especificado. Essa métrica inclui todas as partições do tópico que contribuem para o tráfego de transferência de dados downstream no agente especificado. Categoria: taxas de tráfego e erro. Essa é uma métrica KIP-405.
`RemoteCopyErrorsPerSec (RemoteWriteErrorPerSec in v2.8.2.tiered)`	Após criar um tópico e o tópico estiver produzindo/consumindo.	A taxa de erros em resposta às solicitações de gravação que o agente especificado envia ao armazenamento em camadas para transferir dados upstream. Essa métrica inclui todas as partições do tópico que contribuem para o tráfego de transferência de dados upstream no agente especificado. Categoria: taxas de tráfego e erro. Essa é uma métrica KIP-405.

Monitoramento no nível **`PER_TOPIC_PER_PARTITION`**

Ao definir o nível de monitoramento como `PER_TOPIC_PER_PARTITION`, você obtém as métricas descritas na tabela a seguir, além de todas as métricas dos níveis `PER_TOPIC_PER_BROKER`, `PER_BROKER` e DEFAULT. Somente as métricas de nível `DEFAULT` são gratuitas. As métricas nesta tabela têm as seguintes dimensões: grupo de consumidores, tópico, partição.

Métricas adicionais disponíveis a partir do nível de monitoramento **`PER_TOPIC_PER_PARTITION`**

Nome	Quando visível	Descrição
`EstimatedTimeLag`	Depois que o grupo de consumidores consome de um tópico.	Estimativa de tempo (em segundos) para drenar o atraso no deslocamento da partição.

Nome	Quando visível	Descrição
`OffsetLag`	Depois que o grupo de consumidores consome de um tópico.	Atraso do consumidor no nível de partição em número de deslocamentos.

Visualizando métricas do Amazon MSK usando CloudWatch

Você pode monitorar as métricas do Amazon MSK usando o CloudWatch console, a linha de comando ou a CloudWatch API. Os procedimentos a seguir mostram como acessar as métricas usando os seguintes métodos:

Para acessar métricas usando o CloudWatch console

Faça login no AWS Management Console e abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.

1. No painel de navegação, selecione Métricas.
2. Escolha a guia Todas as métricas e escolha AWS/Kafka.
3. Para visualizar métricas em nível de tópico, escolha Topic, Broker ID, Cluster Name (Tópico, ID do agente, nome do cluster); para métricas em nível de agente, escolha Broker ID, Cluster Name (ID do agente, nome do cluster) e, para métricas em nível de cluster, escolha Cluster Name (Nome do cluster).
4. (Opcional) No painel gráfico, selecione uma estatística e um período de tempo e, em seguida, crie um CloudWatch alarme usando essas configurações.

Para acessar métricas usando o AWS CLI

Use os comandos list-metrics e get-metric-statistics.

Para acessar métricas usando a CloudWatch CLI

Use os comandos mon-list-metrics e mon-get-stats.

Para acessar métricas usando a CloudWatch API

Use as operações ListMetricse GetMetricEstatísticas.

Monitoramento de atraso do consumidor

O monitoramento do atraso do consumidor permite identificar consumidores lentos ou presos que não estão acompanhando os dados mais recentes disponíveis em um tópico. Quando necessário, você poderá adotar medidas corretivas, como escalar ou reinicializar esses consumidores. Para monitorar o atraso do consumidor, você pode usar a Amazon CloudWatch ou abrir o monitoramento com o Prometheus.

As métricas de atraso do consumidor quantificam a diferença entre os dados mais recentes gravados em seus tópicos e os dados lidos por suas aplicações. O Amazon MSK fornece as seguintes métricas de atraso do consumidor, que você pode obter por meio da Amazon CloudWatch ou por meio do monitoramento aberto com o Prometheus:,,, e. `EstimatedMaxTimeLag` `EstimatedTimeLag` `MaxOffsetLag` `OffsetLag` `SumOffsetLag` Para obter informações sobre essas métricas, consulte the section called “Métricas do Amazon MSK para monitoramento com CloudWatch”.

> **Note**
>
> As métricas de atraso do consumidor são visíveis somente para grupos de consumidores em um estado ESTÁVEL. Um grupo de consumidores fica ESTÁVEL após a conclusão bem-sucedida do rebalanceamento, garantindo que as partições sejam distribuídas uniformemente entre os consumidores.

O Amazon MSK é compatível com métricas de atraso do consumidor para clusters com o Apache Kafka 2.2.1 ou versões posteriores.

Monitoramento aberto com o Prometheus

É possível monitorar o cluster do MSK com o Prometheus, um sistema de código aberto para o monitoramento de dados de métrica de séries temporais. Você pode publicar esses dados no Amazon Managed Service for Prometheus usando o recurso de gravação remota do Prometheus. Também é possível usar ferramentas compatíveis com as métricas ou as ferramentas formatadas do Prometheus que se integram ao Monitoramento aberto do Amazon MSK, como a lógica do Datadog, Lenses, New Relic e Sumo. O monitoramento aberto está disponível gratuitamente, mas cobranças

são aplicáveis à transferência de dados entre zonas de disponibilidade. Para obter informações sobre o Prometheus, consulte a documentação do Prometheus.

Como criar um cluster do Amazon MSK com um monitoramento aberto habilitado

Usando o AWS Management Console

1. Faça login no AWS Management Console e abra o console do Amazon MSK em https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Na seção Monitoring (Monitoramento), marque a caixa de seleção ao lado de Enable open monitoring with Prometheus (Habilitar o monitoramento aberto com o Prometheus).
3. Forneça as informações obrigatórias em todas as seções da página e revise todas as opções disponíveis.
4. Selecione Criar cluster.

Usando o AWS CLI

- Invoque o comando create-cluster e especifique a opção `open-monitoring`. Habilite o `JmxExporter`, o `NodeExporter` ou ambos. Se você especificar o `open-monitoring`, os dois exportadores não poderão ser desabilitados ao mesmo tempo.

Uso da API

- Invoque a CreateClusteroperação e especifique`OpenMonitoring`. Habilite o `jmxExporter`, o `nodeExporter` ou ambos. Se você especificar o `OpenMonitoring`, os dois exportadores não poderão ser desabilitados ao mesmo tempo.

Como habilitar o monitoramento aberto para um cluster existente do Amazon MSK

Para habilitar o monitoramento aberto, verifique se o cluster está no estado `ACTIVE`.

Usando o AWS Management Console

1. Faça login no AWS Management Console e abra o console do Amazon MSK em https://console.aws.amazon.com/msk/home?region=us-east-1#/home/.
2. Escolha o nome do cluster que deseja atualizar. Você será redirecionado para uma página com os detalhes do cluster.
3. Na guia Propriedades, role para baixo para encontrar a seção Monitoramento.
4. Selecione a opção Editar.
5. Marque a caixa de seleção ao lado de Enable open monitoring with Prometheus (Habilitar o monitoramento aberto com o Prometheus).
6. Escolha Salvar alterações.

Usando o AWS CLI

- Invoque o comando update-monitoring e especifique a opção `open-monitoring`. Habilite o `JmxExporter`, o `NodeExporter` ou ambos. Se você especificar o `open-monitoring`, os dois exportadores não poderão ser desabilitados ao mesmo tempo.

Uso da API

- Invoque a UpdateMonitoringoperação e especifique`OpenMonitoring`. Habilite o `jmxExporter`, o `nodeExporter` ou ambos. Se você especificar o `OpenMonitoring`, os dois exportadores não poderão ser desabilitados ao mesmo tempo.

Como configurar um host do Prometheus em uma instância do Amazon EC2

1. Baixe o servidor do Prometheus em https://prometheus.io/download/#prometheus para sua instância do Amazon EC2.
2. Extraia o arquivo obtido por download para um diretório e acesse esse diretório.
3. Crie um arquivo com o seguinte conteúdo e nomeie-o como `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. Use a ListNodesoperação para obter uma lista dos corretores do seu cluster.
5. Crie um arquivo denominado `targets.json` com a seguinte JSON: Substitua *broker_dns_1*, *broker_dns_2* e o restante dos nomes do DNS com os nomes do DNS obtidos para os agentes na etapa anterior. Inclua todos os agentes que você obteve na etapa anterior. O Amazon MSK usa a porta 11001 para o JMX Exporter e a porta 11002 para o 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**
>
> Para extrair métricas JMX dos controladores Kraft, adicione nomes DNS do controlador como destinos no arquivo JSON. Por exemplo:`controller_dns_1:11001`, `controller_dns_1` substituindo pelo nome DNS real do controlador.

6. Para iniciar o servidor do Prometheus na instância do Amazon EC2, execute o seguinte comando no diretório no qual extraiu os arquivos do Prometheus e salvou `prometheus.yml` e `targets.json`.

```
./prometheus
```

7. Localize o endereço IP IPv4 público da instância do Amazon EC2 na qual executou o Prometheus na etapa anterior. Esse endereço IP público será necessário na próxima etapa.
8. Para acessar IU Web do Prometheus, abra um navegador capaz de acessar sua instância do Amazon EC2 e acesse *`Prometheus-Instance-Public-IP`*`:9090`, com *`Prometheus-Instance-Public-IP`* indicando o endereço IP público obtido na etapa anterior.

Métricas do Prometheus

Todas as métricas emitidas pelo Apache Kafka para o JMX são acessíveis ao usar o monitoramento aberto com o Prometheus. Para obter informações sobre as métricas do Apache Kafka, consulte Monitoring (Monitoramento) na documentação do Apache Kafka. Junto com as métricas do Apache Kafka, as métricas de atraso do consumidor também estão disponíveis na porta 11001 sob o nome `kafka.consumer.group:type=ConsumerLagMetrics` no JMX MBean. Você também pode usar o Prometheus Node Exporter para obter métricas de CPU e disco para seus agentes na porta 11002.

Como armazenar as métricas do Prometheus no Amazon Managed Service for Prometheus

O Amazon Managed Service for Prometheus é um serviço de monitoramento e emissão de alertas compatível com o Prometheus que você pode usar para monitorar os clusters do Amazon MSK. É um serviço totalmente gerenciado que dimensiona automaticamente a ingestão, o armazenamento, a consulta e o alerta de métricas. Ele também se integra aos serviços de AWS segurança para

oferecer acesso rápido e seguro aos seus dados. É possível usar a linguagem de consulta PromQL de código aberto para consultar suas métricas e emitir alertas sobre elas.

Para obter mais informações, consulte Conceitos básicos do Amazon Managed Service for Prometheus.

Alertas de capacidade de armazenamento do Amazon MSK

Nos clusters provisionados pelo Amazon MSK, você escolhe a capacidade de armazenamento principal do cluster. O esgotamento da capacidade de armazenamento de um agente no cluster provisionado pode afetar a capacidade do cluster de produzir e consumir dados, resultando em um tempo de inatividade dispendioso. O Amazon MSK oferece CloudWatch métricas para ajudar você a monitorar a capacidade de armazenamento do seu cluster. No entanto, para facilitar a detecção e a resolução de problemas de capacidade de armazenamento, o Amazon MSK envia automaticamente alertas dinâmicos de capacidade de armazenamento do cluster. Os alertas de capacidade de armazenamento incluem recomendações para etapas de curto e longo prazo para o gerenciamento da capacidade de armazenamento do cluster. No console do Amazon MSK, você pode usar links rápidos nos alertas para executar imediatamente as ações recomendadas.

Há dois tipos de alertas de capacidade de armazenamento do MSK: proativos e corretivos.

- Alertas proativos (“Ação necessária”) de capacidade de armazenamento avisam você sobre possíveis problemas de armazenamento no cluster. Quando um agente em um cluster do MSK usar mais de 60% ou 80% da capacidade de armazenamento em disco, você receberá alertas proativos para o agente afetado.
- Os alertas de capacidade de armazenamento corretivos (“Ação crítica necessária”) exigem que você tome medidas corretivas para corrigir um problema crítico no cluster quando um dos agentes do cluster do MSK fica sem capacidade de armazenamento em disco.

O Amazon MSK envia automaticamente esses alertas para o console do Amazon MSK, AWS Health Dashboard EventBridge, Amazon e contatos de e-mail da sua AWS conta. Você também pode configurar EventBridge a Amazon para entregar esses alertas ao Slack ou a ferramentas como New Relic e Datadog.

Os alertas de capacidade de armazenamento são habilitados por padrão para todos os clusters provisionados pelo MSK e não podem ser desativados. Esse recurso é compatível em todas as regiões em que o MSK está disponível.

Monitorar alertas de capacidade de armazenamento do Amazon MSK

Você pode verificar os alertas de capacidade de armazenamento de várias maneiras:

- Vá para o console do Amazon MSK. Os alertas de capacidade de armazenamento são exibidos no painel de alertas do cluster por 90 dias. Os alertas contêm recomendações e ações de link com um único clique para resolver problemas de capacidade de armazenamento em disco.
- Use ListClustersAPIs ListClustersV2 ou DescribeClusterV2 para visualizar `CustomerActionStatus` todos os alertas de um cluster. DescribeCluster
- Acesse o AWS Health Dashboard para ver os alertas do MSK e de outros AWS serviços.
- Configure a AWS Health API e EventBridge a Amazon para encaminhar notificações de alerta para plataformas de terceiros NewRelic, como Datadog e Slack.

Usando o LinkedIn Cruise Control para Apache Kafka com o Amazon MSK

Você pode usar LinkedIn o Cruise Control para reequilibrar seu cluster Amazon MSK, detectar e corrigir anomalias e monitorar o estado e a integridade do cluster.

Para baixar e compilar o Cruise Control

1. Crie uma instância do Amazon EC2 na mesma Amazon VPC do cluster do Amazon MSK.
2. Instale o Prometheus na instância do Amazon EC2 que você criou na etapa anterior. Anote o IP privado e a porta. O número padrão da porta é 9090. Para obter informações sobre como configurar o Prometheus de modo a agregar métricas de seu cluster, consulte the section called "Monitoramento aberto com o Prometheus".
3. Faça o download do Cruise Control na instância do Amazon EC2. (Como alternativa, se preferir você pode usar uma instância separada do Amazon EC2 para o Cruise Control.) Para um cluster que tenha o Apache Kafka versão 2.4.*, use a versão 2.4.* mais recente do Cruise Control. Se seu cluster tiver uma versão do Apache Kafka anterior à 2.4.*, use a versão mais recente do 2.0.* Cruise Control.
4. Descompacte o arquivo do Cruise Control e acesse a pasta descompactada.
5. Execute o comando a seguir para instalar o git.

```
sudo yum -y install git
```

6. Execute o comando a seguir para inicializar o repositório local. Substitua *Your-Cruise-Control-Folder* pelo nome da sua pasta atual (a pasta que você obteve ao descompactar o download do Cruise Control).

```
git init && git add . && git commit -m "Init local repo." && git tag -a Your-Cruise-Control-Folder -m "Init local version."
```

7. Execute o seguinte comando para compilar o código-fonte.

```
./gradlew jar copyDependantLibs
```

Para configurar e executar o Cruise Control

1. Faça as seguintes atualizações no arquivo `config/cruisecontrol.properties`. Substitua o exemplo de servidores bootstrap e a string bootstrap-brokers pelos valores do seu cluster. Para obter essas strings para seu cluster, você pode ver os detalhes do cluster no console. Como alternativa, você pode usar as operações GetBootstrapBrokerse da DescribeClusterAPI ou seus equivalentes de CLI.

```
# 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. Edite o arquivo `config/capacityCores.json` para especificar o tamanho correto do disco, os núcleos da CPU e os limites de entrada/saída da rede. Você pode usar a operação da DescribeClusterAPI (ou seu equivalente na CLI) para obter o tamanho do disco. Para núcleos de CPU e limites de entrada/saída de rede, consulte Tipos de instância do 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. Opcionalmente, você pode instalar a interface do usuário do Cruise Control. Para baixá-la, acesse Como configurar o frontend do Cruise Control.
4. Execute o comando a seguir para iniciar o Cruise Control. Considere usar uma ferramenta como `screen` ou `tmux` para manter uma sessão de longa duração aberta.

```
<path-to-your-kafka-installation>/bin/kafka-cruise-control-start.sh config/
cruisecontrol.properties 9091
```

5. Use as APIs do Cruise Control ou a interface do usuário para garantir que o Cruise Control tenha os dados de carga do cluster e que esteja fazendo sugestões de rebalanceamento. A obtenção de uma janela de métricas válida pode levar alguns minutos.

Modelo de implantação automatizada do Cruise Control para Amazon MSK

Você também pode usar esse CloudFormation modelo para implantar facilmente o Cruise Control e o Prometheus para obter informações mais detalhadas sobre o desempenho do seu cluster Amazon MSK e otimizar a utilização de recursos.

Principais recursos:

- Provisionamento automatizado de uma instância do Amazon EC2 com Cruise Control e Prometheus pré-configurados.
- Support para cluster provisionado do Amazon MSK.
- Autenticação PlainText flexível com IAM.
- Sem dependência do Zookeeper para o Cruise Control.
- Personalize facilmente os alvos do Prometheus, as configurações de capacidade do Cruise Control e outras configurações fornecendo seus próprios arquivos de configuração armazenados em um bucket do Amazon S3.

Cota do Amazon MSK

Sua AWS conta tem cotas padrão para o Amazon MSK. Salvo indicação em contrário, cada cota por conta é específica da região em sua conta. AWS

Cota do Amazon MSK

- Até 90 corretores por conta. 30 corretores por cluster de ZooKeeper modo. 60 corretores por cluster de modo Kraft. Para solicitar uma cota maior, acesse o Support Center do AWS console e crie um caso de suporte.
- Um mínimo de 1 GiB de armazenamento por agente.
- Um máximo de 16.384 GiB de armazenamento por agente.
- Um cluster que use the section called “Controle de acesso do IAM” pode ter até 3.000 conexões TCP por agente a qualquer momento. Para aumentar esse limite, você pode ajustar a propriedade de `listener.name.client_iam_public.max.connections` configuração `listener.name.client_iam.max.connections` ou usando a AlterConfig API Kafka ou a `kafka-configs.sh` ferramenta. É importante observar que aumentar qualquer propriedade para um valor alto pode resultar em indisponibilidade.
- Limites nas conexões TCP. Com os picos de taxa de conexão ativados, o MSK permite 100 conexões por segundo. A exceção é o tipo de instância kafka.t3.small, que permite 4 conexões por segundo com picos de taxa de conexão ativados. Clusters mais antigos que não têm picos de taxa de conexão ativados terão o recurso ativado automaticamente quando o cluster for corrigido.

  Para processar novas tentativas em conexões com falha, você pode definir o parâmetro de configuração `reconnect.backoff.ms` no lado do cliente. Por exemplo, se você quiser que um cliente tente novamente as conexões após 1 segundo, defina `reconnect.backoff.ms` como 1.000. Para obter mais informações, consulte reconnect.backoff.ms na documentação do Apache Kafka.
- Até 100 configurações por conta. Para solicitar um ajuste de cota, acesse a Central de suporte no console da AWS e crie um caso de suporte.
- Máximo de 50 revisões por configuração.
- Para atualizar a configuração ou a versão do Apache Kafka de um cluster do MSK, primeiro certifique-se de que o número de partições por agente esteja abaixo dos limites descritos em the section called “ Dimensione seu cluster adequadamente: número de partições por agente”.

Cotas do replicador do MSK

- Máximo de 15 replicadores do MSK por conta.
- O MSK Replicator replica somente até 750 tópicos em ordem ordenada. Se você precisar replicar mais tópicos, recomendamos criar um replicador separado. Acesse o Support Center do AWS console e crie um caso de suporte se precisar de suporte para mais de 750 tópicos por replicador. Você pode monitorar o número de tópicos que estão sendo replicados usando a métrica TopicCount "".
- Um throughput máximo de entrada de 1 GB por segundo por replicador do MSK. Para solicitar uma cota maior, acesse o Support Center do AWS console e crie um caso de suporte.
- Tamanho do registro do MSK Replicator - Um tamanho máximo de registro de 10 MB (message.max.bytes). Para solicitar uma cota maior, acesse o Support Center do AWS console e crie um caso de suporte.

Cota do MSK Serverless

> **Note**
>
> Se você tiver algum problema com os limites de cota, entre em contato com o AWS Support criando um caso de suporte.

Salvo indicação em contrário, os limites são por cluster.

Dimensão	Quota	Resultado de violação de cota
Throughput máximo de entrada	200 MBps	Desaceleração com duração de controle de utilização em resposta
Throughput máximo de saída	400 MBps	Desaceleração com duração de controle de utilização em resposta
Duração máxima de retenção	Ilimitado	N/D

Dimensão	Quota	Resultado de violação de cota
Número máximo de conexões de cliente	3000	Fechamento da conexão
Máximo de tentativas de conexão	100 por segundo	Fechamento da conexão
Tamanho máximo de mensagem	8 MB	A solicitação falha com ErrorCode: INVALID_REQUEST
Taxa máxima de solicitação	15.000 por segundo	Desaceleração com duração de controle de utilização em resposta
Taxa máxima de solicitações de APIs de gerenciamento de tópico	2 por segundo	Desaceleração com duração de controle de utilização em resposta
Máximo de bytes de busca por solicitação	55 MB	A solicitação falha com ErrorCode: INVALID_REQUEST
Número máximo de grupos de consumidores	500	JoinGroup falha na solicitação
Número máximo de partições (líderes)	2.400 para tópicos não compactados, 120 para tópicos compactados. Para solicitar um ajuste de cota, acesse o Support Center do AWS console e crie um caso de suporte.	A solicitação falha com ErrorCode: INVALID_REQUEST
Taxa máxima de criação e exclusão de partições	250 em 5 minutos	A solicitação falha com ErrorCode: THROUGHPUT_QUOTA_EXCEEDED

Dimensão	Quota	Resultado de violação de cota
Throughput máximo de entrada por partição	5 MBps	Desaceleração com duração de controle de utilização em resposta
Throughput máximo de saída por partição	10 MBps	Desaceleração com duração de controle de utilização em resposta
Tamanho máximo da partição (para tópicos compactados)	250 GB	A solicitação falha com ErrorCode: THROUGHPUT_QUOTA_EXCEEDED
Número máximo de VPCs clientes por cluster com tecnologia sem servidor	5	
Número máximo de clusters com tecnologia sem servidor por conta	10. Para solicitar um ajuste de cota, acesse o Support Center do AWS console e crie um caso de suporte.	

Cota do MSK Connect

- Até 100 plug-ins personalizados.
- Até 100 configurações de operador.
- Até 60 operadores conectados. Se um conector estiver configurado com capacidade de ajuste de escala automático, o número máximo de operadores que o conector está configurado para ter é o número que o MSK Connect usa para calcular a cota da conta.
- Até 10 operadores por conector.

Para solicitar uma cota maior para o MSK Connect, acesse o AWS Support Center do console e crie um caso de suporte.

Recursos do Amazon MSK

Dependendo do contexto, o termo recursos tem dois significados no Amazon MSK. No contexto das APIs, um recurso é uma estrutura na qual você pode invocar uma operação. Para obter uma lista desses recursos e das operações que você pode invocar neles, consulte Recursos na Referência de API do Amazon MSK. No contexto do the section called “Controle de acesso do IAM”, um recurso é uma entidade à qual você pode permitir ou proibir o acesso, conforme definido na seção the section called “Recursos”.

Integrações do MSK

Esta seção fornece referências aos AWS recursos que se integram ao Amazon MSK.

Tópicos

Conector do Amazon Athena para o Amazon MSK

O conector do Amazon Athena para o Amazon MSK possibilita que o Amazon Athena execute consultas SQL em tópicos do Apache Kafka. Use esse conector para visualizar os tópicos e as mensagens do Apache Kafka no Athena como tabelas e linhas, respectivamente.

Para obter mais informações, consulte Conector do MSK para Amazon Athena no Guia do usuário do Amazon Athena.

Ingestão de dados de streaming do Amazon Redshift

A ingestão de streaming do Amazon Redshift oferece suporte ao Amazon MSK. O recurso de ingestão de streaming do Amazon Redshift fornece ingestão de dados com baixa latência e alta velocidade do Amazon MSK para uma visão materializada do Amazon Redshift. Como não precisa armazenar dados no Amazon S3, o Amazon Redshift pode ingerir dados de streaming com uma latência menor e com um custo de armazenamento reduzido. Você pode configurar a ingestão de streaming do Amazon Redshift em um cluster do Amazon Redshift usando instruções SQL para autenticar e se conectar a um tópico do Amazon MSK.

Para obter mais informações, consulte Ingestão de streaming no Guia do desenvolvedor de banco de dados do Amazon Redshift.

Firehose

O Amazon MSK se integra ao Firehose para fornecer uma solução sem servidor e sem código para entregar streams dos clusters do Apache Kafka para os data lakes do Amazon S3. O Firehose é

um serviço de streaming de extração, transformação e carregamento (ETL) que lê dados de seus tópicos do Amazon MSK Kafka, realiza transformações, como conversão para Parquet, e agrega e grava os dados no Amazon S3. Com alguns cliques no console, você pode configurar um stream do Firehose para ler um tópico do Kafka e entregá-lo em um local do S3. Não há código para escrever, aplicações de conectores nem recursos para provisionar. O Firehose é escalado automaticamente com base na quantidade de dados publicados no tópico do Kafka, e você paga apenas pelos bytes ingeridos do Kafka.

Veja mais informações sobre esse recurso nos itens abaixo.

- Escrevendo no Kinesis Data Firehose usando o Amazon MSK - Amazon Kinesis Data Firehose no Guia do desenvolvedor do Amazon Data Firehose
- Blog: Amazon MSK apresenta entrega gerenciada de dados do Apache Kafka para seu data lake
- Laboratório: Entrega para o Amazon S3 usando Firehose

Acessando o Amazon EventBridge Pipes por meio do console Amazon MSK

O Amazon EventBridge Pipes conecta as fontes aos alvos. Os tubos são destinados a point-to-point integrações entre fontes e alvos suportados, com suporte para transformações e enriquecimento avançados. EventBridge O Pipes fornece uma maneira altamente escalável de conectar seu cluster Amazon MSK a AWS serviços como Step Functions, Amazon SQS e API Gateway, bem como a aplicativos de software como serviço (SaaS) de terceiros, como o Salesforce.

Para configurar um pipe, você escolhe a origem, adiciona filtragem opcional, define o enriquecimento opcional e escolhe o destino para os dados do evento.

Na página de detalhes do cluster do Amazon MSK, você pode ver os pipes que usam esse cluster como origem. Nessa página, você também pode:

- Inicie o EventBridge console para ver os detalhes do tubo.
- Inicie o EventBridge console para criar um novo canal com o cluster como fonte.

Para obter mais informações sobre como configurar um cluster Amazon MSK como fonte de canal, consulte o cluster Amazon Managed Streaming for Apache Kafka como fonte no Guia do usuário da Amazon. EventBridge Para obter mais informações sobre EventBridge tubos em geral, consulte EventBridge Tubos.

Para acessar EventBridge canais para um determinado cluster Amazon MSK

1. Abra o Console do Amazon MSK e selecione Clusters.
2. Selecione um cluster.
3. Na página de detalhes do cluster, escolha a guia Integração.

   A guia Integração inclui uma lista de todos os pipes configurados para usar o cluster selecionado como origem, inclusive:

   - nome do pipe
   - status atual
   - destino do pipe
   - última modificação do pipe
4. Gerencie os pipes do seu cluster do Amazon MSK conforme desejado:

   Para acessar mais detalhes sobre um pipe

   - Escolha o pipe.

     Isso abre a página de detalhes do Pipe do EventBridge console.

   Para criar um pipe

   - Escolha Conectar cluster do Amazon MSK ao Pipe.

     Isso inicia a página Create pipe do EventBridge console, com o cluster Amazon MSK especificado como a origem do pipe. Para obter mais informações, consulte EventBridge Criação de um tubo no Guia EventBridge do usuário da Amazon.

   - Você também pode criar um canal para um cluster na página Clusters. Selecione o cluster e, no menu Ações, selecione Create EventBridge Pipe.

Versões do Apache Kafka

Ao criar um cluster do Amazon MSK, você especifica qual versão do Apache Kafka deseja que ele tenha. Também é possível atualizar a versão do Apache Kafka de um cluster existente. Os tópicos do capítulo ajudam você a entender os cronogramas de suporte à versão Kafka e as sugestões de melhores práticas.

Tópicos

Versões compatíveis do Apache Kafka

O Amazon Managed Streaming for Apache Kafka (Amazon MSK) é compatível com as seguintes versões do Apache Kafka e do Amazon MSK. A comunidade Apache Kafka fornece aproximadamente 12 meses de suporte para uma versão após sua data de lançamento. Para obter mais detalhes, consulte a política de EOL (fim da vida útil) do Apache Kafka.

Versões compatíveis do Apache Kafka

Versão Apache Kafka	Data de lançamento do MSK	Data de fim do suporte
1.1.1	--	2024-06-05
2.1.0	--	2024-06-05
2.2.1	31-07-2019	2024-06-08
2.3.1	19-12-2019	2024-06-08
2.4.1	02-04-2020	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

Versão Apache Kafka	Data de lançamento do MSK	Data de fim do suporte
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	28/10/2022	2024-09-11
2.8.2 em camadas	28/10/2022	A ser anunciado
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 (recomendado)	2023-09-26	--
3.6.0	2023-11-16	--
3.7.x	2024-05-29	--

Para obter mais informações sobre a política de suporte à versão do Amazon MSK, consulte Política de suporte à versão Amazon MSK.

Apache Kafka versão 3.7.x (com armazenamento em camadas pronto para produção)

O Apache Kafka versão 3.7.x no MSK inclui suporte para o Apache Kafka versão 3.7.0. Você pode criar clusters ou atualizar clusters existentes para usar a nova versão 3.7.x. Com essa mudança no nome da versão, você não precisa mais adotar versões mais recentes de correção de patches, como a 3.7.1, quando elas são lançadas pela comunidade Apache Kafka. O Amazon MSK atualizará automaticamente a versão 3.7.x para oferecer suporte às futuras versões de patch assim que elas estiverem disponíveis. Isso permite que você se beneficie da segurança e das correções de erros disponíveis nas versões de correção de patches sem acionar uma atualização de versão. Essas versões de correção de patches lançadas pelo Apache Kafka não quebram a compatibilidade de versões e você pode se beneficiar das novas versões de correção de patches sem se preocupar com erros de leitura ou gravação em seus aplicativos clientes. Certifique-se de que suas ferramentas de automação de infraestrutura, como CloudFormation, estejam atualizadas para considerar essa alteração na nomenclatura da versão.

O Amazon MSK agora oferece suporte ao modo Kraft (Apache Kafka Raft) no Apache Kafka versão 3.7.x. No Amazon MSK, assim como ZooKeeper nos nós, os controladores Kraft são incluídos sem custo adicional para você e não exigem configuração ou gerenciamento adicionais de sua parte. Agora você pode criar clusters no modo Kraft ou ZooKeeper no modo Apache Kafka versão 3.7.x. No modo Kraft, você pode adicionar até 60 corretores para hospedar mais partições por cluster, sem solicitar um aumento de limite, em comparação com a cota de 30 corretores em clusters baseados no Zookeeper. Para saber mais sobre o Kraft no MSK, consulte o modo Kraft.

A versão 3.7.x do Apache Kafka também inclui várias correções de erros e novos recursos que melhoram o desempenho. As principais melhorias incluem otimizações de descoberta de líderes para clientes e opções de otimização de descarga de segmentos de log. Para obter uma lista completa de melhorias e correções de erros, consulte as notas de lançamento do Apache Kafka para 3.7.0.

Apache Kafka versão 3.6.0 (com armazenamento em camadas pronto para produção)

Para obter informações sobre a versão 3.6.0 (com armazenamento em camadas pronto para produção) do Apache Kafka, consulte as notas de versão no site de downloads do Apache Kafka.

Para fins de estabilidade, o Amazon MSK continuará usando e gerenciando o Zookeeper para gerenciamento de quórum nesta versão.

Amazon MSK versão 3.5.1

O Amazon Managed Streaming for Apache Kafka (Amazon MSK) agora oferece suporte ao Apache Kafka versão 3.5.1 para clusters novos e existentes. O Apache Kafka 3.5.1 inclui várias correções de erros e novos recursos que melhoram o desempenho. Os principais recursos incluem a introdução de uma nova atribuição de partições com reconhecimento de rack para consumidores. O Amazon MSK continuará usando e gerenciando o Zookeeper para gerenciamento de quórum nesta versão. Para obter uma lista completa de melhorias e correções de erros, consulte as notas de lançamento do Apache Kafka para a versão 3.5.1.

Para obter informações sobre a versão 3.5.1 do Apache Kafka, consulte as notas de versão no site de downloads do Apache Kafka.

Amazon MSK versão 3.4.0

O Amazon Managed Streaming for Apache Kafka (Amazon MSK) agora oferece suporte ao Apache Kafka versão 3.4.0 para clusters novos e existentes. O Apache Kafka 3.4.0 inclui várias correções de erros e novos recursos que melhoram o desempenho. Os principais recursos incluem uma correção para melhorar a estabilidade da busca na réplica mais próxima. O Amazon MSK continuará usando e gerenciando o Zookeeper para gerenciamento de quórum nesta versão. Para obter uma lista completa de melhorias e correções de erros, consulte as notas de lançamento do Apache Kafka para 3.4.0.

Para obter informações sobre a versão 3.4.0 do Apache Kafka, consulte as notas de versão no site de downloads do Apache Kafka.

Amazon MSK versão 3.3.2

O Amazon Managed Streaming for Apache Kafka (Amazon MSK) agora oferece suporte ao Apache Kafka versão 3.3.2 para clusters novos e existentes. O Apache Kafka 3.3.2 inclui várias correções de erros e novos recursos que melhoram o desempenho. Os principais recursos incluem uma correção para melhorar a estabilidade da busca na réplica mais próxima. O Amazon MSK continuará usando e gerenciando o Zookeeper para gerenciamento de quórum nesta versão. Para obter uma lista completa de melhorias e correções de erros, consulte as notas de lançamento do Apache Kafka para 3.3.2.

Para obter informações sobre a versão 3.3.2 do Apache Kafka, consulte as notas de versão no site de downloads do Apache Kafka.

Amazon MSK versão 3.3.1

O Amazon Managed Streaming for Apache Kafka (Amazon MSK) agora oferece suporte ao Apache Kafka versão 3.3.1 para clusters novos e existentes. O Apache Kafka 3.3.1 inclui várias correções de erros e novos recursos que melhoram o desempenho. Alguns dos principais recursos incluem aprimoramentos nas métricas e no particionador. Para fins de estabilidade, o Amazon MSK continuará usando e gerenciando o Zookeeper para gerenciamento de quórum nesta versão. Para obter uma lista completa de melhorias e correções de erros, consulte as notas de lançamento do Apache Kafka para 3.3.1.

Para obter informações sobre a versão 3.3.1 do Apache Kafka, consulte as notas de versão no site de downloads do Apache Kafka.

Amazon MSK versão 3.1.1

O Amazon Managed Streaming for Apache Kafka (Amazon MSK) agora oferece suporte ao Apache Kafka versão 3.1.1 e 3.2.0 para clusters novos e existentes. O Apache Kafka 3.1.1 e o Apache Kafka 3.2.0 incluem várias correções de erros e novos recursos que melhoram o desempenho. Alguns dos principais recursos incluem aprimoramentos nas métricas e o uso de IDs de tópicos. A MSK continuará usando e gerenciando o Zookeeper para gerenciamento de quórum nesta versão para fins de estabilidade. Para obter uma lista completa de melhorias e correções de erros, consulte as notas de lançamento do Apache Kafka para 3.1.1 e 3.2.0.

Para obter informações sobre as versões 3.1.1 e 3.2.0 do Apache Kafka, consulte suas notas de lançamento 3.2.0 e 3.1.1 no site de downloads do Apache Kafka.

Armazenamento em camadas do Amazon MSK versão 2.8.2.tiered

Essa versão é uma versão exclusiva do Amazon MSK do Apache Kafka versão 2.8.2, sendo compatível com clientes Apache Kafka de código aberto.

A versão 2.8.2.tiered contém a funcionalidade de armazenamento em camadas que é compatível com as APIs introduzidas no KIP-405 para Apache Kafka. Para obter mais informações sobre o recurso de armazenamento em camadas do Amazon MSK, consulte Armazenamento em camadas.

Apache Kafka versão 2.5.1

A versão 2.5.1 do Apache Kafka inclui várias correções de erros e novos recursos, incluindo criptografia em trânsito para clientes Apache e de administração. ZooKeeper O Amazon MSK fornece ZooKeeper endpoints TLS, que você pode consultar com a operação. DescribeCluster

A saída da DescribeClusteroperação inclui o `ZookeeperConnectStringTls` nó, que lista os endpoints do TLS zookeeper.

O exemplo a seguir mostra o nó `ZookeeperConnectStringTls` da resposta para a operação `DescribeCluster`:

```
"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"
```

Para obter informações sobre o uso da criptografia TLS com o zookeeper, consulte Usando a segurança TLS com o Apache ZooKeeper.

Para obter mais informações sobre a versão 2.5.1 do Apache Kafka, consulte as notas de versão no site de downloads do Apache Kafka.

Correção de bugs do Amazon MSK versão 2.4.1.1

Essa versão é uma versão de correção de bugs do Apache Kafka versão 2.4.1 exclusiva do Amazon MSK. Essa versão de correção de bugs contém uma correção para o KAFKA-9752, um problema raro que faz com que grupos de consumidores façam o rebalanceamento contínuo e permaneçam no estado `PreparingRebalance`. Esse problema afeta clusters que executam as versões 2.3.1 e 2.4.1. Essa versão contém uma correção produzida pela comunidade que está disponível na versão 2.5.0 do Apache Kafka.

> **Note**
>
> Os clusters do Amazon MSK que executam a versão 2.4.1.1 são compatíveis com qualquer cliente Apache Kafka compatível com o Apache Kafka versão 2.4.1.

Recomendamos que você use a correção de bugs do MSK versão 2.4.1.1 para novos clusters do Amazon MSK se preferir usar o Apache Kafka 2.4.1. É possível atualizar os clusters existentes que executam o Apache Kafka versão 2.4.1 para essa versão a fim de incorporar essa correção. Para obter informações sobre como atualizar um cluster existente, consulte Atualizar a versão do Apache Kafka.

Para contornar esse problema sem atualizar o cluster para a versão 2.4.1.1, consulte a seção Grupo de consumidores preso no estado `PreparingRebalance` do guia Solução de problemas no cluster do Amazon MSK.

Apache Kafka versão 2.4.1 (use 2.4.1.1 alternativamente)

> **Note**
>
> Você não pode mais criar um cluster do MSK com o Apache Kafka versão 2.4.1. Em vez disso, você pode usar a versão Correção de bugs do Amazon MSK versão 2.4.1.1 com clientes compatíveis com o Apache Kafka versão 2.4.1. E se você já tiver um cluster do MSK com o Apache Kafka versão 2.4.1, recomendamos que você o atualize para usar o Apache Kafka versão 2.4.1.1.

O KIP-392 é uma das principais propostas de melhoria do Kafka incluídas na versão 2.4.1 do Apache Kafka. Essa melhoria permite que os consumidores busquem a partir da réplica mais próxima. Para usar esse recurso, defina `client.rack` nas propriedades do consumidor como o ID da zona de disponibilidade do consumidor. Um exemplo de ID AZ é `use1-az1`. O Amazon MSK define `broker.rack` como os IDs das zonas de disponibilidade dos agentes. Também é necessário definir a propriedade de configuração `replica.selector.class` como `org.apache.kafka.common.replica.RackAwareReplicaSelector`, que é uma implementação de reconhecimento de rack fornecida pelo Apache Kafka.

Quando você usa esta versão do Apache Kafka, as métricas no nível de monitoramento `PER_TOPIC_PER_BROKER` aparecem somente após os valores se tornarem diferentes de zero pela primeira vez. Para obter mais informações sobre isso, consulte the section called “Monitoramento no nível `PER_TOPIC_PER_BROKER`”.

Para obter informações sobre como encontrar IDs de zona de disponibilidade, consulte IDs AZ para seu recurso no guia AWS Resource Access Manager do usuário.

Para obter informações sobre como definir propriedades de configuração, consulte Configuração.

Para obter mais informações sobre o KIP-392, consulte Permitir que os consumidores busquem a partir da réplica mais próxima nas páginas do Confluence.

Para obter mais informações sobre a versão 2.4.1 do Apache Kafka, consulte as notas de release no site de downloads do Apache Kafka.

Suporte à versão Amazon MSK

Este tópico descreve o Política de suporte à versão Amazon MSK e o procedimento paraAtualizar a versão do Apache Kafka. Se você estiver atualizando sua versão do Kafka, siga as melhores práticas descritas em. Práticas recomendadas para atualizações de versão

Política de suporte à versão Amazon MSK

Esta seção descreve a política de suporte para as versões do Kafka compatíveis com o Amazon MSK.

- Todas as versões do Kafka são suportadas até atingirem a data de fim do suporte. Para obter detalhes sobre as datas de fim do suporte, consulteVersões compatíveis do Apache Kafka. Atualize seu cluster MSK para a versão recomendada do Kafka ou superior antes da data de fim do suporte. Para obter detalhes sobre como atualizar sua versão do Apache Kafka, consulte. Atualizar a versão do Apache Kafka Um cluster usando uma versão do Kafka após a data de término do suporte é atualizado automaticamente para a versão recomendada do Kafka.
- O MSK eliminará gradualmente o suporte para clusters recém-criados que usam versões do Kafka com datas de fim de suporte publicadas.

Atualizar a versão do Apache Kafka

É possível atualizar um cluster do MSK existente para uma versão mais recente do Apache Kafka. Não é possível atualizá-lo para uma versão mais antiga. Ao atualizar a versão do Apache Kafka de um cluster do MSK, verifique também o software no lado do cliente para confirmar se a versão permite que você use os recursos da nova versão do Apache Kafka do cluster. O Amazon MSK atualiza somente o software do servidor. Ele não atualiza os clientes.

Para obter informações sobre como tornar um cluster altamente disponível durante uma atualização, consulte the section called “Criar clusters altamente disponíveis”.

> **Important**
>
> Você não pode atualizar a versão do Apache Kafka para um cluster do MSK que exceda os limites descritos em the section called “ Dimensione seu cluster adequadamente: número de partições por agente”.

Atualizando a versão do Apache Kafka usando o AWS Management Console

1. Abra o console do Amazon MSK em https://console.aws.amazon.com/msk/.
2. Escolha o cluster do MSK no qual você deseja atualizar a versão do Apache Kafka.
3. Na guia Propriedades, escolha Atualizar na seção Versão do Apache Kafka.

Atualizando a versão do Apache Kafka usando o AWS CLI

1. Execute o comando a seguir, *ClusterArn* substituindo-o pelo Amazon Resource Name (ARN) que você obteve ao criar seu cluster. Se você não tiver o ARN do cluster, poderá encontrá-lo listando todos os clusters. Para ter mais informações, consulte the section called “Listar clusters”.

   ```
   aws kafka get-compatible-kafka-versions --cluster-arn ClusterArn
   ```

   A saída desse comando inclui uma lista das versões do Apache Kafka para as quais você pode atualizar o cluster. Ela se parece com o exemplo a seguir.

   ```
   {
       "CompatibleKafkaVersions": [
           {
               "SourceVersion": "2.2.1",
               "TargetVersions": [
                   "2.3.1",
                   "2.4.1",
                   "2.4.1.1",
                   "2.5.1"
               ]
           }
       ]
   }
   ```

2. Execute o comando a seguir, *ClusterArn* substituindo-o pelo Amazon Resource Name (ARN) que você obteve ao criar seu cluster. Se você não tiver o ARN do cluster, poderá encontrá-lo listando todos os clusters. Para ter mais informações, consulte the section called “Listar clusters”.

Substitua *Current-Cluster-Version* pela versão atual do cluster. Pois *TargetVersion*você pode especificar qualquer uma das versões de destino a partir da saída do comando anterior.

> **Important**
>
> As versões de cluster não são inteiros simples. Para encontrar a versão atual do cluster, use a DescribeClusteroperação ou o comando AWS CLI describe-cluster. Uma versão de exemplo é KTVPDKIKX0DER.

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

A saída do comando anterior é semelhante ao JSON a seguir.

```
{

    "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. Para obter o resultado da `update-cluster-kafka-version` operação, execute o comando a seguir, substituindo *ClusterOperationArn* pelo ARN obtido na saída do `update-cluster-kafka-version` comando.

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

A saída desse comando `describe-cluster-operation` é semelhante ao seguinte JSON de exemplo.

```
{
    "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"
        }
    }
}
```

Se `OperationState` tiver o valor `UPDATE_IN_PROGRESS`, aguarde um pouco e execute o comando `describe-cluster-operation` novamente. Quando a operação for concluída, o valor de `OperationState` será transformado em `UPDATE_COMPLETE`. Como o tempo necessário para que o Amazon MSK conclua a operação varia, talvez seja necessário verificar repetidamente até que a operação seja concluída.

Atualizar a versão do Apache Kafka usando a API

1. Invoque a GetCompatibleKafkaVersionsoperação para obter uma lista das versões do Apache Kafka para as quais você pode atualizar o cluster.
2. Invoque a UpdateClusterKafkaVersionoperação para atualizar o cluster para uma das versões compatíveis do Apache Kafka.

Práticas recomendadas para atualizações de versão

Para garantir a continuidade do cliente durante a atualização contínua que é realizada como parte do processo de atualização da versão do Kafka, revise a configuração dos seus clientes e os tópicos do Apache Kafka da seguinte forma:

- Defina o fator de replicação (RF) do tópico como um valor mínimo 2 para clusters de duas AZ e um valor mínimo de 3 para clusters de três AZ. Um valor de RF de 2 pode levar a partições off-line durante a aplicação de patches.
- Defina o mínimo de réplicas sincronizadas (miniSR) para um valor máximo de para garantir que o conjunto de réplicas de `RF - 1` partições possa tolerar que uma réplica fique off-line ou sub-replicada.
- Configure os clientes para usar várias cadeias de conexão do broker. Ter vários corretores na cadeia de conexão de um cliente permite o failover se um agente específico que oferece suporte à E/S do cliente começar a ser corrigido. Para obter informações sobre como obter uma cadeia de conexão com vários agentes, consulte Obter os corretores de bootstrap para um cluster Amazon MSK.
- Recomendamos que você atualize os clientes de conexão para a versão recomendada ou superior para se beneficiar dos recursos disponíveis na nova versão. As atualizações do cliente não estão sujeitas às datas de fim da vida útil (EOL) da versão Kafka do seu cluster MSK e não precisam ser concluídas até a data de EOL. O Apache Kafka fornece uma política bidirecional de compatibilidade de clientes que permite que clientes mais antigos trabalhem com clusters mais novos e vice-versa.
- Os clientes Kafka que usam as versões 3.x.x provavelmente virão com os seguintes padrões: e. `acks=all` `enable.idempotence=true` `acks=all`é diferente do padrão anterior `acks=1` e fornece durabilidade extra ao garantir que todas as réplicas sincronizadas reconheçam a solicitação de produção. Da mesma forma, o padrão para `enable.idempotence` era anteriormente`false`. A alteração para `enable.idempotence=true` o padrão reduz a probabilidade de mensagens duplicadas. Essas alterações são consideradas configurações de

melhores práticas e podem introduzir uma pequena quantidade de latência adicional que está dentro dos parâmetros normais de desempenho.

- Use a versão recomendada do Kafka ao criar novos clusters MSK. Usar a versão recomendada do Kafka permite que você se beneficie dos recursos mais recentes do Kafka e do MSK.

Solução de problemas no cluster do Amazon MSK

As informações a seguir podem ajudar você a solucionar problemas que possam surgir com seu cluster do Amazon MSK. Você também pode publicar seu problema no AWS re:Post.

Tópicos

A substituição do volume causa saturação do disco devido à sobrecarga de replicação

Durante uma falha de hardware de volume não planejada, o Amazon MSK pode substituir o volume por uma nova instância. O Kafka preenche novamente o novo volume replicando partições de outros corretores no cluster. Depois que as partições são replicadas e recuperadas, elas se qualificam para serem membros de liderança e réplica em sincronia (ISR).

Problema

Em uma corretora se recuperando da substituição de volume, algumas partições de tamanhos variados podem voltar a ficar on-line antes de outras. Isso pode ser problemático, pois essas partições podem estar fornecendo tráfego do mesmo agente que ainda está recuperando (replicando) outras partições. Às vezes, esse tráfego de replicação pode saturar os limites de taxa de transferência do volume subjacente, que são 250 MiB por segundo no caso padrão. Quando essa saturação ocorre, todas as partições que já estão ocupadas serão afetadas, resultando em latência em todo o cluster para qualquer corretor que compartilhe ISR com essas partições capturadas (não apenas partições líderes devido a acks remotos). `acks=all` Esse problema é mais comum em clusters maiores que têm um número maior de partições que variam em tamanho.

Recomendação

- Para melhorar a postura de E/S de replicação, certifique-se de que as configurações de encadeamento de práticas recomendadas estejam em vigor.
- Para reduzir a probabilidade de saturação do volume subjacente, habilite o armazenamento provisionado com uma taxa de transferência mais alta. Um valor mínimo de taxa de transferência de 500 MiB/s é recomendado para casos de replicação de alta taxa de transferência, mas o valor real necessário variará de acordo com a taxa de transferência e o caso de uso. Provisionar throughput de armazenamento.
- Para minimizar a pressão de replicação, reduza `num.replica.fetchers` para o valor padrão de2.

Grupo de consumidores preso no estado **`PreparingRebalance`**

Se um ou mais de seus grupos de consumidores estiverem presos em um estado perpétuo de rebalanceamento, a causa disso pode ser o problema KAFKA-9752 do Apache Kafka, que afeta as versões 2.3.1 e 2.4.1 do Apache Kafka.

Para solucionar esse problema, recomendamos que você atualize seu cluster para a versão Correção de bugs do Amazon MSK versão 2.4.1.1, que contém uma correção para esse problema. Para obter informações sobre a atualização de um cluster existente para a versão 2.4.1.1 de correção de bugs do Amazon MSK, consulte Atualizar a versão do Apache Kafka.

As soluções alternativas para resolver esse problema sem atualizar o cluster para a versão 2.4.1.1 de correção de bugs do Amazon MSK são definir os clientes do Kafka para usar Protocolo de associação estática ou Identificar e reiniciar o nó do agente de coordenação do grupo de consumidores que está preso.

Implementação de protocolo de associação estática

Para implementar o protocolo de associação estática em seus clientes, faça o seguinte:

1. Defina a propriedade `group.instance.id` da sua configuração Consumidores do Kafka como uma string estática que identifica o consumidor no grupo.
2. Certifique-se de que outras instâncias da configuração sejam atualizadas para usar a string estática.
3. Implante as mudanças em seus consumidores do Kafka.

O uso do Protocolo de associação estática é mais eficaz se o tempo limite da sessão na configuração do cliente for definido para uma duração que permita ao consumidor se recuperar sem acionar prematuramente um rebalanceamento do grupo de consumidores. Por exemplo, se sua aplicação consumidora conseguir tolerar 5 minutos de indisponibilidade, um valor razoável para o tempo limite da sessão seria 4 minutos em vez do valor padrão de 10 segundos.

> **Note**
>
> O uso do protocolo de associação estática simplesmente reduz a probabilidade de se deparar com esse problema. Você ainda poderá se deparar com esse problema mesmo ao usar o protocolo de associação estática.

Como reinicializar o nó do agente de coordenação

Para reinicializar o nó agente de coordenação, faça o seguinte:

1. Identifique o coordenador do grupo usando o comando `kafka-consumer-groups.sh`.

2. Reinicie o coordenador do grupo de consumidores bloqueados usando a ação RebootBroker da API.

Erro ao entregar os registros do corretor para o Amazon CloudWatch Logs

Ao tentar configurar seu cluster para enviar registros do agente para a Amazon CloudWatch Logs, você pode obter uma das duas exceções.

Se você receber uma exceção `InvalidInput.LengthOfCloudWatchResourcePolicyLimitExceeded`, tente novamente, mas use grupos de log que começam com `/aws/vendedlogs/`. Para obter mais informações, consulte Habilitar o registro em log de determinados serviços da Amazon Web Services.

Se você receber uma `InvalidInput.NumberOfCloudWatchResourcePoliciesLimitExceeded` exceção, escolha uma política existente do Amazon CloudWatch Logs em sua conta e acrescente o seguinte JSON a ela.

```
{"Sid":"AWSLogDeliveryWrite","Effect":"Allow","Principal":
{"Service":"delivery.logs.amazonaws.com"},"Action":
["logs:CreateLogStream","logs:PutLogEvents"],"Resource":["*"]}
```

Se você tentar anexar o JSON acima a uma política existente, mas receber um erro informando que você atingiu o tamanho máximo da política escolhida, tente anexar o JSON a outra de suas políticas do Amazon Logs. CloudWatch Depois de acrescentar o JSON a uma política existente, tente novamente configurar a entrega de registros do corretor para o Amazon Logs. CloudWatch

Nenhum grupo de segurança padrão

Se você tentar criar um cluster e obter um erro indicando que não há grupo de segurança padrão, talvez esteja usando uma VPC que foi compartilhada com você. Peça para o administrador conceder permissão para descrever os grupos de segurança nesta VPC e tente novamente. Para obter um exemplo de uma política que permita esta ação, consulte Amazon EC2: permite o gerenciamento de grupos de segurança do EC2 associados a uma VPC específica de forma programática e no console.

O cluster parece estar preso no estado CRIANDO

Às vezes a criação do cluster pode levar até 30 minutos. Aguarde 30 minutos e verifique o estado do cluster novamente.

O estado do cluster é alterado de CRIANDO para COM FALHA

Tente criar o cluster novamente.

O estado do cluster está ATIVO, mas os produtores não conseguem enviar dados ou os consumidores não conseguem receber dados

- Se a criação do cluster tiver êxito (o estado do cluster será `ACTIVE`), mas não será possível enviar nem receber dados. Certifique-se de que os aplicativos produtor e consumidor tenham acesso ao cluster. Para obter mais informações, consulte as diretrizes no the section called “Etapa 3: criar uma máquina cliente”.

- Caso os produtores e os consumidores tenham acesso ao cluster, mas ainda assim enfrentem problemas ao gerar e consumir dados, a causa pode ser KAFKA-7697, que afeta o Apache Kafka versão 2.1.0 e pode levar a um deadlock em um ou mais agentes. Considere migrar para o Apache Kafka 2.2.1, que não é afetado por este bug. Para obter informações sobre como migrar, consulte Migração.

AWS CLI não reconhece o Amazon MSK

Se você o tiver AWS CLI instalado, mas ele não reconhecer os comandos do Amazon MSK, atualize-o AWS CLI para a versão mais recente. Para obter instruções detalhadas sobre como atualizar o AWS CLI, consulte Instalando AWS Command Line Interface o. Para obter informações sobre como usar os comandos AWS CLI para executar o Amazon MSK, consulteComo funciona.

As partições ficam offline ou as réplicas estão fora de sincronia

Estes podem ser sintomas de pouco espaço em disco. Consulte the section called “O espaço em disco está acabando”.

O espaço em disco está acabando

Consulte as melhores práticas para gerenciar o espaço em disco: the section called “Monitorar o espaço em disco” e the section called “Ajustar os parâmetros de retenção de dados”.

A memória está baixa

Caso a métrica `MemoryUsed` esteja alta ou a `MemoryFree` esteja baixa, isso não significa que existe um problema. O Apache Kafka foi desenvolvido para usar o máximo de memória possível, que é gerenciada de forma ideal.

O produtor recebe NotLeaderForPartitionException

Geralmente, isto é um erro transitório. Defina o parâmetro de configuração de `retries` do produtor com um valor mais alto que o atual.

Número de partições com replicação insuficiente (URP) maior que zero

A `UnderReplicatedPartitions` é uma métrica importante e deve ser monitorada. Em um cluster MSK íntegro, essa métrica tem o valor igual a 0. Se for maior que zero, isso pode ocorrer por um dos motivos a seguir.

- Se `UnderReplicatedPartitions` estiver apresentando picos, o problema pode ser que o cluster não foi provisionado no tamanho correto para tratar o tráfego de entrada e saída. Consulte Práticas recomendadas.
- Se `UnderReplicatedPartitions` for consistentemente maior que 0, inclusive durante períodos de baixo tráfego, talvez o problema decorra de você ter definido ACLs restritivas que não concedem o acesso aos tópicos para os agentes. Para replicar partições, os agentes devem estar autorizados a READ (ler) e DESCRIBE (descrever) os tópicos. DESCRIBE é concedido por padrão com a autorização READ. Para obter informações sobre a configuração de ACLs, consulte Autorização e ACLs na documentação do Apache Kafka.

O cluster tem tópicos chamados __amazon_msk_canary e __amazon_msk_canary_state

Você pode ver que seu cluster do MSK tem um tópico com o nome `__amazon_msk_canary` e outro com o nome `__amazon_msk_canary_state`. Trata-se de tópicos internos que o Amazon MSK cria e usa para métricas de integridade e diagnóstico do cluster. Esses tópicos têm um tamanho insignificante e não podem ser excluídos.

Falha na replicação de partições

Certifique-se de não ter definido ACLs em CLUSTER_ACTIONS.

Não é possível acessar o cluster que está com o acesso público ativado

Siga as etapas abaixo se o seu cluster estiver com o acesso público ativado, mas você ainda não conseguir acessá-lo pela Internet:

1. Certifique-se de que as regras de entrada do grupo de segurança do cluster tenham permissão para seu endereço IP e a porta do cluster. Para obter uma lista dos números de portas do cluster, consulte the section called “Informações de porta”. Certifique-se também de que as regras de saída do grupo de segurança permitam comunicações de saída. Para ter mais informações sobre grupos de segurança e suas regras de entrada e saída, consulte Grupos de segurança para sua VPC no Guia do usuário da Amazon VPC.
2. Certifique-se de que seu endereço IP e a porta do cluster tenham permissão nas regras de entrada da ACL da rede VPC do cluster. Diferentemente dos grupos de segurança, as ACLs de rede não têm estado. Isso significa que você deve configurar as regras de entrada e saída. Nas regras de saída, permita que todo o tráfego (intervalo de portas: 0-65535) chegue ao seu endereço IP. Para obter mais informações, consulte Adicionar e excluir regras no Guia do usuário da Amazon VPC.
3. Verifique se você está usando a string bootstrap-brokers de acesso público para acessar o cluster. Um cluster do MSK com acesso público ativado tem duas strings distintas de agentes de inicialização, uma para acesso público e outra para acesso interno diretamente da AWS. Para ter mais informações, consulte the section called “Obtendo os corretores de bootstrap usando o AWS Management Console”.

Não é possível acessar o cluster de dentro AWS: problemas de rede

Se você tiver uma aplicação do Apache Kafka que não consiga se comunicar com êxito com um cluster do MSK, comece executando o teste de conectividade a seguir.

1. Use qualquer um dos métodos descritos em the section called “Como obter os agentes de bootstrap” para obter os endereços dos agentes de bootstrap.
2. No comando a seguir, substitua *bootstrap-broker* por um dos endereços do agente que você obteve na etapa anterior. Substitua *port-number* por 9094 se o cluster estiver configurado para usar a autenticação TLS. Se o cluster não usar a autenticação TLS, substitua *port-number* por 9092. Execute o comando usando a máquina cliente.

   ```
   telnet bootstrap-broker port-number
   ```

   Onde o número da porta é:

   - 9094 se o cluster estiver configurado para usar a autenticação TLS.
   - 9092 Se o cluster não usar a autenticação TLS.
   - Um número de porta diferente é necessário se o acesso público estiver habilitado.

   Execute o comando usando a máquina cliente.
3. Repita o comando anterior para todos os agentes de bootstrap.

Se a máquina cliente conseguir acessar os corretores, isso significa que não há problemas de conectividade. Nesse caso, execute o comando a seguir para verificar se o cliente do Apache Kafka está configurado corretamente. Para obter *bootstrap-brokers*, use qualquer um dos métodos descritos em the section called “Como obter os agentes de bootstrap”. Substitua *topic* pelo nome do tópico.

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

Se o comando anterior for bem-sucedido, isso indica que o cliente está configurado corretamente. Se você ainda não consegue produzir e consumir de um aplicativo, depure o problema no nível do aplicativo.

Se a máquina cliente não conseguir acessar os corretores, consulte as subseções a seguir para obter orientação baseada na configuração da máquina cliente.

Cliente do Amazon EC2 e cluster do MSK na mesma VPC

Se a máquina cliente estiver na mesma VPC que o cluster do MSK, verifique se o grupo de segurança do cluster tem uma regra de entrada que aceite tráfego do grupo de segurança da máquina cliente. Para obter informações sobre como configurar essas regras, consulte Regras do grupo de segurança. Para obter um exemplo de como acessar um cluster de uma instância do Amazon EC2 que esteja na mesma VPC do cluster, consulte Conceitos básicos.

Cliente do Amazon EC2 e cluster do MSK em VPCs diferentes

Se a máquina cliente e o cluster estiverem em duas VPCs diferentes, verifique se:

- As duas VPCs estão emparelhadas.
- O status da conexão de emparelhamento está ativo.
- As tabelas de rotas das duas VPCs estão configuradas corretamente.

Para obter informações sobre o emparelhamento de VPC, consulte Trabalhar com conexões de emparelhamento de VPC.

Cliente on-premises

No caso de um cliente local configurado para se conectar ao cluster MSK usando AWS VPN, verifique o seguinte:

- O status da conexão VPN é UP. Para obter informações sobre como verificar o status da conexão VPN, consulte Como verificar o status atual do meu túnel VPN?.
- A tabela de rotas da VPC do cluster contém a rota para um CIDR on-premises, cujo destino tem o formato `Virtual private gateway(vgw-xxxxxxxx)`.
- O grupo de segurança do cluster do MSK permite tráfego na porta 2181, na porta 9092 (se o cluster aceitar tráfego em texto simples) e na porta 9094 (se o cluster aceitar tráfego com criptografia TLS).

Para obter mais orientações sobre AWS VPN solução de problemas, consulte Solução de problemas do Client VPN.

AWS Direct Connect

Se o cliente usar AWS Direct Connect, consulte Solução de problemas AWS Direct Connect.

Se as orientações para a solução de problemas anteriores não resolverem a situação, certifique-se de que nenhum firewall esteja bloqueando o tráfego de rede. Para depuração adicional, use ferramentas como `tcpdump` e `Wireshark` para analisar o tráfego e garantir que ele esteja alcançando o cluster do MSK.

Falha na autenticação: muitas conexões

O erro `Failed authentication ... Too many connects` indica que um agente está se protegendo porque um ou mais clientes do IAM estão tentando se conectar a ele em um ritmo agressivo. Para ajudar os agentes a aceitarem uma taxa maior de novas conexões do IAM, você pode aumentar o parâmetro de configuração `reconnect.backoff.ms`.

Para saber mais sobre os limites de taxa para novas conexões por agente, consulte a página Cota do Amazon MSK.

MSK com tecnologia sem servidor: falha na criação do cluster

Se você tentar criar um cluster do MSK com a tecnologia sem servidor e o fluxo de trabalho falhar, talvez você não tenha permissão para criar um endpoint da VPC. Verifique se o administrador concedeu permissão para você criar um endpoint da VPC permitindo a ação `ec2:CreateVpcEndpoint`.

Para obter uma lista completa das permissões necessárias para realizar todas as ações do Amazon MSK, consulte AWS política gerenciada: AmazonMSK FullAccess.

Práticas recomendadas

Este tópico descreve algumas práticas recomendadas para seguir ao usar o Amazon MSK.

Dimensione seu cluster adequadamente: número de partições por agente

A tabela a seguir mostra o número recomendado de partições (incluindo partições líderes e seguidoras) por agente.

Tamanho do corretor	Número recomendado de partições (incluindo partições líderes e seguidoras) por agente
`kafka.t3.small`	300
`kafka.m5.large` ou `kafka.m5.xlarge`	1000
`kafka.m5.2xlarge`	2000
`kafka.m5.4xlarge` , `kafka.m5.8xlarge` , `kafka.m5.12xlarge` , `kafka.m5.16xlarge` ou `kafka.m5.24xlarge`	4000
`kafka.m7g.large` ou `kafka.m7g.xlarge`	1000
`kafka.m7g.2xlarge`	2000
`kafka.m7g.4xlarge` ,`kafka.m7g.8xlarge` ,`kafka.m7g.12xlarge` , ou `kafka.m7g.16xlarge`	4000

Se o número de partições por agente exceder o valor recomendado e seu cluster ficar sobrecarregado, você poderá ser impedido de realizar as seguintes operações:

- Atualizar a configuração do cluster

- Atualize o cluster para um tamanho de agente menor
- Associe um AWS Secrets Manager segredo a um cluster que tenha autenticação SASL/SCRAM

Um grande número de partições também pode resultar na falta de métricas do Kafka na coleta de dados do Prometheus. CloudWatch

Para obter orientações sobre como escolher o número de partições, consulte Apache Kafka Supports 200K Partitions Per Cluster. Também recomendamos que você realize seus próprios testes para determinar o tamanho certo para seus corretores. Para obter mais informações sobre os diferentes tamanhos de corretores, consultethe section called “Tamanhos de corretores”.

Dimensione seu cluster adequadamente: número de agentes por cluster

Para determinar o número certo de agentes correto para seu cluster do MSK e entender os custos, consulte a planilha Preço e dimensionamento do MSK. Essa planilha fornece uma estimativa para dimensionar um cluster do MSK e os custos associados do Amazon MSK em relação a um cluster do Apache Kafka semelhante, autogerenciado e baseado no EC2. Para obter mais informações sobre os parâmetros de entrada na planilha, passe o mouse sobre as descrições dos parâmetros. As estimativas fornecidas por essa planilha são conservadoras e fornecem um ponto de partida para um novo cluster. O desempenho, o tamanho e os custos do cluster dependerão do seu caso de uso e recomendamos que você os verifique com testes reais.

Para entender como a infraestrutura subjacente afeta o desempenho do Apache Kafka, consulte Melhores práticas para dimensionar corretamente seus clusters do Apache Kafka para otimizar desempenho e custo no blog de Big Data. AWS A postagem do blog fornece informações sobre como dimensionar seus clusters para atender aos requisitos de throughput, disponibilidade e latência. Ela também fornece respostas para perguntas como quando você deve aumentar a escala verticalmente ou horizontalmente, além de orientações sobre como verificar continuamente o tamanho dos seus clusters de produção.

Otimize a taxa de transferência do cluster para instâncias m5.4xl, m7g.4xl ou maiores

Ao usar instâncias m5.4xl, m7g.4xl ou maiores, você pode otimizar a taxa de transferência do cluster ajustando as configurações num.io.threads e num.network.threads.

Num.io.threads é o número de threads que um agente usa para processar solicitações. Adicionar mais threads, até o número de núcleos de CPU compatíveis com o tamanho da instância, pode ajudar a melhorar a taxa de transferência do cluster.

Num.network.threads é o número de threads que o agente usa para receber todas as solicitações recebidas e retornar respostas. Os threads de rede colocam as solicitações recebidas em uma fila de solicitações para processamento por io.threads. Definir num.network.threads como metade do número de núcleos de CPU compatíveis com o tamanho da instância permite o uso total do novo tamanho da instância.

> **Important**
>
> Não aumente num.network.threads sem antes aumentar num.io.threads, pois isso pode causar congestionamento relacionado à saturação da fila.

Configurações recomendadas

Tamanho da instância	Valor recomendado para num.io.threads	Valor recomendado para 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

Use o Kafka mais recente AdminClient para evitar problemas de incompatibilidade de ID de tópico

O ID de um tópico é perdido (Erro: não corresponde ao ID do tópico para partição) quando você usa uma AdminClient versão do Kafka inferior à 2.8.0 com o sinalizador `--zookeeper` para aumentar ou reatribuir partições de tópico para um cluster usando a versão 2.8.0 ou superior do Kafka. Observe que o sinalizador `--zookeeper` ficou obsoleto no Kafka 2.5 e foi removido desde o Kafka 3.0. Consulte Atualização para a versão 2.5.0 de qualquer versão entre 0.8.x e 2.4.x.

Para evitar incompatibilidade de ID de tópico, use um cliente do Kafka versão 2.8.0 ou superior para operações administrativas do Kafka. Como alternativa, clientes 2.5 e superiores podem usar o sinalizador `--bootstrap-servers` em vez do sinalizador `--zookeeper`.

Criar clusters altamente disponíveis

Use as recomendações a seguir para que seu cluster MSK possa estar altamente disponível durante uma atualização (como quando você estiver atualizando o tamanho do agente ou a versão do Apache Kafka, por exemplo) ou quando o Amazon MSK estiver substituindo um agente.

- Configure um cluster com três zonas de disponibilidade.
- Certifique-se de que o Replication factor (RF – Fator de replicação) seja pelo menos 3. Observe que um RF de 1 pode resultar em partições offline durante uma atualização contínua; e um RF de 2 pode resultar em perda de dados.
- Defina réplicas mínimas em sincronização (minISR) para, no máximo, RF - 1. Uma minISR igual ao RF pode impedir a produção no cluster durante uma atualização sem interrupção. Uma minISR de 2 permite que tópicos replicados de três vias estejam disponíveis quando uma réplica estiver offline.
- Certifique-se de que as strings de conexão do cliente incluam pelo menos um agente de cada zona de disponibilidade. Ter vários agentes na string de conexão de um cliente possibilita o failover quando um agente específico estiver offline para uma atualização. Para obter informações sobre como obter uma string de conexão com vários agentes, consulte the section called “Como obter os agentes de bootstrap”.

Monitorar uso da CPU

O Amazon MSK recomenda veementemente que você mantenha a utilização total da CPU de seus agentes (definida como `CPU User + CPU System`) abaixo de 60%. Quando você tiver ao menos 40% da CPU total do seu cluster disponível, o Apache Kafka poderá redistribuir a carga da CPU entre os agentes no cluster quando necessário. Por exemplo, isso é necessário quando o Amazon MSK detecta e se recupera de uma falha do agente. Nesse caso, o Amazon MSK realiza a manutenção automática, como a aplicação de patches. Outro exemplo é quando um usuário solicita uma alteração no tamanho do corretor ou um upgrade de versão; nesses dois casos, o Amazon MSK implanta fluxos de trabalho contínuos que colocam um corretor off-line por vez. Quando os agentes com partições principais ficam offline, o Apache Kafka reatribui a liderança da partição para redistribuir o trabalho para outros agentes no cluster. Ao seguir essa prática recomendada, você pode garantir a disponibilidade suficiente de CPU em seu cluster para tolerar eventos operacionais como esses.

Você pode usar a matemática CloudWatch métrica da Amazon para criar uma métrica composta que seja`CPU User + CPU System`. Defina um alarme que seja acionado quando a métrica composta atingir uma utilização média de 60% da CPU. Quando esse alarme for acionado, escale o cluster usando uma das seguintes opções:

- Opção 1 (recomendada): atualize o tamanho do seu corretor para o próximo tamanho maior. Por exemplo, se o tamanho atual for`kafka.m5.large`, atualize o cluster a ser usado`kafka.m5.xlarge`. Lembre-se de que, ao atualizar o tamanho do agente no cluster, o Amazon MSK coloca os corretores off-line de forma contínua e transfere temporariamente a liderança da partição para outros corretores. Normalmente uma atualização de tamanho leva de 10 a 15 minutos por agente.
- Opção 2: se houver tópicos com todas as mensagens ingeridas de produtores que usam gravações de ida e volta (em outras palavras, as mensagens não recebem chaves e a ordenação não é importante para os consumidores), expanda seu cluster adicionando agentes. Também adicione partições aos tópicos existentes com o maior throughput. Em seguida, use `kafka-topics.sh --describe` para garantir que as partições recém-adicionadas sejam atribuídas aos novos agentes. O principal benefício dessa opção em comparação com a anterior é que você pode gerenciar recursos e custos de modo mais granular. Além disso, você pode usar essa opção se a carga da CPU exceder significativamente 60%, pois essa forma de escalabilidade normalmente não resulta em aumento de carga nos agentes existentes.
- Opção 3: expanda seu cluster adicionando agentes e, em seguida, reatribua as partições existentes usando a ferramenta de reatribuição de partições chamada `kafka-reassign-`

`partitions.sh`. No entanto, se você usar essa opção, o cluster precisará gastar recursos para replicar dados de um agente para outro após a reatribuição das partições. Em comparação com as duas opções anteriores, inicialmente isso pode aumentar significativamente a carga no cluster. Como resultado, o Amazon MSK não recomenda usar essa opção quando a utilização da CPU estiver acima de 70%, pois a replicação causará carga adicional da CPU e tráfego de rede. O Amazon MSK recomenda usar essa opção somente se as duas opções anteriores não forem viáveis.

Outras recomendações:

- Monitore a utilização total da CPU por agente como um indicador da distribuição de carga. Se os agentes tiverem uma utilização consistentemente desigual da CPU, isso pode ser um sinal de que a carga não está sendo distribuída uniformemente no cluster. O Amazon MSK recomenda o uso do Cruise Control para gerenciar continuamente a distribuição de carga por meio da atribuição de partições.
- Monitore a latência da produção e do consumo. A latência da produção e do consumo pode aumentar linearmente com a utilização da CPU.
- Intervalo de extração do JMX: se você habilitar o monitoramento aberto com o recurso Prometheus, recomenda-se usar um intervalo de extração de 60 segundos ou mais (scrape_interval: 60s) para a configuração do host do Prometheus (prometheus.yml). A redução do intervalo de coleta pode levar a um alto uso da CPU em seu cluster.

Monitorar o espaço em disco

Para evitar a falta de espaço em disco para mensagens, crie um CloudWatch alarme que observe a `KafkaDataLogsDiskUsed` métrica. Quando o valor dessa métrica atingir ou exceder 85%, execute uma ou mais das seguintes ações:

- Usar the section called “Escalabilidade automática”. Você também pode aumentar manualmente o armazenamento do agente, conforme descrito em the section called “Escalabilidade manual”.
- Reduza o período de retenção de mensagens ou o tamanho do log. Para obter informações sobre como fazer isso, consulte the section called “Ajustar os parâmetros de retenção de dados”.
- Exclua tópicos não utilizados.

Para obter informações sobre como configurar e usar alarmes, consulte Usando alarmes da Amazon CloudWatch . Para obter uma lista completa das métricas do Amazon MSK, consulte Como monitorar um cluster.

Ajustar os parâmetros de retenção de dados

Consumir mensagens não as remove do log. Para liberar espaço em disco regularmente, é possível especificar explicitamente um período de retenção, ou seja, por quanto tempo as mensagens permanecem no log. Também é possível especificar um tamanho do log de retenção. Quando o período de retenção ou o tamanho do log de retenção são atingidos, o Apache Kafka começa a remover segmentos inativos do log.

Para especificar uma política de retenção no nível do cluster, defina um ou mais dos seguintes parâmetros: `log.retention.hours`, `log.retention.minutes`, `log.retention.ms` ou `log.retention.bytes`. Para ter mais informações, consulte the section called “Configurações personalizadas”.

Também é possível especificar parâmetros de retenção no nível do tópico:

- Para especificar um período de retenção por tópico, use o comando a seguir.

```
kafka-configs.sh --bootstrap-server $bs --alter --entity-type topics --entity-
name TopicName --add-config retention.ms=DesiredRetentionTimePeriod
```

- Para especificar um tamanho de log de retenção por tópico, use o comando a seguir.

```
kafka-configs.sh --bootstrap-server $bs --alter --entity-type topics --entity-
name TopicName --add-config retention.bytes=DesiredRetentionLogSize
```

Os parâmetros de retenção especificados no nível do tópico têm precedência sobre os parâmetros no nível do cluster.

Como acelerar a recuperação de logs após um desligamento inadequado

Após um desligamento inadequado, um agente pode demorar um pouco para reiniciar, pois registra a recuperação em log. Por padrão, o Kafka usa apenas um thread por diretório de log para realizar

essa recuperação. Por exemplo, se você tiver milhares de partições, a conclusão da recuperação do log pode levar horas. Para acelerar a recuperação do log, recomenda-se aumentar o número de threads usando a propriedade de configuração `num.recovery.threads.per.data.dir`. É possível defini-la com o número de núcleos de CPU.

Monitorar a memória do Apache Kafka

Recomendamos que você monitore a memória que o Apache Kafka usa. Caso contrário, o cluster pode ficar indisponível.

Para determinar quanta memória o Apache Kafka usa, você pode monitorar a métrica `HeapMemoryAfterGC`. `HeapMemoryAfterGC` é o percentual da memória total da pilha que está em uso após a coleta de resíduos. Recomendamos que você crie um CloudWatch alarme que atue quando `HeapMemoryAfterGC` aumentar acima de 60%.

As etapas que você pode seguir para diminuir o uso da memória variam. Elas dependem da forma como você configura o Apache Kafka. Por exemplo, se você usar a entrega de mensagens transacionais, poderá diminuir o valor `transactional.id.expiration.ms` na configuração do Apache Kafka de `604800000` ms para `86400000` ms (de 7 dias para 1 dia). Isso diminui o espaço ocupado na memória de cada transação.

Não adicionar agentes que não são do MSK

Para clusters ZooKeeper baseados, se você usar ZooKeeper comandos do Apache para adicionar agentes, esses agentes não serão adicionados ao seu cluster MSK e seu Apache ZooKeeper conterá informações incorretas sobre o cluster. Isso pode resultar em perda de dados. Para consultar as operações de cluster compatíveis, consulte Como funciona.

Ativar a criptografia em trânsito

Para obter informações sobre a criptografia em trânsito e como ativá-la, consulte the section called “Criptografia em trânsito”.

Reatribuir partições

Para mover partições para agentes diferentes no mesmo cluster, é possível usar a ferramenta de reatribuição de partições, chamada `kafka-reassign-partitions.sh`. Por exemplo, depois de

adicionar novos agentes para expandir um cluster ou mover partições para remover agentes, você pode reequilibrar esse cluster reatribuindo partições aos novos corretores. Para obter informações sobre como adicionar agentes a um cluster, consulte the section called “Expandir um cluster”. Para obter informações sobre como remover agentes de um cluster, consultethe section called “Remover um corretor”. Para obter informações sobre a ferramenta de reatribuição de partições, consulte Expanding your cluster na documentação do Apache Kafka.

Histórico do documento para o Guia do desenvolvedor do Amazon MSK

A tabela a seguir descreve as alterações importantes feitas no Guia do desenvolvedor do Amazon MSK.

Última atualização da documentação: 25 de junho de 2024

Alteração	Descrição	Data
Foi adicionado o recurso Graviton Upgrade in place.	Você pode atualizar o tamanho do cluster broker de M5 ou T3 para M7g ou de M7g para M5.	25/06/2024
Data de fim do suporte 3.4.0 anunciada.	A data de fim do suporte para o Apache Kafka versão 3.4.0 é 17 de junho de 2025.	2024-6-24
Recurso de remoção de corretor adicionado.	Você pode reduzir a capacidade de armazenamento e computação do seu cluster provisionado removendo conjuntos de corretores, sem impacto na disponibilidade, risco de durabilidade de dados ou interrupção em seus aplicativos de streaming de dados.	16/05/2024
`WriteDataIdempotently` adicionado ao AWSMSKReplicatorExecutionRole	WriteDataIdempotently a permissão é adicionada à AWSMSKReplicatorExecutionRole política para oferecer suporte à replicação de dados entre clusters MSK.	16/05/2024

Alteração	Descrição	Data
Corretores Graviton M7g lançados no Brasil e no Bahrein.	O Amazon MSK agora oferece suporte à disponibilidade de corretores m7G nas regiões da América do Sul (sa-east-1, São Paulo) e Oriente Médio (me-south-1, Bahrein) usando processadores Graviton (processadores personali zados baseados em ARM criados pela Amazon Web Services). AWS	2024-2-07
Libere os corretores Graviton M7g para a região da China	O Amazon MSK agora oferece suporte à disponibilidade de corretores m7G na região da China usando processadores AWS Graviton (processadores personalizados baseados em ARM criados pela Amazon Web Services).	2024-01-11
Política de suporte da versão Amazon MSK Kafka	Foi adicionada uma explicaçã o sobre a política de suporte da versão Kafka compatíve l com o Amazon MSK. Para obter mais informações, consulte Versões do Apache Kafka.	2023-12-08

Alteração	Descrição	Data
Nova política de função de execução de serviços para dar suporte ao Amazon MSK Replicator.	O Amazon MSK adicionou uma nova `AWSMSKReplicatorExecutionRole` política para dar suporte ao Amazon MSK Replicator. Para obter mais informações, consulte Políticas gerenciadas pela AWS : `AWSMSKReplicatorExecutionRole`.	2023-12-06
Suporte para 7mG Graviton	O Amazon MSK agora oferece suporte a corretores M7g usando processadores AWS Graviton (processadores personalizados baseados em ARM criados pela Amazon Web Services).	2023-11-27
Replicador do Amazon MSK	O replicador do Amazon MSK é um novo recurso que você pode usar para replicar dados entre clusters do Amazon MSK. O Amazon MSK Replicator inclui uma atualização da política do FullAccess AmazonMSK. Para obter mais informações, consulte Políticas gerenciadas pela AWS : `AmazonMSKFullAccess`.	2023-09-28

Alteração	Descrição	Data
Atualização com as práticas recomendadas do IAM.	Guia atualizado para alinhamento com as práticas recomendadas do IAM. Para obter mais informações, consulte Práticas recomendadas de segurança no IAM.	2023-03-08
Atualizações ao perfil vinculado a serviço para compatibilidade com conectividade privada multi-VPC	O Amazon MSK agora inclui atualizações de funções AWSServiceRoleForKafka vinculadas a serviços para gerenciar interfaces de rede e endpoints de VPC em sua conta, tornando os agentes de cluster acessíveis aos clientes em sua VPC. O Amazon MSK usa permissões para `DescribeVpcEndpoints` , `ModifyVpcEndpoint` e `DeleteVpcEndpoints` . Para ter mais informações, consulte Uso de perfis vinculados a serviço para o Amazon MSK.	2023-03-08
Compatibilidade com Apache Kafka 2.7.2	Agora, o Amazon MSK é compatível com o Apache Kafka versão 2.7.2. Para ter mais informações, consulte Versões compatíveis do Apache Kafka.	2021-12-21

Alteração	Descrição	Data
Compatibilidade com Apache Kafka 2.6.3	Agora, o Amazon MSK é compatível com o Apache Kafka versão 2.6.3. Para ter mais informações, consulte Versões compatíveis do Apache Kafka.	2021-12-21
Pré-lançamento do MSK Serverless	O MSK Serverless é um novo recurso que você pode usar para criar clusters com a tecnologia sem servidor. Para ter mais informações, consulte MSK Serverless.	2021-11-29
Compatibilidade com Apache Kafka 2.8.1	Agora, o Amazon MSK é compatível com o Apache Kafka versão 2.8.1. Para ter mais informações, consulte Versões compatíveis do Apache Kafka.	2021-09-30
MSK Connect	O MSK Connect é um novo recurso que você pode usar para criar e gerenciar conectores do Apache Kafka. Para ter mais informações, consulte MSK Connect.	2021-09-16
Compatibilidade com Apache Kafka 2.7.1	Agora, o Amazon MSK é compatível com o Apache Kafka versão 2.7.1. Para ter mais informações, consulte Versões compatíveis do Apache Kafka.	2021-05-25

Alteração	Descrição	Data
Compatibilidade com Apache Kafka 2.8.0	Agora, o Amazon MSK é compatível com o Apache Kafka versão 2.8.0. Para ter mais informações, consulte Versões compatíveis do Apache Kafka.	2021-04-28
Compatibilidade com Apache Kafka 2.6.2	Agora, o Amazon MSK é compatível com o Apache Kafka versão 2.6.2. Para ter mais informações, consulte Versões compatíveis do Apache Kafka.	2021-04-28
Compatibilidade com atualização do tipo de agente	Agora, você pode alterar o tipo de agente de um cluster existente. Para ter mais informações, consulte Atualizando o tamanho do corretor.	2021-01-21
Compatibilidade com Apache Kafka 2.6.1	Agora, o Amazon MSK é compatível com o Apache Kafka versão 2.6.1. Para ter mais informações, consulte Versões compatíveis do Apache Kafka.	2021-01-19
Compatibilidade com Apache Kafka 2.7.0	Agora, o Amazon MSK é compatível com o Apache Kafka versão 2.7.0. Para ter mais informações, consulte Versões compatíveis do Apache Kafka.	2020-12-29

Alteração	Descrição	Data
Não há novos clusters no Apache Kafka versão 1.1.1	Você não pode mais criar um novo cluster do Amazon MSK com o Apache Kafka versão 1.1.1. No entanto, se você tiver clusters existente s do MSK executando o Apache Kafka versão 1.1.1, poderá continuar usando todos os recursos atualment e suportados nesses clusters existentes. Para ter mais informações, consulte Versões do Apache Kafka.	2020-11-24
Métricas de atraso do consumidor	Agora, o Amazon MSK fornece métricas que você pode usar para monitorar o atraso do consumidor. Para ter mais informações, consulte Como monitorar um cluster do Amazon MSK.	2020-11-23
Compatibilidade com Cruise Control	O Amazon MSK agora oferece suporte LinkedIn ao Cruise Control. Para ter mais informações, consulte Usando o LinkedIn Cruise Control para Apache Kafka com o Amazon MSK.	2020-11-17

Alteração	Descrição	Data
Compatibilidade com Apache Kafka 2.6.0	Agora, o Amazon MSK é compatível com o Apache Kafka versão 2.6.0. Para ter mais informações, consulte Versões compatíveis do Apache Kafka.	2020-10-21
Compatibilidade com Apache Kafka 2.5.1	Agora, o Amazon MSK é compatível com o Apache Kafka versão 2.5.1. Com o Apache Kafka versão 2.5.1, o Amazon MSK oferece suporte à criptografia em trânsito entre clientes e endpoints . ZooKeeper Para ter mais informações, consulte Versões compatíveis do Apache Kafka.	2020-09-30
Expansão automática de aplicação	Você pode configurar o Amazon Managed Streaming for Apache Kafka para expandir automaticamente o armazenamento do seu cluster em resposta ao aumento do uso. Para ter mais informações, consulte Escalabilidade automática.	2020-09-30

Alteração	Descrição	Data
Compatibilidade com segurança de nome de usuário e senha	Agora, o Amazon MSK é compatível com login em clusters usando nome de usuário e senha. O Amazon MSK armazena credenciais no AWS Secrets Manager. Para ter mais informações, consulte Autenticação SASL/SCRAM.	2020-09-17
Compatibilidade com a atualização da versão do Apache Kafka de um cluster do Amazon MSK	Agora, é possível atualizar a versão do Apache Kafka de um cluster existente do MSK.	28-05-2020
Suporte para nós de agente T3.small	Agora, o Amazon MSK é compatível com a criação de clusters com agentes do tipo T3.small do Amazon EC2.	2020-04-08
Compatibilidade com Apache Kafka 2.4.1	Agora, o Amazon MSK é compatível com o Apache Kafka versão 2.4.1.	02-04-2020
Suporte para logs de agente do streaming	Agora, o Amazon MSK pode transmitir registros do broker para CloudWatch Logs, Amazon S3 e Amazon Data Firehose. O Firehose pode, por sua vez, entregar esses registros aos destinos que ele suporta, como OpenSearch o Service.	25-02-2020
Compatibilidade com Apache Kafka 2.3.1	Agora, o Amazon MSK é compatível com o Apache Kafka versão 2.3.1.	19-12-2019

Alteração	Descrição	Data
Monitoramento aberto	Agora, o Amazon MSK é compatível com monitoramento aberto usando o Prometheus.	04-12-2019
Compatibilidade com Apache Kafka 2.2.1	Agora, o Amazon MSK é compatível com o Apache Kafka versão 2.2.1.	31-07-2019
Disponibilidade geral	Os novos recursos incluem suporte ao uso de tags, autenticação, criptografia TLS, configurações e a capacidade de atualizar o armazenamento de agentes.	30-05-2019
Compatibilidade com Apache Kafka 2.1.0	Agora, o Amazon MSK é compatível com o Apache Kafka versão 2.1.0.	05-02-2019

AWS Glossário

Para obter a AWS terminologia mais recente, consulte o AWS glossário na Glossário da AWS Referência.

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.