aws

Guia do Desenvolvedor

Amazon Data Firehose

Amazon Data Firehose: Guia do Desenvolvedor

Table of Contents

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.

O que é o Amazon Data Firehose?

O Amazon Data Firehose é um serviço totalmente gerenciado para fornecer streaming de dados em streaming em tempo real, a destinos como o Amazon Simple Storage Service (Amazon S3), o Amazon Redshift, o Amazon Serverless, o Splunk, o Apache Iceberg Tables e qualquer endpoint HTTP ou endpoint HTTP personalizado ou endpoints HTTP de propriedade de provedores de serviços terceirizados compatíveis, incluindo Datadog OpenSearch , Dynatrace e Monache Iceberg Tables e qualquer endpoint HTTP personalizado ou endpoints HTTP de propriedade de provedores de OpenSearch serviços terceirizados compatíveis, incluindo Datadog, Dynatrace e Monache Iceberg Tables e qualquer endpoint HTTP personalizado ou endpoints HTTP de propriedade de provedores de serviços terceirizados compatíveis, incluindo LogicMonitor Datadog, Dynatrace,, Monache Iceberg Tables e qualquer endpoint MongoDB, New Relic, Coralogix e Elastic. Com o Amazon Data Firehose, você não precisa escrever aplicações nem gerenciar recursos. Você configura os produtores de dados para enviar dados ao Amazon Data Firehose e ele entrega automaticamente os dados ao destino especificado. Você também pode configurar o Amazon Data Firehose para transformar os dados antes de entregá-los.

Para obter mais informações sobre as soluções de AWS big data da, consulte Big Data na AWS. Para obter mais informações sobre as soluções de dados em streaming da AWS , consulte O que são dados em streaming?

Noções básicas dos principais conceitos

Ao começar a usar o Amazon Data Firehose, pode ser vantajoso compreender os conceitos a seguir.

Fluxo do Firehose

A entidade subjacente do Amazon Data Firehose. Você usa o Amazon Data Firehose criando um fluxo do Firehose e enviando dados a ele. Para obter mais informações, consulte Tutorial: Criação de um fluxo do Firehose a partir do console e Envio de dados a um fluxo do Firehose.

Registro

Os dados de interesse que seu produtor de dados envia para um fluxo do Firehose. Um registro pode ter, no máximo, 1000 KB.

Produtor de dados

Os produtores enviam registros para os fluxos do Firehose. Por exemplo, um servidor Web que envia dados de log para um fluxo do Firehose é um produtor de dados. Você também pode

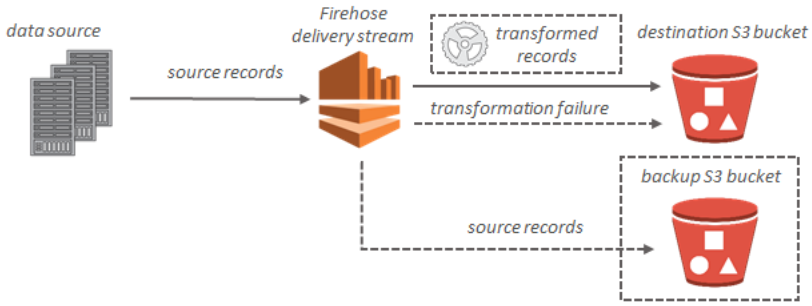
configurar o fluxo do Firehose para ler automaticamente os dados de um fluxo de dados existente do Kinesis e carregá-lo nos destinos. Para obter mais informações, consulte Envio de dados a um fluxo do Firehose.

Tamanho e intervalo de buffer

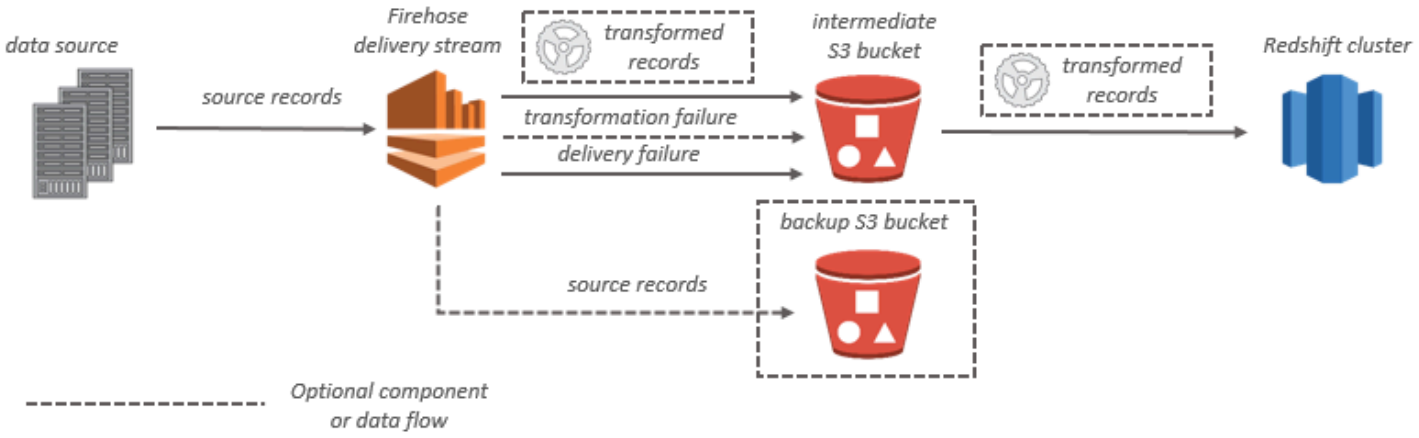
O Amazon Data Firehose armazena em buffer os dados em streaming recebidos até um determinado tamanho ou por um determinado período antes de entregá-los aos destinos. Buffer Sizeestá dentro MBs e Buffer Interval está em segundos.

Noções básicas sobre o fluxo de dados no Amazon Data Firehose

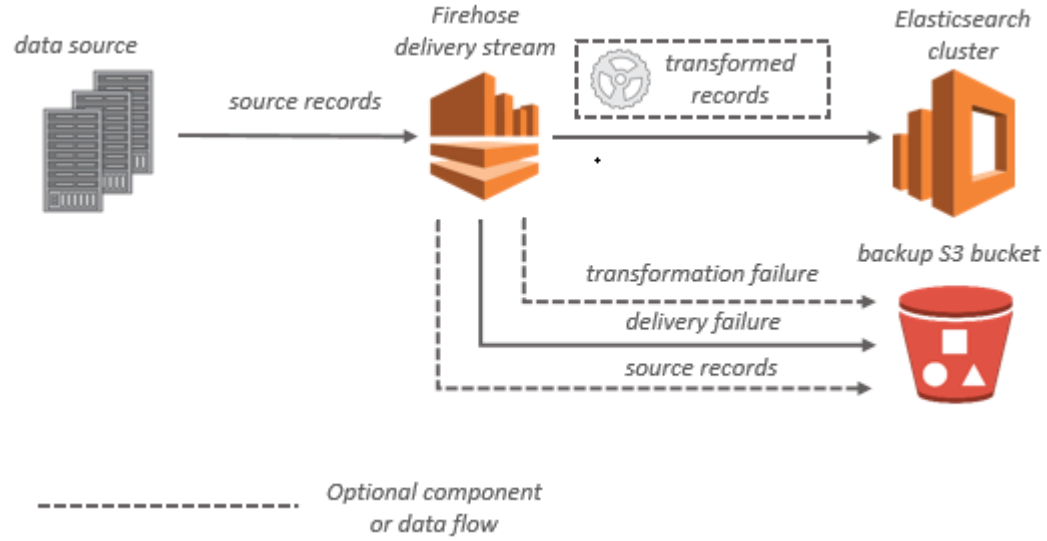
Para destinos do Amazon S3, os dados em streaming são entregues no bucket do S3. Se a transformação de dados estiver habilitada, você também poderá fazer backup dos dados da fonte em outro bucket do Amazon S3.



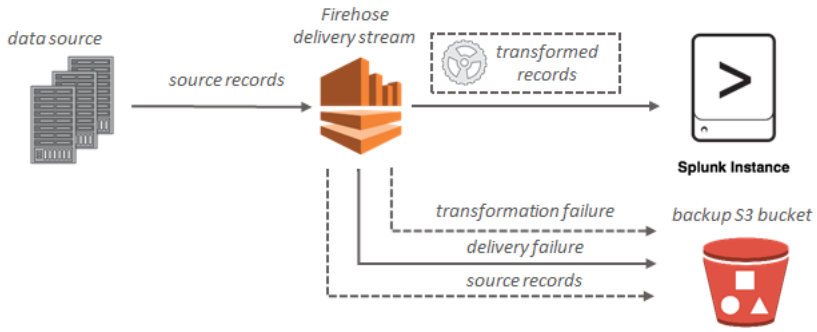
Para os destinos do Amazon Redshift, os dados em streaming são entregues primeiro no bucket do S3. Depois, o Amazon Data Firehose emite um comando COPY do Amazon Redshift para carregar os dados do bucket do S3 no cluster provisionado do Amazon Redshift. Se a transformação de dados estiver habilitada, você também poderá fazer backup dos dados da fonte em outro bucket do Amazon S3.



Para destinos no OpenSearch serviço, os dados em streaming são entregues ao cluster OpenSearch de serviços e você tem a opção de fazer backup desses dados em um bucket do S3 simultaneamente.



Para destinos do Splunk, os dados em streaming são entregues ao Splunk e eles podem ser submetidos a backup no bucket do S3 simultaneamente, se você desejar.



Uso do Firehose com um SDK AWS

AWS Os kits de desenvolvimento de software (SDKs) estão disponíveis para várias linguagens de programação populares. Cada SDK fornece uma API, exemplos de código e documentação que permitem que os desenvolvedores criem facilmente aplicações em seu idioma de preferência.

Documentação do SDK	Exemplos de código
AWS SDK para C++	AWS SDK para C++ exemplos de código
AWS CLI	AWS CLI exemplos de código
AWS SDK para Go	AWS SDK para Go exemplos de código
AWS SDK para Java	AWS SDK para Java exemplos de código
AWS SDK para JavaScript	AWS SDK para JavaScript exemplos de código
AWS SDK para Kotlin	AWS SDK para Kotlin exemplos de código
AWS SDK para .NET	AWS SDK para .NET exemplos de código
AWS SDK para PHP	AWS SDK para PHP exemplos de código
Ferramentas da AWS para PowerShell	Ferramentas para exemplos PowerShell de código
AWS SDK para Python (Boto3)	AWS SDK para Python (Boto3) exemplos de código

Documentação do SDK	Exemplos de código
AWS SDK para Ruby	AWS SDK para Ruby exemplos de código
AWS SDK para Rust	AWS SDK para Rust exemplos de código
SDK da AWS para SAP ABAP	SDK da AWS para SAP ABAP exemplos de código
AWS SDK for Swift	AWS SDK for Swift exemplos de código

Exemplo de disponibilidade

Não consegue encontrar o que precisa? Solicite um exemplo de código usando o link Fornecer feedback na parte inferior desta página.

Pré-requisitos completos para configuração do Amazon Data Firehose

Antes de usar o Amazon Data Firehose pela primeira vez, conclua as tarefas a seguir.

Tarefas

- Inscreva-se para AWS
- (Opcional) Download de bibliotecas e ferramentas

Inscreva-se para AWS

Quando você se inscreve no Amazon Web Services (AWS), sua AWS conta é automaticamente cadastrada em todos os serviços AWS, incluindo o Amazon Data Firehose. A cobrança incorrerá apenas pelos serviços utilizados.

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

Para se inscrever em uma AWS conta

1. Abra a https://portal.aws.amazon.com/billing/inscrição.
2. Siga as instruções online.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica ou uma mensagem de texto e inserir um código de verificação pelo 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 prática recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar tarefas que exigem acesso de usuário-raiz.

(Opcional) Download de bibliotecas e ferramentas

As bibliotecas e ferramentas a seguir o ajudarão a trabalhar com o Amazon Data Firehose de modo programático e usando a linha de comando:

- As Operações da API do Firehose são o conjunto básico de operações com suporte no Amazon Data Firehose.
- As AWS SDKs versões para Go, Java, .NET, Node.js, Python e Ruby incluem suporte e amostras do Amazon Data Firehose.

  Se sua versão do AWS SDK para Java não incluir amostras para o Amazon Data Firehose, você também pode baixar o AWS SDK mais recente em. GitHub
- A AWS Command Line Interface oferece suporte ao Amazon Data Firehose. AWS CLI Isso permite que você controle vários AWS serviços a partir da linha de comando e os automatize por meio de scripts.

Tutorial: Criação de um fluxo do Firehose a partir do console

Você pode usar o AWS Management Console ou um AWS SDK para criar um stream do Firehose para o destino escolhido.

Você pode atualizar a configuração do seu stream do Firehose a qualquer momento após sua criação, usando o console Amazon Data Firehose ou. UpdateDestination Seu fluxo do Firehose permanecerá no estado `Active` enquanto a configuração for atualizada, e será possível continuar enviando dados. A configuração atualizada normalmente entra em vigor em poucos minutos. O número da versão de um fluxo do Firehose será aumentado em 1 depois que você atualizar a configuração. Ele é refletido no nome do objeto do Amazon S3 entregue. Para obter mais informações, consulte Configuração de formato de nome de objeto do Amazon S3.

Execute as etapas nos tópicos a seguir para criar um fluxo do Firehose.

Tópicos

Escolha da fonte e do destino para seu fluxo do Firehose

1. Abra o console Firehose em. https://console.aws.amazon.com/firehose/
2. Escolha Criar fluxo do Firehose.
3. Na página Criar fluxo do Firehose, escolha uma fonte para seu fluxo do Firehose em uma das opções a seguir.

    - Direct PUT: escolha esta opção para criar um fluxo do Firehose no qual as aplicações de produção gravem diretamente. Para obter uma lista de serviços e agentes da AWS e dos serviços de código aberto integrados com o Direct PUT no Amazon Data Firehose. Essa lista não é completa e pode haver serviços adicionais que podem ser usados para enviar dados diretamente para o Firehose.

- AWS SDK
- AWS Lambda
- AWS CloudWatch Registros
- AWS CloudWatch Eventos
- AWS Fluxos métricos de nuvem
- AWS IoT
- AWS Eventbridge
- Amazon Simple Email Service
- Amazon SNS
- AWS Registros de ACL na web do WAF
- Amazon API Gateway: logs de acesso
- Amazon Pinpoint
- Logs do agente do Amazon MSK
- Logs de consultas do Amazon Route 53 Resolver
- AWS Registros de alertas do Firewall de Rede
- AWS Registros de fluxo do Firewall de Rede
- SLOWLOG do Amazon ElastiCache Redis
- Kinesis Agent (linux)
- Kinesis Tap (windows)
- Fluentbit
- Fluentd
- Apache Nifi
- Snowflake

- Amazon Kinesis Data Streams: escolha esta opção para configurar um fluxo do Firehose que use um fluxo de dados do Kinesis como fonte de dados. Você então poderá usar o Firehose para ler dados com facilidade de um fluxo de dados do Kinesis existente e carregá-lo nos destinos. Para obter mais informações sobre o uso do Kinesis Data Streams como fonte de dados, consulte Envio de dados a um fluxo do Firehose usando o Kinesis Data Streams.
- Amazon MSK: escolha esta opção para configurar um fluxo do Firehose que use o Amazon MSK como fonte de dados. Em seguida, é possível usar o Firehose para ler dados facilmente

de um cluster do Amazon MSK existente e carregá-los nos buckets do S3 especificados. Para obter mais informações, consulte Envio de dados a um fluxo do Firehose com o Amazon MSK.

4. Escolha um destino para seu fluxo do Firehose a partir de um dos destinos a seguir com suporte no Firehose.

   - OpenSearch Serviço Amazon
   - Amazon sem OpenSearch servidor
   - Amazon Redshift
   - Amazon S3
   - Tabelas do Apache Iceberg
   - Coralogix
   - Datadog
   - Dynatrace
   - Elastic
   - Endpoint de HTTP
   - Honeycomb
   - Logic Monitor
   - Logz.io
   - MongoDB Cloud
   - New Relic
   - Splunk
   - Splunk Observability Cloud
   - Sumo Logic
   - Snowflake

5. Em nome do fluxo do Firehose, é possível usar o nome que o console gerar para você ou adicionar um fluxo do Firehose de sua escolha.

Definição de configurações da fonte

É possível definir as configurações da fonte com base na fonte escolhida para enviar informações para um fluxo do Firehose a partir do console. É possível definir as configurações da fonte do

Amazon MSK e do Amazon Kinesis Data Streams como fonte. Não há configurações de fonte disponíveis para o Direct PUT como fonte.

Definição de configurações de fonte para o Amazon MSK

Ao escolher o Amazon MSK para enviar informações para um fluxo do Firehose, será possível escolher entre clusters provisionados pelo MSK e clusters do MSK com tecnologia sem servidor. Em seguida, é possível usar o Firehose para ler dados facilmente de um determinado cluster e tópico Amazon MSK e carregá-los no destino do S3 especificado.

Na seção Configurações da fonte da página, forneça valores para os campos a seguir.

Conectividade com o cluster do Amazon MSK

Escolha a opção Agentes privados de bootstrap (recomendado) ou Agentes públicos de bootstrap de acordo com a configuração do cluster. Os agentes de bootstrap são o que o cliente Apache Kafka usa como ponto de partida para se conectar ao cluster. Os corretores de bootstrap públicos são destinados ao acesso público externo AWS, enquanto os corretores de bootstrap privados são destinados ao acesso interno. AWS Para obter mais informações sobre o Amazon MSK, consulte Amazon Managed Streaming for Apache Kafka.

Para se conectar a um cluster do Amazon MSK provisionado ou sem servidor por meio de agentes privados de bootstrap, o cluster deve atender a todos os requisitos a seguir.

- O cluster deve estar ativo.
- O cluster deve ter o IAM como um dos métodos de controle de acesso.
- A conectividade privada de várias VPCs deve estar habilitada para o método de controle de acesso do IAM.
- Você deve adicionar a esse cluster uma política baseada em recursos que conceda à entidade principal do serviço do Firehose permissão de invocar a operação de API `CreateVpcConnection` do Amazon MSK.

Para se conectar a um cluster do Amazon MSK provisionado por meio de agentes de bootstrap públicos, o cluster deve atender a todos os requisitos a seguir.

- O cluster deve estar ativo.
- O cluster deve ter o IAM como um dos métodos de controle de acesso.
- O cluster deve ser acessível ao público.

Conta do cluster do MSK

É possível escolher a conta em que o cluster do Amazon MSK reside. Ela pode ser uma das opções a seguir.

- Conta atual — permite que você ingira dados de um cluster MSK na conta atual AWS . Para isso, você deve especificar o ARN do cluster do Amazon MSK no qual o fluxo do Firehose lerá os dados.
- Conta cruzada — permite que você ingira dados de um cluster MSK em outra conta. AWS Para obter mais informações, consulte Entrega entre contas do Amazon MSK.

Tópico

Especifique o tópico do Apache Kafka do qual você deseja que fluxo do Firehose ingira os dados. Você não pode atualizar este tópico após a conclusão da criação do fluxo do Firehose.

> **Note**
>
> O Firehose descompacta automaticamente as mensagens do Apache Kafka.

Definição de configurações de fonte para o Amazon Kinesis Data Streams

Defina as configurações de fonte do Amazon Kinesis Data Streams para enviar informações para um fluxo do Firehose da forma a seguir.

> **Important**
>
> Ao usar a Kinesis Producer Library (KPL) para gravar dados em um fluxo de dados do Kinesis, é possível usar agregação para combinar os registros gravados. Ao usar esse fluxo de dados como fonte para seu fluxo do Firehose, o Amazon Data Firehose desagregará os registros antes de entregá-los ao destino. Se você configurar seu fluxo do Firehose transformar os dados, o Amazon Data Firehose desagregará os registros antes de entregá-los ao AWS Lambda. Para obter mais informações, consulte Developing Amazon Kinesis Data Streams Producers Using the Kinesis Producer Library e Aggregation.

Em Configurações de fonte, escolha um fluxo existente na lista Fluxo de dados do Kinesis ou insira um ARN de fluxo de dados no formato `arn:aws:kinesis:[Region]:[AccountId]:stream/[StreamName]`.

Se não houver um fluxo de dados existente, escolha Criar para criar um novo no console do Amazon Kinesis Data Streams. Talvez você precise de um perfil do IAM que tenha a permissão necessária no fluxo do Kinesis. Para obter mais informações, consulte ???. Após criar um novo ´fluxo, selecione o ícone de atualização para atualizar a lista Fluxo do Kinesis. Se você tiver um grande número de fluxos, filtre a lista com a opção Filter by name.

> **Note**
>
> Quando um fluxo de dados do Kinesis é configurado como a fonte de um fluxo do Firehose, as operações `PutRecord` e `PutRecordBatch` do Amazon Data Firehose são desabilitadas. Para adicionar dados ao seu fluxo do Firehose nesse caso, use as operações `PutRecord` e `PutRecords` do Kinesis Data Streams.

O Amazon Data Firehose começa a ler os dados a partir da posição `LATEST` do seu fluxo do Kinesis. Para obter mais informações sobre as posições do Kinesis Data Streams, consulte. GetShardIterator

O Amazon Data Firehose chama a operação do Kinesis Data GetRecordsStreams uma vez por segundo para cada fragmento. Entretanto, quando o backup completo está ativado, o Firehose chama a operação `GetRecords` do Kinesis Data Streams duas vezes por segundo para cada fragmento, uma para o destino de entrega principal e outra para o backup completo.

Mais de um fluxo do Firehose podem ler o mesmo fluxo do Kinesis. Outras aplicações do Kinesis (consumidores) também podem ler o mesmo fluxo. Cada chamada de qualquer fluxo do Firehose ou de outra aplicação consumidora conta em relação ao limite total do controle de utilização para o fragmento. Para evitar a limitação, planeje suas aplicações cuidadosamente. Para obter mais informações sobre os limites do Kinesis Data Streams, consulte Limites do Amazon Kinesis Data Streams.

Vá para a próxima etapa para configurar a transformação do registro e a conversão de formato.

(Opcional) Configuração de transformação de registro e conversão de formato

Configure o Amazon Data Firehose para transformar e converter seus dados de registros.

Se você escolher o Amazon MSK como fonte para seu fluxo do Firehose.

Na seção Transformar registros de origem com AWS Lambda, forneça valores para o campo a seguir.

1. Transformação de dados

   Para criar um fluxo do Firehose que não transforme os dados recebidos, não marque a caixa de seleção Habilitar transformação de dados.

   Para especificar uma função do Lambda para o Firehose invocar e usar para transformar os dados recebidos antes de entregá-los, marque a caixa de seleção Habilitar transformação de dados. É possível configurar uma nova função do Lambda usando um dos esquemas do Lambda ou selecionar uma função do Lambda já existente. Sua função do Lambda deve conter o modelo de status exigido pelo Firehose. Para obter mais informações, consulte Transformação de dados da fonte no Amazon Data Firehose.
2. Na seção Convert record format (Converter formato do registro), forneça valores para o seguinte campo:

   Record format conversion (Conversão do formato do registro)

   Para criar um fluxo do Firehose que não converta o formato dos registros dos dados de entrada, selecione Desabilitado.

   Para converter o formato dos registros de entrada, selecione Enabled (Habilitado) e especifique o formato de saída que deseja. Você precisa especificar uma AWS Glue tabela que contenha o esquema que você deseja que o Firehose use para converter seu formato de registro. Para obter mais informações, consulte Conversão de formato de dados de entrada.

   Para obter um exemplo de como configurar a conversão do formato de registro com AWS CloudFormation, consulte AWS::KinesisFirehose:: DeliveryStream.

Se você escolher o Amazon Kinesis Data Streams ou o Direct PUT como fonte para seu stream do Firehose

Na seção Configurações da fonte, forneça os campos a seguir.

1. Em Transformar registros, escolha uma das opções a seguir:

a. Se seu destino for Amazon S3 ou Splunk, na seção Descompactar registros de origem CloudWatch Amazon Logs, escolha Ativar descompressão.

b. Na seção Transformar registros de origem com AWS Lambda, forneça valores para o seguinte campo:

Transformação de dados

Para criar um fluxo do Firehose que não transforme os dados recebidos, não marque a caixa de seleção Habilitar transformação de dados.

Para especificar uma função do Lambda para o Amazon Data Firehose invocar e usar para transformar os dados recebidos antes de entregá-los, marque a caixa de seleção Habilitar transformação de dados. É possível configurar uma nova função do Lambda usando um dos esquemas do Lambda ou selecionar uma função do Lambda já existente. A função do Lambda deve conter o modelo de status exigido pelo Amazon Data Firehose. Para obter mais informações, consulte Transformação de dados da fonte no Amazon Data Firehose.

2. Na seção Convert record format (Converter formato do registro), forneça valores para o seguinte campo:

Record format conversion (Conversão do formato do registro)

Para criar um fluxo do Firehose que não converta o formato dos registros dos dados de entrada, selecione Desabilitado.

Para converter o formato dos registros de entrada, selecione Enabled (Habilitado) e especifique o formato de saída que deseja. Você precisa especificar uma AWS Glue tabela que contenha o esquema que você deseja que o Amazon Data Firehose use para converter seu formato de registro. Para obter mais informações, consulte Conversão de formato de dados de entrada.

Para obter um exemplo de como configurar a conversão do formato de registro com AWS CloudFormation, consulte AWS::KinesisFirehose:: DeliveryStream.

Definição de configurações do destino

Esta seção descreve as configurações que você deve definir para seu fluxo do Firehose com base no destino selecionado.

Tópicos

Definições de configurações de destino para o Amazon S3

Você deve especificar as configurações a seguir para usar o Amazon S3 como destino para seu fluxo do Firehose.

- Insira valores para os seguintes campos.

S3 bucket

Escolha um bucket do S3 do qual você seja proprietário; os dados em streaming serão entregues nesse bucket. É possível criar um novo bucket do S3 ou escolher um já existente.

Novo delimitador de linha

É possível configurar seu fluxo do Firehose para adicionar um novo delimitador de linha entre os registros nos objetos que são entregues ao Amazon S3. Para fazer isso, escolha Habilitado. Para não adicionar um novo delimitador de linha entre registros nos objetos que são entregues ao Amazon S3, escolha Desabilitado. Se você planeja usar o Athena para consultar objetos do S3 com registros agregados, habilite essa opção.

Particionamento dinâmico

Escolha Habilitado para habilitar e configurar o particionamento dinâmico.

Desagregação de vários registros

Esse é o processo de análise de todos os registros no fluxo do Firehose, com sua separação baseada em JSON válido ou no delimitador de nova linha especificado.

Se você agregar vários eventos, logs ou registros em uma única PutRecord chamada de PutRecordBatch API, ainda poderá habilitar e configurar o particionamento dinâmico. Com dados agregados, quando você habilita o particionamento dinâmico, o Amazon Data Firehose analisa os registros e procura vários objetos JSON válidos em cada chamada de API. Quando o fluxo do Firehose é configurado com o Kinesis Data Stream como fonte, você também pode usar a agregação integrada na Kinesis Producer Library (KPL). A funcionalidade de partição de dados é executada após a desagregação dos dados. Portanto, cada registro em cada chamada de API pode ser entregue a diferentes prefixos do Amazon S3. Você também pode aproveitar a integração da função do Lambda para realizar qualquer outra desagregação ou qualquer outra transformação antes da funcionalidade de particionamento de dados.

> **Important**
>
> Se os dados estiverem agregados, o particionamento dinâmico só poderá ser aplicado após a desagregação de dados ser realizada. Portanto, se você habilitar o particionamento dinâmico para seus dados agregados, deverá escolher Habilitado para habilitar a desagregação de vários registros.

O fluxo do Firehose realiza as etapas de processamento a seguir, nesta ordem: desagregação de KPL (protobuf), desagregação de JSON ou delimitador, processamento de Lambda, particionamento de dados, conversão de formato dos dados e entrega ao Amazon S3.

Tipo de desagregação de vários registros

Se você habilitou a desagregação de vários registros, deverá especificar o método a ser usado pelo Firehose para desagregar os dados. Use o menu suspenso para escolher JSON ou Delimitado.

Análise em linha

Esse é um dos mecanismos com suporte para o particionamento dinâmico dos dados vinculados ao Amazon S3. Para usar a análise em linha para fazer o particionamento dinâmico de dados, você deve especificar os parâmetros de registro de dados a serem usados como chaves de particionamento e fornecer um valor para cada chave de particionamento especificada. Escolha Habilitado para habilitar e configurar o particionamento em linha.

> **Important**
>
> Se você especificou uma função do AWS Lambda nas etapas acima para transformar os registros da fonte, poderá usar essa função para particionar dinamicamente os dados vinculados ao S3 e ainda poderá criar as chaves de particionamento com análise em linha. Com o particionamento dinâmico, você pode usar a análise em linha ou a função do AWS Lambda para criar as chaves de particionamento. Ou é possível usar a análise em linha e a função do AWS Lambda ao mesmo tempo para criar as chaves de particionamento.

Chaves de particionamento dinâmico

É possível usar os campos Chave e Valor para especificar os parâmetros de registro de dados a serem usados como chaves de particionamento dinâmico e consultas jq para gerar os valores das chaves de particionamento dinâmico. O Firehose oferece suporte somente ao jq 1.6. É possível especificar até 50 chaves de particionamento dinâmico. Você deve inserir

expressões jq válidas para os valores de chave de particionamento dinâmico para configurar com êxito o particionamento dinâmico para o fluxo do Firehose.

Prefixo de bucket do S3

Ao habilitar e configurar o particionamento dinâmico, você deve especificar os prefixos de bucket do S3 para os quais o Amazon Data Firehose deverá entregar os dados particionados.

Para que o particionamento dinâmico seja configurado corretamente, o número dos prefixos de bucket do S3 deve ser idêntico ao número de chaves de particionamento especificadas.

É possível particionar os dados da fonte com análise em linha ou com a função do Lambda especificada AWS . Se você especificou uma função do AWS Lambda para criar chaves de particionamento para os dados da fonte, deverá digitar manualmente os valores dos prefixos de bucket do S3 usando o seguinte formato: "lambda:keyID". partitionKeyFrom Se for usar análise em linha para especificar as chaves de particionamento para os dados da fonte, será possível digitar manualmente os valores de visualização de buckets do S3 usando o formato a seguir: "partitionKeyFromquery:keyID" ou escolher o botão Aplicar chaves de particionamento dinâmico para usar os pares de chave/valor de particionamento dinâmico para gerar automaticamente os prefixos de bucket do S3. Ao particionar os dados com análise em linha ou com o AWS Lambda, você também pode usar os seguintes formulários de expressão no prefixo de bucket do S3:! {namespace:value}, em que o namespace pode ser Query ou Lambda. partitionKeyFrom partitionKeyFrom

Fuso horário do bucket do S3 e do prefixo de saída de erro do S3

Escolha um fuso horário que você deseja usar para data e hora em prefixos personalizados para objetos do Amazon S3. Por padrão, o Firehose adiciona um prefixo de hora em UTC. É possível alterar o fuso horário usado nos prefixos do S3 se quiser usar um fuso horário diferente.

Sugestões de armazenamento em buffer

O Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços.

Compactação do S3

Escolha a compactação de dados GZIP, Snappy, Zip ou Snappy compatível com Hadoop ou nenhuma compactação de dados. As compactações Snappy, Zip e Snappy compatível

com Hadoop não estão disponíveis para fluxos do Firehose com o Amazon Redshift como destino.

Formato de extensão de arquivo do S3 (opcional)

Especifique um formato de extensão de arquivo para objetos entregues ao bucket de destino do Amazon S3. Se você habilitar esse atributo, a extensão de arquivo especificada substituirá as extensões de arquivo padrão anexadas pelos atributos de conversão de formato de dados ou compactação do S3, como .parquet ou .gz. Verifique se você configurou a extensão de arquivo correta ao usar esse atributo com a conversão de formato de dados ou a compactação do S3. A extensão do arquivo deve começar com um ponto (.) e pode conter os caracteres permitidos: 0-9a-z!-_.*' (). A extensão do arquivo não pode exceder 128 caracteres.

Criptografia do S3

O Firehose oferece suporte à criptografia do lado do servidor do Amazon S3 AWS Key Management Service com o (SSE-KMS) para criptografar os dados entregues no Amazon S3. É possível escolher usar o tipo de criptografia padrão especificado no bucket do S3 de destino ou criptografar com uma chave da lista de AWS KMS chaves do que você possui. Se você criptografar os dados com AWS KMS chaves do, poderá usar a chave AWS gerenciada pela padrão (aws/s3) ou uma chave gerenciada pelo cliente. Para obter mais informações, consulte Proteção de dados usando criptografia do lado do servidor com chaves AWS gerenciadas pelo KMS (SSE-KMS).

Definições de configurações de destino para tabelas do Apache Iceberg

O Firehose oferece suporte ao Apache Iceberg Tables como destino em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia).

Para obter mais informações sobre as tabelas do Apache Iceberg como seu destino, consulte Entrega de dados às tabelas do Apache Iceberg com o Amazon Data Firehose.

Definições de configurações de destino para o Amazon Redshift

Esta seção descreve as configurações para usar o Amazon Redshift como destino do seu fluxo do Firehose.

Escolha um dos procedimentos a seguir dependendo de você ter um cluster provisionado pelo Amazon Redshift ou um grupo de trabalho do Amazon Redshift Sem Servidor.

- Cluster provisionado do Amazon Redshift
- Definições de configurações de destino para grupo de trabalho do Amazon Redshift sem servidor

> **Note**
>
> O Firehose não pode gravar em clusters do Amazon Redshift que usem encaminhamento de VPC aprimorado.

Cluster provisionado do Amazon Redshift

Esta seção descreve as configurações para usar o cluster provisionado do Amazon Redshift como destino do seu fluxo do Firehose.

- Insira valores para os seguintes campos:

  Cluster

  O cluster do Amazon Redshift no qual os dados do bucket do S3 são copiados. Configure o cluster do Amazon Redshift para que esteja acessível publicamente e desbloqueie os endereços IP do Amazon Data Firehose IP. Para obter mais informações, consulte Conceder ao Firehose acesso a um destino do Amazon Redshift .

  Autenticação

  É possível optar por inserir o nome de usuário/senha diretamente ou recuperar o segredo do AWS Secrets Manager para acessar o cluster do Amazon Redshift.

  - Nome do usuário

    Especifique um usuário do Amazon Redshift que tenha permissões para acessar o cluster do Amazon Redshift. Esse usuário deve ter a permissão `INSERT` do Amazon Redshift para copiar dados do bucket do S3 no cluster do Amazon Redshift.

  - Senha

    Especifique a senha do usuário com permissões para acessar o cluster.

  - Secret

    Selecione um segredo do AWS Secrets Manager que contenha as credenciais para o cluster do Amazon Redshift. Se você não vir o segredo na lista suspensa, crie um no AWS

Secrets Manager para suas credenciais do Amazon Redshift. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

Banco de dados

O banco de dados do Amazon Redshift no qual os dados são copiados.

Tabela

A tabela do Amazon Redshift no qual os dados são copiados.

Columns

(Opcional) As colunas específicas da tabela na qual os dados serão copiados. Use essa opção se o número de colunas definidas nos objetos do Amazon S3 for menor que o número de colunas na tabela do Amazon Redshift.

Destino intermediário do S3

O Firehose entrega os dados ao bucket do S3 primeiro e, em seguida, emite um comando COPY do Amazon Redshift para carregar os dados no cluster do Amazon Redshift. Especifique um bucket do S3 do qual você seja proprietário; os dados em streaming serão entregues nesse bucket. Crie um novo bucket do S3 ou escolha um bucket já existente do qual você seja proprietário.

O Firehose não exclui os dados do bucket do S3 depois de carregá-los no cluster do Amazon Redshift. É possível gerenciar os dados no bucket do S3 usando uma configuração de ciclo de vida. Para obter mais informações, consulte Gerenciamento do ciclo de vida de objetos no Guia do usuário do Amazon Simple Storage Service.

Prefixo de bucket intermediário do S3

(Opcional) Para usar o prefixo padrão para objetos do Amazon S3, deixe esta opção em branco. O Firehose usa automaticamente um prefixo no formato de hora UTC "`YYYY/MM/dd/HH`" para objetos entregues ao Amazon S3. É possível adicionar ao início deste prefixo. Para obter mais informações, consulte Configuração de formato de nome de objeto do Amazon S3.

Opções do COPY

Parâmetros que podem ser especificados no comando COPY do Amazon Redshift. Eles podem ser necessários para a configuração. Por exemplo, "`GZIP`" é necessário se a compactação de dados do Amazon S3 estiver ativada. "`REGION`" será necessário se o bucket do S3 não estiver na mesma AWS região da que o cluster do Amazon Redshift. Para obter

mais informações, consulte COPY no Guia do desenvolvedor de banco de dados do Amazon Redshift.

COPY command

O comando COPY do Amazon Redshift. Para obter mais informações, consulte COPY no Guia do desenvolvedor de banco de dados do Amazon Redshift.

Período de novas tentativas

Período (0 a 7.200 segundos) para o Firehose fazer nova tentativa se o comando COPY dos dados no cluster do Amazon Redshift falhar. O Firehose faz uma nova tentativa a cada 5 minutos, até que a o período de novas tentativas termine. Se você definir o período de novas tentativas como 0 (zero) segundos, o Firehose não fará novas tentativas após uma falha no comando COPY.

Sugestões de armazenamento em buffer

O Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços.

Compactação do S3

Escolha a compactação de dados GZIP, Snappy, Zip ou Snappy compatível com Hadoop ou nenhuma compactação de dados. As compactações Snappy, Zip e Snappy compatível com Hadoop não estão disponíveis para fluxos do Firehose com o Amazon Redshift como destino.

Formato de extensão de arquivo do S3 (opcional)

Formato de extensão de arquivo S3 (opcional): especifique um formato de extensão de arquivo para objetos entregues ao bucket de destino do Amazon S3. Se você habilitar esse atributo, a extensão de arquivo especificada substituirá as extensões de arquivo padrão anexadas pelos atributos de conversão de formato de dados ou compactação do S3, como .parquet ou .gz. Verifique se você configurou a extensão de arquivo correta ao usar esse atributo com a conversão de formato de dados ou a compactação do S3. A extensão do arquivo deve começar com um ponto (.) e pode conter os caracteres permitidos: 0-9a-z!-_.*' (). A extensão do arquivo não pode exceder 128 caracteres.

Criptografia do S3

O Firehose oferece suporte à criptografia do lado do servidor do Amazon S3 AWS Key Management Service com o (SSE-KMS) para criptografar os dados entregues no Amazon S3. É possível escolher usar o tipo de criptografia padrão especificado no bucket do S3 de destino ou criptografar com uma chave da lista de AWS KMS chaves do que você possui. Se você criptografar os dados com AWS KMS chaves do, poderá usar a chave AWS gerenciada pela padrão (aws/s3) ou uma chave gerenciada pelo cliente. Para obter mais informações, consulte Proteção de dados usando criptografia do lado do servidor com chaves AWS gerenciadas pelo KMS (SSE-KMS).

Definições de configurações de destino para grupo de trabalho do Amazon Redshift sem servidor

Esta seção descreve as configurações para usar o grupo de trabalho do Amazon Redshift sem servidor como destino do fluxo do Firehose.

- Insira valores para os seguintes campos:

  Workgroup name (Nome do grupo de trabalho)

  O grupo de trabalho do Amazon Redshift Sem Servidor ou um grupo de trabalho do Amazon Redshift no qual os dados do bucket do S3 são copiados. Configure o grupo de trabalho do Amazon Redshift sem servidor para ser acessível publicamente e desbloquear os endereços IP do Firehose. Para obter mais informações, consulte a seção Conectar-se a uma instância do Amazon Redshift Sem Servidor acessível publicamente em Conectar-se ao Amazon Redshift Sem Servidor e também Conceder ao Firehose acesso a um destino do Amazon Redshift .

  Autenticação

  É possível optar por inserir o nome de usuário/senha diretamente ou recuperar o segredo do AWS Secrets Manager para acessar o grupo de trabalho do Amazon Redshift sem servidor.

  - Nome do usuário

    Especifique um usuário do Amazon Redshift com permissões para acessar o grupo de trabalho do Amazon Redshift sem servidor. Esse usuário deve ter a permissão `INSERT` do

Amazon Redshift para copiar dados do bucket do S3 para o grupo de trabalho do Amazon Redshift Sem Servidor.

- Senha

  Especifique a senha do usuário com permissões para acessar o grupo de trabalho do Amazon Redshift sem servidor.

- Secret

  Selecione um segredo do AWS Secrets Manager que contenha as credenciais para o grupo de trabalho d Amazon Redshift Sem Servidor. Se você não vir o segredo na lista suspensa, crie um no AWS Secrets Manager para suas credenciais do Amazon Redshift. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

Banco de dados

O banco de dados do Amazon Redshift no qual os dados são copiados.

Tabela

A tabela do Amazon Redshift no qual os dados são copiados.

Columns

(Opcional) As colunas específicas da tabela na qual os dados serão copiados. Use essa opção se o número de colunas definidas nos objetos do Amazon S3 for menor que o número de colunas na tabela do Amazon Redshift.

Destino intermediário do S3

O Amazon Data Firehose entrega os dados ao bucket do S3 primeiro e, em seguida, emite um comando COPY do Amazon Redshift para carregar os dados no grupo de trabalho do Amazon Redshift sem servidor. Especifique um bucket do S3 do qual você seja proprietário; os dados em streaming serão entregues nesse bucket. Crie um novo bucket do S3 ou escolha um bucket já existente do qual você seja proprietário.

O Firehose não exclui os dados do bucket do S3 depois de carregá-los no grupo de trabalho do Amazon Redshift sem servidor. É possível gerenciar os dados no bucket do S3 usando uma configuração de ciclo de vida. Para obter mais informações, consulte Gerenciamento do ciclo de vida de objetos no Guia do usuário do Amazon Simple Storage Service.

Prefixo de bucket intermediário do S3

(Opcional) Para usar o prefixo padrão para objetos do Amazon S3, deixe esta opção em branco. O Firehose usa automaticamente um prefixo no formato de hora UTC "YYYY/MM/dd/HH" para objetos entregues ao Amazon S3. É possível adicionar ao início deste prefixo. Para obter mais informações, consulte Configuração de formato de nome de objeto do Amazon S3.

Opções do COPY

Parâmetros que podem ser especificados no comando COPY do Amazon Redshift. Eles podem ser necessários para a configuração. Por exemplo, "GZIP" é necessário se a compactação de dados do Amazon S3 estiver ativada. "REGION" será necessário se o bucket do S3 não estiver na mesma AWS região da que o grupo de trabalho d Amazon Redshift Sem Servidor. Para obter mais informações, consulte COPY no Guia do desenvolvedor de banco de dados do Amazon Redshift.

COPY command

O comando COPY do Amazon Redshift. Para obter mais informações, consulte COPY no Guia do desenvolvedor de banco de dados do Amazon Redshift.

Período de novas tentativas

Período (0 a 7.200 segundos) para o Firehose fazer nova tentativa se o comando COPY dos dados no grupo de trabalho do Amazon Redshift sem servidor falhar. O Firehose faz uma nova tentativa a cada 5 minutos, até que a o período de novas tentativas termine. Se você definir o período de novas tentativas como 0 (zero) segundos, o Firehose não fará novas tentativas após uma falha no comando COPY.

Sugestões de armazenamento em buffer

O Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços.

Compactação do S3

Escolha a compactação de dados GZIP, Snappy, Zip ou Snappy compatível com Hadoop ou nenhuma compactação de dados. As compactações Snappy, Zip e Snappy compatível com Hadoop não estão disponíveis para fluxos do Firehose com o Amazon Redshift como destino.

Formato de extensão de arquivo do S3 (opcional)

Formato de extensão de arquivo S3 (opcional): especifique um formato de extensão de arquivo para objetos entregues ao bucket de destino do Amazon S3. Se você habilitar esse atributo, a extensão de arquivo especificada substituirá as extensões de arquivo padrão anexadas pelos atributos de conversão de formato de dados ou compactação do S3, como .parquet ou .gz. Verifique se você configurou a extensão de arquivo correta ao usar esse atributo com a conversão de formato de dados ou a compactação do S3. A extensão do arquivo deve começar com um ponto (.) e pode conter os caracteres permitidos: 0-9a-z!-_.*' (). A extensão do arquivo não pode exceder 128 caracteres.

Criptografia do S3

O Firehose oferece suporte à criptografia do lado do servidor do Amazon S3 AWS Key Management Service com o (SSE-KMS) para criptografar os dados entregues no Amazon S3. É possível escolher usar o tipo de criptografia padrão especificado no bucket do S3 de destino ou criptografar com uma chave da lista de AWS KMS chaves do que você possui. Se você criptografar os dados com AWS KMS chaves do, poderá usar a chave AWS gerenciada pela padrão (aws/s3) ou uma chave gerenciada pelo cliente. Para obter mais informações, consulte Protecting Data Using Server-Side Encryption with AWS KMS-Managed Keys (SSE-KMS).

Definições de as configurações de destino para o OpenSearch Service

Esta seção descreve as opções para uso do OpenSearch Serviço como destino.

- Insira valores para os seguintes campos:

  OpenSearch Domínio do serviço

  O domínio do OpenSearch serviço ao qual os dados serão entregues.

  Índice

  O nome do índice do OpenSearch serviço a ser usado ao indexar dados no cluster OpenSearch de serviços.

Index rotation

Escolha se deve feito o rodízio OpenSearch de índices de serviços e com que frequência. Se o rodízio de índices estiver habilitado, o Amazon Data Firehose adicionará o carimbo de data/hora correspondente ao nome de índice especificado e fará o rodízio. Para obter mais informações, consulte Configurar a rotação do índice para o OpenSearch serviço.

Tipo

O nome do tipo de OpenSearch serviço a ser usado ao indexar dados no cluster OpenSearch de serviços. Para o Elasticsearch 7.x e OpenSearch 1.x, pode haver apenas um tipo por índice. Se você tentar especificar um novo tipo para um índice existente que já tem outro tipo, o Firehose retornará um erro durante o runtime.

Para o Elasticsearch 7.x, deixe esse campo vazio.

Período de novas tentativas

Período durante o qual o Firehose fará novas tentativas se uma solicitação de índice falhar. OpenSearch Para o período de novas tentativas, é possível definir qualquer valor entre 0 e 7.200 segundos. A o período de novas tentativas padrão é de 300 segundos. Firehose fará uma nova tentativa várias vezes com o período de novas tentativas até que a o período de novas tentativas expire.

Depois que o período de novas tentativas expirar, o Firehose entregará os dados para a fila de mensagens não entregues (DLQ), um bucket de erros S3 configurado. Para dados entregues ao DLQ, você precisa redirecionar os dados do bucket de erros do S3 configurado para o destino. OpenSearch

Se você quiser impedir que o fluxo do Firehose entregue dados à DLQ devido ao tempo de inatividade ou à manutenção dos OpenSearch clusters, é possível configurar a duração da nova tentativa para um valor maior em segundos. É possível aumentar o valor da duração da nova tentativa acima para 7.200 segundos entrando em contato com o suporte da AWS.

Tipo DocumentID

Indica o método para configurar o ID do documento. Os métodos com suporte são ID do documento gerado pelo Firehose e ID do documento gerado pelo OpenSearch serviço. ID do documento gerado pelo Firehose é a opção padrão quando o valor do ID do documento não está definido. OpenSearch O ID do documento gerado pelo serviço é a opção recomendada porque é compatível com operações de gravação intensiva, inclusive análise de log e

observabilidade, consumindo menos recursos de CPU no domínio do OpenSearch Serviço e, portanto, resultando em melhor performance.

Destination VPC connectivity (Conectividade da VPC de destino)

Se o domínio do OpenSearch Service estiver em uma VPC privada, use esta seção para especificar essa VPC. Especifique também as sub-redes e os subgrupos que você deseja que o Amazon Data Firehose use quando enviar dados para o domínio do serviço. OpenSearch Você pode usar os mesmos grupos de segurança que o domínio do OpenSearch Service está usando. Se você especificar outros grupos de segurança, certifique-se de que eles permitam tráfego HTTPS de saída para o grupo de segurança do domínio do OpenSearch Serviço. Certifique-se também de que o grupo de segurança do domínio do OpenSearch Service permita o tráfego HTTPS dos grupos de segurança que você especificou ao configurar o fluxo do Firehose. Se você usar o mesmo grupo de segurança para o fluxo do Firehose e para o domínio do OpenSearch Service, certifique-se de que a regra de entrada do grupo de segurança permita tráfego de HTTPS. Para obter mais informações sobre regras de grupo de segurança, consulte Regras de grupo de segurança na documentação da Amazon VPC.

> **Important**
>
> Ao especificar sub-redes para entregar dados ao destino em uma VPC privada, verifique se você tem um número suficiente de endereços IP livres nas sub-redes escolhidas. Se não houver endereços IP gratuitos disponíveis em uma sub-rede especificada, o Firehose não poderá criar ou ENIs adicionar dados para a VPC privada, e a entrega será degradada ou falhará.

Sugestões de buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços.

Definições de as configurações de destino para o lado do OpenSearch servidor

Esta seção descreve as opções para uso do lado do OpenSearch servidor como destino.

- Insira valores para os seguintes campos:

  OpenSearch Coleção sem servidor

  O endpoint para um grupo de índices OpenSearch Sem Servidor ao qual os dados são entregues.

  Índice

  O nome do índice OpenSearch Sem Servidor a ser usado ao indexar dados para a coleção Sem Servidor. OpenSearch

  Destination VPC connectivity (Conectividade da VPC de destino)

  Se a coleção OpenSearch Sem Servidor estiver em uma VPC privada, use esta seção para especificar essa VPC. Especifique também as sub-redes e os subgrupos que você deseja que o Amazon Data Firehose use quando enviar dados para a coleção do Sem Servidor. OpenSearch

  > **Important**
  >
  > Ao especificar sub-redes para entregar dados ao destino em uma VPC privada, verifique se você tem um número suficiente de endereços IP livres nas sub-redes escolhidas. Se não houver endereços IP gratuitos disponíveis em uma sub-rede especificada, o Firehose não poderá criar ou ENIs adicionar dados para a VPC privada, e a entrega será degradada ou falhará.

  Período de novas tentativas

  Período durante o qual o Firehose fará novas tentativas se uma solicitação de índice ao lado do servidor falhar OpenSearch . Para o período de novas tentativas, é possível definir qualquer valor entre 0 e 7.200 segundos. A o período de novas tentativas padrão é de 300 segundos. Firehose fará uma nova tentativa várias vezes com o período de novas tentativas até que a o período de novas tentativas expire.

  Depois que o período de novas tentativas expirar, o Firehose entregará os dados para a fila de mensagens não entregues (DLQ), um bucket de erros S3 configurado. Para dados entregues ao DLQ, você precisa redirecionar os dados do bucket de erros do S3 configurado para OpenSearch o destino sem servidor.

Se você quiser impedir que o fluxo do Firehose entregue dados à DLQ devido ao tempo de inatividade ou à manutenção dos clusters OpenSearch sem servidor, é possível configurar a duração da nova tentativa para um valor maior em segundos. É possível aumentar o valor da duração da nova tentativa acima para 7.200 segundos entrando em contato com o suporte da AWS.

Sugestões de buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços.

Definição das configurações de destino para Endpoint de HTTP

Esta seção descreve as opções para usar o endpoint de HTTP como destino.

> **Important**
>
> Se você escolher um endpoint de HTTP como destino, revise e siga as instruções em Noções básicas das especificações de solicitação e resposta de entrega de endpoint de HTTP.

- Forneça os valores para os seguintes campos:

  Nome do endpoint de HTTP - opcional

  Especifique um nome de usuário amigável para o endpoint de HTTP. Por exemplo, .`My HTTP Endpoint Destination`

  URL do endpoint de HTTP

  Especifique o URL para o endpoint de HTTP no formato a seguir: `https://xyz.httpendpoint.com`. A origem deve ser um URL HTTPS.

  Autenticação

  É possível optar por inserir a chave de acesso diretamente ou recuperar o segredo do AWS Secrets Manager para acessar o endpoint de HTTP.

- (Opcional) Chave de acesso

  Entre em contato com o proprietário do endpoint se você precisar obter a chave de acesso para permitir a entrega de dados do Firehose ao endpoint.

- Secret

  Selecione um segredo do AWS Secrets Manager que contenha a chave de acesso para o endpoint de HTTP. Se você não vir o segredo na lista suspensa, crie um no AWS Secrets Manager para a chave de acesso. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

Codificação de conteúdo

O Amazon Data Firehose usa a codificação de conteúdo para compactar o corpo de uma solicitação antes de enviá-la ao destino. Escolha GZIP ou Desabilitado para habilitar/desabilitar a codificação de conteúdo da solicitação.

Período de novas tentativas

Especifique por quanto tempo o Amazon Data Firehose fará novas tentativas de enviar os dados ao endpoint de HTTP selecionado.

Depois de enviar os dados, o Amazon Data Firehose primeiro esperará por uma confirmação do endpoint de HTTP. Se ocorrer um erro ou se a confirmação não chegar dentro do tempo limite para confirmação, o Amazon Data Firehose iniciará o contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Amazon Data Firehose considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Toda vez que o Amazon Data Firehose envia dados para o endpoint de HTTP (seja a tentativa inicial ou uma nova tentativa), ele reinicia o contador de tempo limite para confirmação e aguarda uma confirmação do endpoint de HTTP.

Mesmo que o período de novas tentativas expire, o Amazon Data Firehose ainda aguardará a confirmação até recebê-la ou até que o período de tempo limite para confirmação seja atingido. Se o tempo limite para confirmação expirar, o Amazon Data Firehose determinará se ainda resta algum tempo no contador de novas tentativas. Se houver tempo restante, ele tentará executar novamente e repetirá a lógica até receber um reconhecimento ou determinará que o tempo de tentar novamente expirou.

Se você não quiser que o Amazon Data Firehose tente enviar os dados novamente, defina esse valor como 0.

Parâmetros: opcional

O Amazon Data Firehose inclui esses pares de valores-chave em toda chamada HTTP. Esses parâmetros ajudam a identificar e organizar os destinos.

Sugestões de armazenamento em buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços.

> **Important**
>
> Para destinos de endpoint de HTTP, se você estiver vendo códigos de resposta 413 do endpoint de destino no CloudWatch Logs, reduza o tamanho da sugestão de buffer no seu fluxo do Firehose e tente novamente.

Definições de configurações de destino para o Datadog

Esta seção descreve as opções para usar o Datadog como destino. Para obter mais informações sobre o Datadog, consulte https://docs.datadoghq.com/integrations/ amazon_web_services/.

- Forneça os valores para os campos a seguir.

  URL do endpoint de HTTP

  Escolha para onde você deseja enviar dados dentre uma das opções a seguir no menu suspenso.

  - Registros do Datadog - US1
  - Registros do Datadog - US3
  - Registros do Datadog - US5
  - Registros do Datadog - AP1
  - Logs do Datadog: EU
  - Logs do Datadog: GOV

- Métricas do Datadog - EUA
- Métricas do Datadog - US5
- Métricas do Datadog - AP1
- Métricas do Datadog: EU
- Configurações do Datadog - US1
- Configurações do Datadog - US3
- Configurações do Datadog - US5
- Configurações do Datadog - AP1
- Configurações do Datadog - UE
- Configurações do Datadog - GOV EUA

Autenticação

É possível optar por inserir a chave da API diretamente ou recuperar o segredo do AWS Secrets Manager para acessar o Datadog.

- Chave de API

  Entre em contato com o Datadog para obter a chave de API necessária para permitir a entrega de dados do Firehose a esse endpoint.

- Secret

  Selecione um segredo do AWS Secrets Manager que contenha a chave da API do Datadog. Se você não vir o segredo na lista suspensa, crie um no AWS Secrets Manager. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

Codificação de conteúdo

O Amazon Data Firehose usa a codificação de conteúdo para compactar o corpo de uma solicitação antes de enviá-la ao destino. Escolha GZIP ou Desabilitado para habilitar/desabilitar a codificação de conteúdo da solicitação.

Período de novas tentativas

Especifique por quanto tempo o Amazon Data Firehose fará novas tentativas de enviar os dados ao endpoint de HTTP selecionado.

Depois de enviar os dados, o Amazon Data Firehose primeiro esperará por uma confirmação do endpoint de HTTP. Se ocorrer um erro ou se a confirmação não chegar dentro do tempo limite para confirmação, o Amazon Data Firehose iniciará o contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Amazon Data Firehose considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Toda vez que o Amazon Data Firehose envia dados para o endpoint de HTTP (seja a tentativa inicial ou uma nova tentativa), ele reinicia o contador de tempo limite para confirmação e aguarda uma confirmação do endpoint de HTTP.

Mesmo que o período de novas tentativas expire, o Amazon Data Firehose ainda aguardará a confirmação até recebê-la ou até que o período de tempo limite para confirmação seja atingido. Se o tempo limite para confirmação expirar, o Amazon Data Firehose determinará se ainda resta algum tempo no contador de novas tentativas. Se houver tempo restante, ele tentará executar novamente e repetirá a lógica até receber um reconhecimento ou determinará que o tempo de tentar novamente expirou.

Se você não quiser que o Amazon Data Firehose tente enviar os dados novamente, defina esse valor como 0.

Parâmetros: opcional

O Amazon Data Firehose inclui esses pares de valores-chave em toda chamada HTTP. Esses parâmetros ajudam a identificar e organizar os destinos.

Sugestões de armazenamento em buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços.

Definições de configurações de destino para o Honeycomb

Esta seção descreve as opções para usar o Honeycomb como destino. Para obter mais informações sobre o Honeycomb, consulte https://docs.honeycomb. -cloudwatch-metrics/. io/getting-data-in/metrics/aws

- Forneça os valores para os seguintes campos:

Endpoint do Honeycomb Kinesis

Especifique o URL para o endpoint de HTTP no formato a seguir: https://api.honeycom b.io/1/kinesis_events/ {{dataset}}

Autenticação

É possível optar por inserir a chave da API diretamente ou recuperar o segredo do AWS Secrets Manager para acessar o Honeycomb.

- Chave de API

  Entre em contato com o Honeycomb para obter a chave de API necessária para permitir a entrega de dados do Firehose a esse endpoint.

- Secret

  Selecione um segredo do AWS Secrets Manager que contenha a chave da API do Honeycomb. Se você não vir o segredo na lista suspensa, crie um no AWS Secrets Manager. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

Codificação de conteúdo

O Amazon Data Firehose usa a codificação de conteúdo para compactar o corpo de uma solicitação antes de enviá-la ao destino. Escolha GZIP para habilitar a codificação de conteúdo da solicitação. Essa é a opção recomendada quando o destino é o Honeycomb.

Período de novas tentativas

Especifique por quanto tempo o Amazon Data Firehose fará novas tentativas de enviar os dados ao endpoint de HTTP selecionado.

Depois de enviar os dados, o Amazon Data Firehose primeiro esperará por uma confirmação do endpoint de HTTP. Se ocorrer um erro ou se a confirmação não chegar dentro do tempo limite para confirmação, o Amazon Data Firehose iniciará o contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Amazon Data Firehose considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Toda vez que o Amazon Data Firehose envia dados para o endpoint de HTTP (seja a tentativa inicial ou uma nova tentativa), ele reinicia o contador de tempo limite para confirmação e aguarda uma confirmação do endpoint de HTTP.

Mesmo que o período de novas tentativas expire, o Amazon Data Firehose ainda aguardará a confirmação até recebê-la ou até que o período de tempo limite para confirmação seja atingido. Se o tempo limite para confirmação expirar, o Amazon Data Firehose determinará se ainda resta algum tempo no contador de novas tentativas. Se houver tempo restante, ele tentará executar novamente e repetirá a lógica até receber um reconhecimento ou determinará que o tempo de tentar novamente expirou.

Se você não quiser que o Amazon Data Firehose tente enviar os dados novamente, defina esse valor como 0.

Parâmetros: opcional

O Amazon Data Firehose inclui esses pares de valores-chave em toda chamada HTTP. Esses parâmetros ajudam a identificar e organizar os destinos.

Sugestões de armazenamento em buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços.

Definição de as configurações de destino para o Coralogix

Esta seção descreve as opções para usar o Coralogix como destino. Para obter mais informações sobre o Coralogix, consulte Conceitos básicos do Coralogix.

- Forneça os valores para os seguintes campos:

  URL do endpoint de HTTP

  Escolha o URL do endpoint de HTTP entre as opções a seguir no menu suspenso:

  - Coralogix - EUA
  - Coralogix - SINGAPURA
  - Coralogix - IRLANDA

- Coralogix - ÍNDIA
- Coralogix - ESTOCOLMO

Autenticação

É possível optar por inserir a chave privada diretamente ou recuperar o segredo do AWS Secrets Manager para acessar o Coralogix.

- Chave privada

  Entre em contato com o Coralogix para obter a chave privada necessária para permitir a entrega de dados do Firehose a esse endpoint.

- Secret

  Selecione um segredo do AWS Secrets Manager que contenha a chave privada do Coralogix. Se você não vir o segredo na lista suspensa, crie um no AWS Secrets Manager. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

Codificação de conteúdo

O Amazon Data Firehose usa a codificação de conteúdo para compactar o corpo de uma solicitação antes de enviá-la ao destino. Escolha GZIP para habilitar a codificação de conteúdo da solicitação. Essa é a opção recomendada quando o destino é o Coralogix.

Período de novas tentativas

Especifique por quanto tempo o Amazon Data Firehose fará novas tentativas de enviar os dados ao endpoint de HTTP selecionado.

Depois de enviar os dados, o Amazon Data Firehose primeiro esperará por uma confirmação do endpoint de HTTP. Se ocorrer um erro ou se a confirmação não chegar dentro do tempo limite para confirmação, o Amazon Data Firehose iniciará o contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Amazon Data Firehose considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Toda vez que o Amazon Data Firehose envia dados para o endpoint de HTTP (seja a tentativa inicial ou uma nova tentativa), ele reinicia o contador de tempo limite para confirmação e aguarda uma confirmação do endpoint de HTTP.

Mesmo que o período de novas tentativas expire, o Amazon Data Firehose ainda aguardará a confirmação até recebê-la ou até que o período de tempo limite para confirmação seja atingido. Se o tempo limite para confirmação expirar, o Amazon Data Firehose determinará se ainda resta algum tempo no contador de novas tentativas. Se houver tempo restante, ele tentará executar novamente e repetirá a lógica até receber um reconhecimento ou determinará que o tempo de tentar novamente expirou.

Se você não quiser que o Amazon Data Firehose tente enviar os dados novamente, defina esse valor como 0.

Parâmetros: opcional

O Amazon Data Firehose inclui esses pares de valores-chave em toda chamada HTTP. Esses parâmetros ajudam a identificar e organizar os destinos.

- applicationName: o ambiente em que você está executando o Data Firehose
- subsystemName: o nome da integração do Data Firehose
- computerName: o nome do fluxo do Firehose em uso

Sugestões de armazenamento em buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia com base no provedor de serviços.

Definições de configurações de destino para o Dynatrace

Esta seção descreve as opções para usar o Dynatrace como destino. Para obter mais informações, consulte https://www.dynatrace.com/support/help/technology-support/cloud-platforms/amazon-web-services/integrations/cloudwatch-metric-streams/.

- Escolha as opções para usar o Dynatrace como destino de seu fluxo do Firehose.

  Tipo de ingestão

  Escolha se você deseja entregar Métricas ou Logs (padrão) no Dynatrace para análise e processamento adicionais.

URL do endpoint de HTTP

Escolha o URL do endpoint de HTTP (Dynatrace EUA, Dynatrace UE ou Dynatrace Global) no menu suspenso.

Autenticação

É possível optar por inserir o token da API diretamente ou recuperar o segredo do AWS Secrets Manager para acessar o Dynatrace.

- Token de API

  Gere o token de API necessário para habilitar a entrega de dados do Dynatrace a esse endpoint a partir do Firehose. Para obter mais informações, consulte API do Dynatrace - Tokens e autenticação.

- Secret

  Selecione um segredo do AWS Secrets Manager que contenha o token da API do Dynatrace. Se você não vir o segredo na lista suspensa, crie um no AWS Secrets Manager. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

URL da API

Forneça o URL da API do ambiente do Dynatrace.

Codificação de conteúdo

Escolha se você deseja habilitar a codificação de conteúdo para compactar o corpo da solicitação. O Amazon Data Firehose usa a codificação de conteúdo para compactar o corpo de uma solicitação antes de enviá-la ao destino. Quando habilitado, o conteúdo será compactado no formato GZIP.

Período de novas tentativas

Especifique por quanto tempo o Firehose faz novas tentativas de enviar os dados ao endpoint de HTTP selecionado.

Depois de enviar os dados, o Firehose primeiro esperará por uma confirmação do endpoint de HTTP. Se ocorrer um erro ou se a confirmação não chegar dentro do tempo limite de confirmação, o Firehose iniciará contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Firehose

considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Toda vez que o Firehose envia dados para o endpoint de HTTP, seja a tentativa inicial ou uma nova tentativa, ele reinicia o contador de tempo limite para confirmação e aguarda uma confirmação do endpoint de HTTP.

Mesmo que o período de novas tentativas expire, o Firehose ainda aguardará a confirmação até recebê-la ou até que o tempo limite de espera de confirmação seja atingido. Se o tempo limite para confirmação expirar, o Firehose determinará se ainda resta algum tempo no contador de novas tentativas. Se houver tempo restante, ele tentará executar novamente e repetirá a lógica até receber um reconhecimento ou determinará que o tempo de tentar novamente expirou.

Se você não quiser que o Firehose tente enviar os dados novamente, defina esse valor como 0.

Parâmetros: opcional

O Amazon Data Firehose inclui esses pares de valores-chave em toda chamada HTTP. Esses parâmetros ajudam a identificar e organizar os destinos.

Sugestões de armazenamento em buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. As sugestões de buffer incluem o tamanho e o intervalo do buffer para seus fluxos. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços.

Definições de as configurações de destino para LogicMonitor

Esta seção descreve as opções para usar o LogicMonitor como destino. Para obter mais informações, consulte https://www.logicmonitor.com.

- Forneça os valores para os seguintes campos:

  URL do endpoint de HTTP

  Especifique o URL para o endpoint de HTTP no formato a seguir.

```
https://ACCOUNT.logicmonitor.com
```

Autenticação

É possível optar por inserir a chave da API diretamente ou recuperar o segredo do AWS Secrets Manager para acessar LogicMonitor.

- Chave de API

  Entre em contato LogicMonitor para obter a chave de API necessária para permitir a entrega de dados do Firehose a esse endpoint.

- Secret

  Selecione um segredo do AWS Secrets Manager que contenha a chave da API do LogicMonitor. Se você não vir o segredo na lista suspensa, crie um no AWS Secrets Manager. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

Codificação de conteúdo

O Amazon Data Firehose usa a codificação de conteúdo para compactar o corpo de uma solicitação antes de enviá-la ao destino. Escolha GZIP ou Desabilitado para habilitar/desabilitar a codificação de conteúdo da solicitação.

Período de novas tentativas

Especifique por quanto tempo o Amazon Data Firehose fará novas tentativas de enviar os dados ao endpoint de HTTP selecionado.

Depois de enviar os dados, o Amazon Data Firehose primeiro esperará por uma confirmação do endpoint de HTTP. Se ocorrer um erro ou se a confirmação não chegar dentro do tempo limite para confirmação, o Amazon Data Firehose iniciará o contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Amazon Data Firehose considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Toda vez que o Amazon Data Firehose envia dados para o endpoint de HTTP (seja a tentativa inicial ou uma nova tentativa), ele reinicia o contador de tempo limite para confirmação e aguarda uma confirmação do endpoint de HTTP.

Mesmo que o período de novas tentativas expire, o Amazon Data Firehose ainda aguardará a confirmação até recebê-la ou até que o período de tempo limite para confirmação seja atingido. Se o tempo limite para confirmação expirar, o Amazon Data Firehose determinará se ainda resta algum tempo no contador de novas tentativas. Se houver tempo restante, ele tentará executar novamente e repetirá a lógica até receber um reconhecimento ou determinará que o tempo de tentar novamente expirou.

Se você não quiser que o Amazon Data Firehose tente enviar os dados novamente, defina esse valor como 0.

Parâmetros: opcional

O Amazon Data Firehose inclui esses pares de valores-chave em toda chamada HTTP. Esses parâmetros ajudam a identificar e organizar os destinos.

Sugestões de armazenamento em buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços.

Definição das configurações de destino para o Logz.io

Esta seção descreve as opções para usar o Logz.io como destino. Para obter mais informações, consulte https://logz.io/.

> **Note**
>
> Na região da Europa (Milão), não há suporte para o Logz.io como destino do Amazon Data Firehose.

- Forneça os valores para os seguintes campos:

  URL do endpoint de HTTP

  Especifique o URL para o endpoint de HTTP no formato a seguir. A origem deve ser um URL HTTPS.

```
https://listener-aws-metrics-stream-<region>.logz.io/
```

Por exemplo

```
https://listener-aws-metrics-stream-us.logz.io/
```

Autenticação

É possível optar por inserir o token de envio diretamente ou recuperar o segredo do AWS Secrets Manager para acessar o Logz.io.

- Token de envio

  Entre em contato com o Logz.io obter o token de envio necessário para permitir a entrega de dados do Firehose a esse endpoint.

- Secret

  Selecione um segredo do AWS Secrets Manager que contenha o token de envio para o Logz.io. Se você não vir o segredo na lista suspensa, crie um no AWS Secrets Manager. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

Período de novas tentativas

Especifique por quanto tempo o Amazon Data Firehose fará novas tentativas de enviar os dados ao Logz.io.

Depois de enviar os dados, o Amazon Data Firehose primeiro esperará por uma confirmação do endpoint de HTTP. Se ocorrer um erro ou se a confirmação não chegar dentro do tempo limite para confirmação, o Amazon Data Firehose iniciará o contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Amazon Data Firehose considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Toda vez que o Amazon Data Firehose envia dados para o endpoint de HTTP (seja a tentativa inicial ou uma nova tentativa), ele reinicia o contador de tempo limite para confirmação e aguarda uma confirmação do endpoint de HTTP.

Mesmo que o período de novas tentativas expire, o Amazon Data Firehose ainda aguardará a confirmação até recebê-la ou até que o período de tempo limite para confirmação seja atingido. Se o tempo limite para confirmação expirar, o Amazon Data Firehose determinará se ainda resta algum tempo no contador de novas tentativas. Se houver tempo restante, ele tentará executar novamente e repetirá a lógica até receber um reconhecimento ou determinará que o tempo de tentar novamente expirou.

Se você não quiser que o Amazon Data Firehose tente enviar os dados novamente, defina esse valor como 0.

Parâmetros: opcional

O Amazon Data Firehose inclui esses pares de valores-chave em toda chamada HTTP. Esses parâmetros ajudam a identificar e organizar os destinos.

Sugestões de armazenamento em buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços.

Definições de configurações de destino para o MongoDB Cloud

Esta seção descreve as opções para usar a MongoDB Cloud como destino. Para obter mais informações, consulte https://www.mongodb.com.

- Forneça os valores para os seguintes campos:

  URL do webhook Realm do MongoDB

  Especifique o URL para o endpoint de HTTP no formato a seguir.

  ```
  https://webhooks.mongodb-realm.com
  ```

  A origem deve ser um URL HTTPS.

  Autenticação

  É possível optar por inserir a chave da API diretamente ou recuperar o segredo do AWS Secrets Manager para acessar o MongoDB Cloud Cloud.

- Chave de API

  Entre em contato com o MongoDB Cloud para obter a chave de API necessária para permitir a entrega de dados do Firehose a esse endpoint.

- Secret

  Selecione um segredo do AWS Secrets Manager que contenha a chave da API do MongoDB Cloud Cloud Cloud Cloud Cloud. Se você não vir o segredo na lista suspensa, crie um no AWS Secrets Manager. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

Codificação de conteúdo

O Amazon Data Firehose usa a codificação de conteúdo para compactar o corpo de uma solicitação antes de enviá-la ao destino. Escolha GZIP ou Desabilitado para habilitar/desabilitar a codificação de conteúdo da solicitação.

Período de novas tentativas

Especifique por quanto tempo o Amazon Data Firehose fará novas tentativas de enviar os dados ao provedor terceirizado selecionado.

Depois de enviar os dados, o Amazon Data Firehose primeiro esperará por uma confirmação do endpoint de HTTP. Se ocorrer um erro ou se a confirmação não chegar dentro do tempo limite para confirmação, o Amazon Data Firehose iniciará o contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Amazon Data Firehose considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Toda vez que o Amazon Data Firehose envia dados para o endpoint de HTTP (seja a tentativa inicial ou uma nova tentativa), ele reinicia o contador de tempo limite para confirmação e aguarda uma confirmação do endpoint de HTTP.

Mesmo que o período de novas tentativas expire, o Amazon Data Firehose ainda aguardará a confirmação até recebê-la ou até que o período de tempo limite para confirmação seja atingido. Se o tempo limite para confirmação expirar, o Amazon Data Firehose determinará se ainda resta algum tempo no contador de novas tentativas. Se houver tempo restante, ele tentará executar novamente e repetirá a lógica até receber um reconhecimento ou determinará que o tempo de tentar novamente expirou.

Se você não quiser que o Amazon Data Firehose tente enviar os dados novamente, defina esse valor como 0.

Sugestões de armazenamento em buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços.

Parâmetros: opcional

O Amazon Data Firehose inclui esses pares de valores-chave em toda chamada HTTP. Esses parâmetros ajudam a identificar e organizar os destinos.

Definições de configurações de destino para o New Relic

Esta seção descreve as opções para usar o New Relic como destino. Para obter mais informações, consulte https://newrelic.com.

- Forneça os valores para os seguintes campos:

  URL do endpoint de HTTP

  Escolha o URL do endpoint de HTTP entre as opções a seguir na lista suspensa.

  - Logs do New Relic - EUA
  - Métricas do New Relic - EUA
  - Métricas do New Relic - UE

  Autenticação

  É possível optar por inserir a chave da API diretamente ou recuperar o segredo do AWS Secrets Manager para acessar o New Relic.

  - Chave de API

    Insira sua chave de licença, que é uma sequência hexadecimal de 40 caracteres, nas configurações da One Account do New Relic. Essa chave de API é necessária para habilitar a entrega de dados do Firehose a esse endpoint.

  - Secret

Selecione um segredo do AWS Secrets Manager que contenha a chave da API do New Relic. Se você não vir o segredo na lista suspensa, crie um no AWS Secrets Manager. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

Codificação de conteúdo

O Amazon Data Firehose usa a codificação de conteúdo para compactar o corpo de uma solicitação antes de enviá-la ao destino. Escolha GZIP ou Desabilitado para habilitar/desabilitar a codificação de conteúdo da solicitação.

Período de novas tentativas

Especifique por quanto tempo o Amazon Data Firehose fará novas tentativas de enviar dados ao endpoint de HTTP do New Relic.

Depois de enviar os dados, o Amazon Data Firehose primeiro esperará por uma confirmação do endpoint de HTTP. Se ocorrer um erro ou se a confirmação não chegar dentro do tempo limite para confirmação, o Amazon Data Firehose iniciará o contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Amazon Data Firehose considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Toda vez que o Amazon Data Firehose envia dados para o endpoint de HTTP (seja a tentativa inicial ou uma nova tentativa), ele reinicia o contador de tempo limite para confirmação e aguarda uma confirmação do endpoint de HTTP.

Mesmo que o período de novas tentativas expire, o Amazon Data Firehose ainda aguardará a confirmação até recebê-la ou até que o período de tempo limite para confirmação seja atingido. Se o tempo limite para confirmação expirar, o Amazon Data Firehose determinará se ainda resta algum tempo no contador de novas tentativas. Se houver tempo restante, ele tentará executar novamente e repetirá a lógica até receber um reconhecimento ou determinará que o tempo de tentar novamente expirou.

Se você não quiser que o Amazon Data Firehose tente enviar os dados novamente, defina esse valor como 0.

Parâmetros: opcional

O Amazon Data Firehose inclui esses pares de valores-chave em toda chamada HTTP. Esses parâmetros ajudam a identificar e organizar os destinos.

Sugestões de armazenamento em buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços.

Definição de configurações de destino para o Snowflake

Esta seção descreve as opções para uso do Snowflake como destino.

> **Note**
>
> A integração do Firehose com o Snowflake está disponível no Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda), Leste dos EUA (Ohio), Ásia-Pacífico (Tóquio), Europa (Cingapura), Ásia-Pacífico (Seul) e Ásia-Pacífico (Sydney), Ásia-Pacífico (Mumbai), Europa (Londres), América do Sul (São Paulo), Canadá (Central), Europa (Paris), Ásia-Pacífico (Osaka), Europa (Estocolmo), Ásia-Pacífico (Jacarta). Regiões da AWS

Configurações de conexão

- Forneça os valores para os seguintes campos:

  URL da conta do Snowflake

  Especifique um URL de conta do Snowflake. Por exemplo: `xy12345.us-east-1.aws.snowflakecomputing.com`. Consulte a Documentação do Snowflake para saber como determinar o URL da sua conta. Observe que você não deve especificar o número da porta, enquanto o protocolo (https://) é opcional.

  Autenticação

  É possível optar por inserir o login do usuário, a chave privada e a senha manualmente, ou recuperar o segredo do para acessar o Snowflake. AWS Secrets Manager

  - Login do usuário

    Especifique o usuário do Snowflake a ser usado para carregar dados. Certifique-se de que o usuário tenha acesso para inserir dados na tabela do Snowflake.

  - Chave privada

Especifique a chave privada para autenticação com o Snowflake no formato PKCS8. Além disso, não inclua cabeçalho e rodapé do PEM como parte da chave privada. Se a chave estiver dividida em várias linhas, remova as quebras de linha. Veja a seguir um exemplo de como sua chave privada deve se parecer.

```
-----BEGIN PRIVATE KEY-----
KEY_CONTENT
-----END PRIVATE KEY-----
```

Remova o espaço em KEY_CONTENT e forneça-o ao Firehose. Não são necessários caracteres de cabeçalho/rodapé ou nova linha.

- Senha

  Especifique a senha para descriptografar a chave privada criptografada. É possível deixar esse campo vazio se a chave privada não estiver criptografada. Para obter informações, consulte Uso da autenticação de pares de chaves e alternância de chaves.

- Secret

  Selecione um segredo do AWS Secrets Manager que contenha as credenciais para o Snowflake. Se você não vir o segredo na lista suspensa, crie um no AWS Secrets Manager. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

Configuração de perfil

Usar o perfil padrão do Snowflake: se esta opção for selecionada, o Firehose não passará nenhum perfil para o Snowflake. O perfil padrão será assumido para carregar dados. Certifique-se de que o perfil padrão tenha permissão para inserir dados na tabela do Snowflake.

Usar o perfil personalizado do Snowflake: insira um perfil não padrão do Snowflake a ser assumido pelo Firehose ao carregar dados na tabela do Snowflake.

Conectividade do Snowflake

As opções são Privada ou Pública.

ID da VPCE privada (opcional)

O ID da VPCE do Firehose para se conectar de forma privada com o Snowflake. O formato do ID é com.amazonaws.vpce. [região] .vpce-svc-. *[id]* Para obter mais informações, consulte AWS PrivateLink & Snowflake.

> **Note**
>
> Se seu cluster do Snowflake tiver um link privado habilitado, use uma política de rede baseada em `AwsVpceIds` para permitir dados do Amazon Data Firehose. O Firehose não exige que você configure uma política de rede baseada em IP na sua conta do Snowflake. Ter uma política de rede baseada em IP ativada pode interferir na conectividade do Firehose. Se você tiver um caso extremo que exija uma política baseada em IP, entre em contato com a equipe do Firehose enviando um ticket de suporte. Para obter uma lista do VPCE IDs que podem ser usados, consulte o. Acesso ao Snowflake na VPC

Configurar o banco de dados:

- Você deve especificar as configurações a seguir para usar o Snowflake como destino para seu fluxo do Firehose.
  - Banco de dados do Snowflake: todos os dados no Snowflake são mantidos em bancos de dados.
  - Esquema do Snowflake: cada banco de dados consiste em um ou mais esquemas, que são agrupamentos lógicos de objetos de banco de dados, como tabelas e visualizações
  - Tabela do Snowflake: todos os dados no Snowflake são armazenados em tabelas de banco de dados, estruturadas logicamente como coleções de colunas e linhas.

Opções de carregamento de dados para sua tabela do Snowflake

- Uso de chaves JSON como nomes de colunas
- Uso de colunas VARIANT
  - Nome da coluna de conteúdo: especifique um nome de coluna na tabela, onde os dados brutos devem ser carregados.

- Nome da coluna de metadados (opcional): especifique um nome de coluna na tabela onde as informações de metadados devem ser carregadas. Ao ativar esse campo, você verá a coluna a seguir na tabela do Snowflake com base no tipo de fonte.

  Para Direct PUT como fonte

  ```
  {
  "firehoseDeliveryStreamName" : "streamname",
  "IngestionTime" : "timestamp"
  }
  ```

  Para o Kinesis Data Stream como fonte

  ```
  {
  "kinesisStreamName" : "streamname",
  "kinesisShardId" : "Id",
  "kinesisPartitionKey" : "key",
  "kinesisSequenceNumber" : "1234",
  "subsequenceNumber" : "2334",
  "IngestionTime" : "timestamp"
  }
  ```

Período de novas tentativas

Período (0 a 7.200 segundos) durante o qual o Firehose fará novas tentativas se a abertura de canal ou a entrega ao Snowflake falhar devido a problemas de serviço do Snowflake. O Firehose fará novas tentativas com um recuo exponencial, até que a o período de novas tentativas termine. Se você definir a o período de novas tentativas como 0 (zero) segundos, o Firehose não fará novas tentativas após falhas no Snowflake e encaminhará os dados para o bucket de erros do Amazon S3.

Sugestões de buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços. Para obter mais informações, consulte Configuração de sugestões de armazenamento em buffer.

Definição de configurações de destino para o Splunk

Esta seção descreve as opções para usar o Splunk como destino.

> **Note**
>
> O Firehose entrega dados para clusters do Splunk configurados com Classic Load Balancer ou Application Load Balancer.

- Forneça os valores para os seguintes campos:

  Splunk cluster endpoint

  Para determinar o endpoint, consulte Configuração do Amazon Firehose para envio de dados para a plataforma do Splunk na documentação do Splunk.

  Splunk endpoint type

  Escolha `Raw endpoint` na maioria dos casos. Escolha `Event endpoint` se você pré-processou seus dados usando AWS Lambda para enviar dados para índices diferentes por tipo de evento. Para obter informações sobre qual endpoint usar, consulte Configurar o Amazon Kinesis Firehose para enviar dados para a plataforma Splunk na documentação do Splunk.

  Autenticação

  É possível optar por inserir o token de autenticação diretamente ou recuperar o segredo do AWS Secrets Manager para acessar o Splunk.

  - Token de autenticação

    Para configurar um endpoint do Splunk que possa receber dados do Amazon Data Firehose, consulte Visão geral da instalação e configuração do complemento do Splunk para o Amazon Data Firehose na documentação do Splunk. Salve o token que você obtiver do Splunk quando configurar o endpoint para esse fluxo do Firehose e adicione-o aqui.

  - Secret

    Selecione um segredo do AWS Secrets Manager que contenha o token de autenticação do Splunk. Se você não vir o segredo na lista suspensa, crie um no AWS Secrets Manager. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

HEC acknowledgement timeout

Especifique quanto tempo o Amazon Data Firehose aguardará a confirmação do índice do Splunk. Se o Splunk não enviar a confirmação antes de o tempo limite ser atingido, o Amazon Data Firehose considerará que houve uma falha na entrega de dados. Em seguida, o Amazon Data Firehose fará uma nova tentativa ou fará o backup dos dados no bucket do Amazon S3, dependendo do valor do período de novas tentativas que você definir.

Período de novas tentativas

Especifique por quanto o tempo o Amazon Data Firehose fará novas tentativas de enviar os dados ao Splunk.

Depois de enviar os dados, o Amazon Data Firehose primeiro esperará por uma confirmação do Splunk. Se ocorrer um erro ou se a confirmação não chegar dentro do tempo limite para confirmação, o Amazon Data Firehose iniciará o contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Amazon Data Firehose considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Toda vez que o Amazon Data Firehose envia dados para o Splunk (seja uma tentativa inicial ou uma nova tentativa), ele reinicia o contador de tempo limite para confirmação e aguarda uma confirmação do Splunk.

Mesmo que o período de novas tentativas expire, o Amazon Data Firehose ainda aguardará a confirmação até recebê-la ou até que o período de tempo limite para confirmação seja atingido. Se o tempo limite para confirmação expirar, o Amazon Data Firehose determinará se ainda resta algum tempo no contador de novas tentativas. Se houver tempo restante, ele tentará executar novamente e repetirá a lógica até receber um reconhecimento ou determinará que o tempo de tentar novamente expirou.

Se você não quiser que o Amazon Data Firehose tente enviar os dados novamente, defina esse valor como 0.

Sugestões de armazenamento em buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia com base no provedor de serviços.

Definição de configurações de destino para a Splunk Observability Cloud

Esta seção descreve as opções para usar a Splunk Observability Cloud como destino. Para obter mais informações, consulte https://docs.splunk.com/observability/en/gdi/get-data-in/connect/aws/aws-apiconfig.html# - connect-to-aws-using -api. the-splunk-observability-cloud

- Forneça os valores para os seguintes campos:

  URL do endpoint de ingestão na nuvem

  É possível encontrar o URL de ingestão de dados em tempo real da Splunk Observability Cloud em Profile > Organizations > Real-time Data Ingest Endpoint no console do Splunk Observability.

  Autenticação

  É possível optar por inserir o token de acesso diretamente ou recuperar o segredo do AWS Secrets Manager para acessar o Splunk Observability Cloud.

  - Token de acesso

    Copie seu token de acesso do Splunk Observability com o escopo de autorização de INGEST de Tokens de acesso, nas Configurações do console do Splunk Observability.

  - Secret

    Selecione um segredo do AWS Secrets Manager que contenha o token de acesso para o Splunk Observability Cloud Cloud. Se você não vir o segredo na lista suspensa, crie um no AWS Secrets Manager. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

  Codificação de conteúdo

  O Amazon Data Firehose usa a codificação de conteúdo para compactar o corpo de uma solicitação antes de enviá-la ao destino. Escolha GZIP ou Desabilitado para habilitar/desabilitar a codificação de conteúdo da solicitação.

  Período de novas tentativas

  Especifique por quanto tempo o Amazon Data Firehose fará novas tentativas de enviar os dados ao endpoint de HTTP selecionado.

Depois de enviar os dados, o Amazon Data Firehose primeiro esperará por uma confirmação do endpoint de HTTP. Se ocorrer um erro ou se a confirmação não chegar dentro do tempo limite para confirmação, o Amazon Data Firehose iniciará o contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Amazon Data Firehose considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Toda vez que o Amazon Data Firehose envia dados para o endpoint de HTTP (seja a tentativa inicial ou uma nova tentativa), ele reinicia o contador de tempo limite para confirmação e aguarda uma confirmação do endpoint de HTTP.

Mesmo que o período de novas tentativas expire, o Amazon Data Firehose ainda aguardará a confirmação até recebê-la ou até que o período de tempo limite para confirmação seja atingido. Se o tempo limite para confirmação expirar, o Amazon Data Firehose determinará se ainda resta algum tempo no contador de novas tentativas. Se houver tempo restante, ele tentará executar novamente e repetirá a lógica até receber um reconhecimento ou determinará que o tempo de tentar novamente expirou.

Se você não quiser que o Amazon Data Firehose tente enviar os dados novamente, defina esse valor como 0.

Parâmetros: opcional

O Amazon Data Firehose inclui esses pares de valores-chave em toda chamada HTTP. Esses parâmetros ajudam a identificar e organizar os destinos.

Sugestões de armazenamento em buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino varia de acordo com o provedor de serviços.

Definições de configurações de destino para o Sumo Logic

Esta seção descreve as opções para usar o Sumo Logic como destino. Para obter mais informações, consulte https://www.sumologic.com.

- Forneça os valores para os seguintes campos:

URL do endpoint de HTTP

Especifique o URL para o endpoint de HTTP no formato a seguir: `https://deployment name.sumologic.net/receiver/v1/kinesis/dataType/access token`. A origem deve ser uma URL HTTPS.

Codificação de conteúdo

O Amazon Data Firehose usa a codificação de conteúdo para compactar o corpo de uma solicitação antes de enviá-la ao destino. Escolha GZIP ou Desabilitado para habilitar/desabilitar a codificação de conteúdo da solicitação.

Período de novas tentativas

Especifique por quanto tempo o Amazon Data Firehose fará novas tentativas de enviar os dados ao Sumo Logic.

Depois de enviar os dados, o Amazon Data Firehose primeiro esperará por uma confirmação do endpoint de HTTP. Se ocorrer um erro ou se a confirmação não chegar dentro do tempo limite para confirmação, o Amazon Data Firehose iniciará o contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Amazon Data Firehose considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Toda vez que o Amazon Data Firehose envia dados para o endpoint de HTTP (seja a tentativa inicial ou uma nova tentativa), ele reinicia o contador de tempo limite para confirmação e aguarda uma confirmação do endpoint de HTTP.

Mesmo que o período de novas tentativas expire, o Amazon Data Firehose ainda aguardará a confirmação até recebê-la ou até que o período de tempo limite para confirmação seja atingido. Se o tempo limite para confirmação expirar, o Amazon Data Firehose determinará se ainda resta algum tempo no contador de novas tentativas. Se houver tempo restante, ele tentará executar novamente e repetirá a lógica até receber um reconhecimento ou determinará que o tempo de tentar novamente expirou.

Se você não quiser que o Amazon Data Firehose tente enviar os dados novamente, defina esse valor como 0.

Parâmetros: opcional

O Amazon Data Firehose inclui esses pares de valores-chave em toda chamada HTTP. Esses parâmetros ajudam a identificar e organizar os destinos.

Sugestões de armazenamento em buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado para o destino do Elastic varia de acordo com o provedor de serviços.

Definição de configurações de destino para o Elastic

Esta seção descreve as opções para usar o Elastic como destino.

- Forneça os valores para os seguintes campos:

  URL do endpoint do Elastic

  Especifique o URL para o endpoint de HTTP no formato a seguir: `https://<cluster-id>.es.<region>.aws.elastic-cloud.com`. A origem deve ser um URL HTTPS.

  Autenticação

  É possível optar por inserir a chave da API diretamente ou recuperar o segredo do AWS Secrets Manager para acessar o Elastic.

  - Chave de API

    Entre em contato com o Elastic para obter a chave de API necessária para permitir a entrega de dados do Firehose para esse serviço.

  - Secret

    Selecione um segredo do AWS Secrets Manager que contenha a chave da API do Elastic. Se você não vir o segredo na lista suspensa, crie um no AWS Secrets Manager. Para obter mais informações, consulte Autentique-se com o AWS Secrets Manager Amazon Data Firehose.

Codificação de conteúdo

O Amazon Data Firehose usa a codificação de conteúdo para compactar o corpo de uma solicitação antes de enviá-la ao destino. Escolha GZIP (que é o selecionado por padrão) ou Desabilitado para habilitar/desabilitar a codificação de conteúdo da solicitação.

Período de novas tentativas

Especifique por quanto tempo o Amazon Data Firehose fará novas tentativas de enviar os dados ao Elastic.

Depois de enviar os dados, o Amazon Data Firehose primeiro esperará por uma confirmação do endpoint de HTTP. Se ocorrer um erro ou se a confirmação não chegar dentro do tempo limite para confirmação, o Amazon Data Firehose iniciará o contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Amazon Data Firehose considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Toda vez que o Amazon Data Firehose envia dados para o endpoint de HTTP (seja a tentativa inicial ou uma nova tentativa), ele reinicia o contador de tempo limite para confirmação e aguarda uma confirmação do endpoint de HTTP.

Mesmo que o período de novas tentativas expire, o Amazon Data Firehose ainda aguardará a confirmação até recebê-la ou até que o período de tempo limite para confirmação seja atingido. Se o tempo limite para confirmação expirar, o Amazon Data Firehose determinará se ainda resta algum tempo no contador de novas tentativas. Se houver tempo restante, ele tentará executar novamente e repetirá a lógica até receber um reconhecimento ou determinará que o tempo de tentar novamente expirou.

Se você não quiser que o Amazon Data Firehose tente enviar os dados novamente, defina esse valor como 0.

Parâmetros: opcional

O Amazon Data Firehose inclui esses pares de valores-chave em toda chamada HTTP. Esses parâmetros ajudam a identificar e organizar os destinos.

Sugestões de armazenamento em buffer

O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los ao destino especificado. O tamanho de buffer recomendado quando o destino é O Elastic é de 1 MiB.

Definição das configurações de backup

O Amazon Data Firehose usa o Amazon S3 para fazer backup de todos os dados ou apenas dos dados com falha que ele tenta entregar ao destino escolhido.

> **Important**
>
> - Só há suporte para as configurações de backup se a fonte do fluxo do Firehose for o Direct PUT ou o Kinesis Data Streams.
> - O atributo de buffer zero está disponível somente para os destinos da aplicação, e não está disponível para o destino de backup do Amazon S3.

É possível especificar as configurações de backup do S3 para seu fluxo do Firehose se tiver feito uma das escolhas a seguir.

- Se você definir o Amazon S3 como destino para seu stream do Firehose e optar por especificar uma função AWS Lambda para transformar registros de dados ou se optar por converter formatos de registro de dados para seu stream do Firehose.
- Se você definir o Amazon Redshift como destino para seu stream do Firehose e optar por especificar uma função AWS Lambda para transformar registros de dados.
- Se você definir qualquer um dos seguintes serviços como destino para seu stream do Firehose — Amazon OpenSearch Service, Datadog, Dynatrace, HTTP Endpoint, LogicMonitor MongoDB Cloud, New Relic, Splunk ou Sumo Logic, Snowflake, Apache Iceberg Tables.

Veja a seguir as configurações de backup para seu fluxo do Firehose.

- Backup do registro de origem no Amazon S3: se o S3 ou o Amazon Redshift for o destino selecionado, essa configuração indicará se você deseja habilitar o backup dos dados da fonte ou mantê-lo desabilitado. Se qualquer outro serviço compatível (exceto o S3 ou o Amazon Redshift)

estiver definido como seu destino selecionado, essa configuração indicará se você deseja fazer backup de todos os dados da fonte ou apenas dos dados com falha.

- Bucket de backup do S3: esse é o bucket do S3 em que o Amazon Data Firehose faz backup dos dados.
- Prefixo de bucket de backup do S3: esse é o prefixo em que o Amazon Data Firehose faz backup dos dados.
- Prefixo da saída de erros do bucket de backup do S3: todos os dados com falha são copiados nesse prefixo da saída de erros do bucket do S3.
- Sugestões sobre armazenamento em buffer, compactação e criptografia de backup: o Amazon Data Firehose usa o Amazon S3 para fazer backup de todos os dados ou apenas dos dados com falha que ele tenta entregar ao destino escolhido. O Amazon Data Firehose armazena em buffer os dados recebidos antes de entregá-los (colocá-los no backup) ao Amazon S3. Você pode escolher um tamanho de buffer de 1 a 128 MiBs e um intervalo de buffer de 60 a 900 segundos. A condição que é satisfeita primeiro aciona a entrega de dados ao Amazon S3. Se você habilitar a transformação dos dados, o intervalo de buffer é aplicado desde o momento em que os dados transformados são recebidos pelo Amazon Data Firehose até a entrega de dados ao Amazon S3. Se a entrega de dados ao destino ficar atrasada em relação à gravação de dados no fluxo do Firehose, o Amazon Data Firehose aumentará o tamanho do buffer dinamicamente para recuperar o atraso. Essa ação ajuda a garantir que todos os dados sejam entregues no destino.
- Compactação do S3: escolha compactação de dados Snappy compatível com GZIP, Snappy, Zip ou Hadoop, ou nenhuma compactação de dados. A compactação Snappy, Zip e Snappy compatível com Hadoop não está disponível para fluxos do Firehose com o Amazon Redshift como destino.
- Formato de extensão de arquivo S3 (opcional): especifique um formato de extensão de arquivo para objetos entregues ao bucket de destino do Amazon S3. Se você habilitar esse atributo, a extensão de arquivo especificada substituirá as extensões de arquivo padrão anexadas pelos atributos de conversão de formato de dados ou compactação do S3, como .parquet ou .gz. Verifique se você configurou a extensão de arquivo correta ao usar esse atributo com a conversão de formato de dados ou a compactação do S3. A extensão do arquivo deve começar com um ponto (.) e pode conter os caracteres permitidos: 0-9a-z!-_.*' (). A extensão do arquivo não pode exceder 128 caracteres.
- O Firehose oferece suporte à criptografia do lado do servidor Amazon S3 AWS Key Management Service com (SSE-KMS) para criptografar dados entregues no Amazon S3. Você pode optar por usar o tipo de criptografia padrão especificado no bucket S3 de destino ou criptografar com uma chave da lista de AWS KMS chaves que você possui. Se você criptografar os dados com AWS

KMS chaves, poderá usar a chave AWS gerenciada padrão (aws/s3) ou uma chave gerenciada pelo cliente. Para obter mais informações, consulte Proteção de dados usando criptografia do lado do servidor com chaves AWS gerenciadas pelo KMS (SSE-KMS).

Configuração de sugestões de armazenamento em buffer

O Amazon Data Firehose armazena em buffer os dados em streaming em memória recebidos até um determinado tamanho (tamanho de armazenamento em buffer) e por um determinado período (intervalo de armazenamento em buffer) antes de entregá-los aos destinos especificados. Você deve usar as sugestões de armazenamento em buffer quando quiser entregar arquivos de tamanho ideal para o Amazon S3 e obter melhor performance das aplicações de processamento de dados ou para ajustar a taxa de entrega do Firehose de acordo com a velocidade de destino.

É possível configurar o tamanho do armazenamento em buffer e o intervalo do buffer ao criar novos fluxos do Firehose ou atualizar o tamanho do armazenamento buffer e o intervalo de armazenamento em buffer nos fluxos do Firehose existentes. O tamanho do buffer é medido em MBs e o intervalo de buffer é medido em segundos. Contudo, se especificar um valor para um deles, você também deverá fornecer um valor para o outro. A primeira condição de buffer atendida aciona o Firehose para entregar os dados. Se você não configurar os valores de armazenamento em buffer, os valores padrão serão usados.

Você pode configurar dicas de buffer do Firehose por meio do,, ou. AWS Management Console AWS Command Line Interface AWS SDKs Para streams existentes, você pode reconfigurar dicas de buffer com um valor adequado aos seus casos de uso usando a opção Editar no console ou usando a API. UpdateDestination Para novos streams, você pode configurar dicas de buffer como parte da criação de um novo stream usando o console ou usando a API. CreateDeliveryStream Para ajustar o tamanho do buffer, defina `SizeInMBs` e `IntervalInSeconds` no `DestinationConfiguration` parâmetro específico de destino da API CreateDeliveryStreamou UpdateDestination.

> **Note**
>
> - As sugestões de armazenamento em buffer são aplicadas em um nível de fragmento ou partição, enquanto as sugestões de armazenamento em buffer de particionamento dinâmico são aplicadas em nível de fluxo ou tópico.
> - Para atender às latências mais baixas dos casos de uso em tempo real, é possível usar a sugestão de intervalo de armazenamento em buffer zero. Quando você configura o intervalo de armazenamento em buffer como zero segundos, o Firehose não armazena

dados em buffer e os entrega em alguns segundos. Antes de alterar as sugestões de armazenamento em buffer para um valor menor, consulte o fornecedor as sugestões de armazenamento em recomendadas do Firehose para seus destinos.

- O atributo de buffer zero está disponível somente para os destinos da aplicação, e não está disponível para o destino de backup do Amazon S3.
- O atributo de armazenamento em buffer zero não está disponível para o particionamento dinâmico.
- O Firehose usa o upload de várias partes para o destino do S3 quando você configura um intervalo de tempo de armazenamento em buffer inferior a 60 segundos para oferecer latências mais baixas. Devido ao upload de várias partes para o destino do S3, você verá algum aumento nos custos da API PUT do S3 se escolher um intervalo de tempo de buffer menor que 60 segundos.

Para intervalos de sugestões de armazenamento em buffer e valores padrão específicos do destino, consulte a tabela a seguir:

Destino	Tamanho do armazenamento em buffer, em MB (padrão entre parênteses)	Intervalo de armazenamento em buffer, em segundos (padrão entre parênteses)
Amazon S3	1-128 (5)	0-900 (300)
Tabelas do Apache Iceberg	1-128 (5)	0-900 (300)
Amazon Redshift	1-128 (5)	0-900 (300)
OpenSearch Sem servidor	1-100 (5)	0-900 (300)
OpenSearch	1-100 (5)	0-900 (300)
Splunk	1-5 (5)	0-60 (60)
Datadog	1-4 (4)	0-900 (60)

Destino	Tamanho do armazenamento em buffer, em MB (padrão entre parênteses)	Intervalo de armazenamento em buffer, em segundos (padrão entre parênteses)
Coralogix	1-64 (6)	0-900 (60)
Dynatrace	1-64 (5)	0-900 (60)
Elastic	1	0-900 (60)
Honeycomb	1-64 (15)	0-900 (60)
Endpoint de HTTP	1-64 (5)	0-900 (60)
LogicMonitor	1-64 (5)	0-900 (60)
Logzio	1-64 (5)	0-900 (60)
mongoDB	1-16 (5)	0-900 (60)
newRelic	1-64 (5)	0-900 (60)
sumoLogic	1-64 (1)	0-900 (60)
Splunk Observability Cloud	1-64 (1)	0-900 (60)
Snowflake	1 - 128 (1)	0 - 900 (0)

Definir as configurações avançadas

A seção a seguir contém detalhes sobre as configurações avançadas do fluxo do Firehose.

- Criptografia do lado do servidor - O Amazon Data Firehose oferece suporte à criptografia do lado do servidor do Amazon S3 AWS com o Key Management Service (AWS KMS) para criptografar dados entregues no Amazon S3. Para obter mais informações, consulte Proteção de dados usando criptografia do lado do servidor com chaves gerenciadas pelo KMS (AWS SSE-KMS).

- Registro em log de erros: o Amazon Data Firehose registra em log os erros relacionados a processamento e entrega. Além disso, quando a transformação de dados está ativada, ela pode registrar invocações do Lambda e enviar erros de entrega de dados para o Logs. CloudWatch Para obter mais informações, consulte Monitoramento do Amazon Data Firehose usando logs CloudWatch .

  > **Important**
  >
  > Embora opcional, é extremamente recomendável habilitar o registro em log dos erros do Amazon Data Firehose durante a criação do fluxo do Firehose. Essa prática garante que você possa acessar os detalhes do erro em caso de falhas no processamento de registros ou na entrega.

- O Amazon Data Firehose usa os perfis do IAM para todas as permissões de que o fluxo do Firehose precisa. É possível escolher criar um novo perfil quando as permissões necessárias são atribuídas automaticamente ou escolher um perfil existente criado para o Amazon Data Firehose. A função é usada para conceder ao Firehose acesso a vários serviços, incluindo seu bucket do S3, chave AWS KMS (se a criptografia de dados estiver ativada) e função Lambda (se a transformação de dados estiver ativada). O console talvez crie um perfil com espaços reservados. Para obter mais informações, consulte O que é IAM?.

  > **Note**
  >
  > O perfil do IAM (incluindo espaços reservados) é criada com base na configuração que você escolhe ao criar um fluxo do Firehose. Se você fizer alguma alteração na fonte ou no destino do fluxo do Firehose, será necessário atualizar manualmente o perfil do IAM.

- Tags - Você pode adicionar tags para organizar seus AWS recursos, monitorar custos e controlar o acesso.

  Se tags forem especificadas na ação `CreateDeliveryStream`, o Amazon Data Firehose realizará uma autorização adicional na ação `firehose:TagDeliveryStream` para verificar se os usuários têm permissões para criar tags. Se essa permissão não for fornecida, as solicitações para criar novos fluxos do Firehose com tags de recursos do IAM falharão com `AccessDeniedException`, conforme a seguir.

  ```
  AccessDeniedException
  ```

```
User: arn:aws:sts::x:assumed-role/x/x is not authorized to perform:
 firehose:TagDeliveryStream on resource: arn:aws:firehose:us-east-1:x:deliverystream/
x with an explicit deny in an identity-based policy.
```

O exemplo a seguir demonstra uma política que permite aos usuários criar um fluxo do Firehose e aplicar tags.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "firehose:CreateDeliveryStream",
            "Resource": "*",
            }
        },
        {
            "Effect": "Allow",
            "Action": "firehose:TagDeliveryStream",
            "Resource": "*",
            }
        }
    ]
}
```

Depois que você definir as configurações avançadas e as configurações de backup, revise suas opções e escolha Criar fluxo do Firehose.

O novo fluxo do Firehose passa alguns segundos no estado Em criação antes de ficar disponível. Depois que o fluxo do Firehose entrar no estado Ativo, será possível iniciar o envio de dados do produtor para ele.

Testes do fluxo do Firehose com dados de amostra

Você pode usar o AWS Management Console para ingerir dados simulados de ações. seu O console executa um script no navegador para colocar registros de exemplo no seu fluxo do Firehose. Isso permite que você teste a configuração do seu fluxo do Firehose sem precisar gerar seus próprios dados de teste.

Este é um exemplo dos dados simulados:

```
{"TICKER_SYMBOL":"QXZ","SECTOR":"HEALTHCARE","CHANGE":-0.05,"PRICE":84.51}
```

Observe que os custos padrão do Amazon Data Firehose se aplicam quando seu fluxo do Firehose transmite os dados, mas não há custos quando os dados são gerados. Para interromper essas cobranças, é possível parar o fluxo de exemplo no console a qualquer momento.

Pré-requisitos

Antes de começar, crie um fluxo do Firehose. Para obter mais informações, consulte Tutorial: Criação de um fluxo do Firehose a partir do console.

Teste com o Amazon S3

Execute o procedimento a seguir para testar o fluxo do Firehose com o Amazon Simple Storage Service (Amazon S3) como destino.

Para testar um fluxo do Firehose usando o Amazon S3

1. Abra o console Firehose em. https://console.aws.amazon.com/firehose/
2. Escolha um fluxo ativo do Firehose. O fluxo do Firehose precisa estar no status Ativo para que você possa começar a enviar dados.
3. Em Test with demo data, escolha Start sending demo data para gerar dados de exemplo do índice de ações.
4. Siga as instruções na tela para verificar se os dados estão sendo entregues ao bucket do S3. Observe que pode demorar alguns minutos para que os novos objetos apareçam no bucket, com base na configuração de armazenamento em buffer do bucket.
5. Quando o teste for concluído, escolha Stop sending demo data para cessar a cobrança de uso.

Teste com o Amazon Redshift

Execute o procedimento a seguir para testar seu fluxo do Firehose com o Amazon Redshift como destino.

Para testar um fluxo do Firehose usando o Amazon Redshift

1. Seu fluxo do Firehose espera que uma tabela esteja presente no cluster do Amazon Redshift. Conecte-se ao Amazon Redshift por meio de uma interface SQL e execute a instrução a seguir para criar uma tabela que aceite a amostra de dados.

   ```
   create table firehose_test_table
   (
    TICKER_SYMBOL varchar(4),
    SECTOR varchar(16),
    CHANGE float,
    PRICE float
   );
   ```

2. Abra o console Firehose em. https://console.aws.amazon.com/firehose/
3. Escolha um fluxo ativo do Firehose. O fluxo do Firehose precisa estar no status Ativo para que você possa começar a enviar dados.
4. Edite os detalhes do destino do seu fluxo do Firehose para apontar para a tabela `firehose_test_table` recém-criada.
5. Em Test with demo data, escolha Start sending demo data para gerar dados de exemplo do índice de ações.
6. Siga as instruções na tela para verificar se os dados estão sendo entregues na tabela. Observe que pode demorar alguns minutos para que novas linhas apareçam na tabela, com base na configuração de armazenamento em buffer.
7. Quando o teste for concluído, escolha Stop sending demo data para cessar a cobrança de uso.
8. Edite os detalhes do destino do seu fluxo do Firehose para apontar para uma outra tabela.
9. (Opcional) Exclua a tabela `firehose_test_table`.

Teste com OpenSearch serviço

Use o procedimento a seguir para testar seu stream do Firehose usando o Amazon OpenSearch Service como destino.

Para testar um stream do Firehose usando o Service OpenSearch

1. Abra o console Firehose em. https://console.aws.amazon.com/firehose/
2. Escolha um fluxo ativo do Firehose. O fluxo do Firehose precisa estar no status Ativo para que você possa começar a enviar dados.
3. Em Test with demo data, escolha Start sending demo data para gerar dados de exemplo do índice de ações.
4. Siga as instruções na tela para verificar se os dados estão sendo entregues ao seu domínio do OpenSearch Serviço. Para obter mais informações, consulte Pesquisando documentos em um domínio OpenSearch de serviço no Amazon OpenSearch Service Developer Guide.
5. Quando o teste for concluído, escolha Stop sending demo data para cessar a cobrança de uso.

Teste com o Splunk

Execute o procedimento a seguir para testar o fluxo do Firehose com o Splunk como destino.

Para testar um fluxo do Firehose usando o Splunk

1. Abra o console Firehose em. https://console.aws.amazon.com/firehose/
2. Escolha um fluxo ativo do Firehose. O fluxo do Firehose precisa estar no status Ativo para que você possa começar a enviar dados.
3. Em Test with demo data, escolha Start sending demo data para gerar dados de exemplo do índice de ações.
4. Verifique se os dados estão sendo entregues para o seu índice do Splunk. Exemplo de termos de pesquisa no Splunk são `sourcetype="aws:firehose:json"` e `index="name-of-your-splunk-index"`. Para obter mais informações sobre como pesquisar eventos no Splunk, consulte Pesquisar manual na documentação do Splunk.

   Se os dados de teste não aparecerem no índice do Splunk, verifique se há eventos com falha no bucket do Amazon S3. Consulte também Dados não entregues ao Splunk.
5. Quando concluir o teste, escolha Stop sending demo data para cessar a cobrança de uso.

Teste com tabelas do Apache Iceberg

Execute o procedimento a seguir para testar seu fluxo do Firehose com o tabelas do Apache Iceberg como destino.

Para testar um fluxo do Firehose usando tabelas do Apache Iceberg

1. Abra o console Firehose em. https://console.aws.amazon.com/firehose/
2. Escolha um fluxo ativo do Firehose. O fluxo do Firehose precisa estar no status Ativo para que você possa começar a enviar dados.
3. Em Test with demo data, escolha Start sending demo data para gerar dados de exemplo do índice de ações.
4. Siga as instruções na tela para verificar se os dados estão sendo entregues às suas tabelas do Apache Iceberg. Observe que pode demorar alguns minutos para que os novos objetos apareçam no seu bucket, com base na configuração de armazenamento em buffer.
5. Se os dados de teste não aparecerem nas suas tabelas do Apache Iceberg, verifique se há eventos com falha no bucket do Amazon S3.
6. Quando concluir o teste, escolha Stop sending demo data para cessar a cobrança de uso.

Envio de dados a um fluxo do Firehose

Esta seção descreve como é possível usar fontes de dados diferentes para enviar dados para seu fluxo do Firehose. Se você estiver começando a usar o Amazon Data Firehose, dedique algum tempo para se familiarizar com os conceitos e a terminologia apresentados em O que é o Amazon Data Firehose?.

> **Note**
>
> Alguns AWS serviços só podem enviar mensagens e eventos para um stream do Firehose que esteja na mesma região. Se seu stream do Firehose não aparecer como uma opção quando você estiver configurando um destino para Amazon CloudWatch Logs, CloudWatch Events AWS IoT, ou verifique se seu stream do Firehose está na mesma região que seus outros serviços. Para obter informações sobre endpoints de serviço para cada região, consulte endpoints do Amazon Data Firehose.

É possível enviar dados para seu fluxo do Firehose a partir das fontes de dados a seguir.

Tópicos

- Configurar o agente do Kinesis para enviar dados
- Envie dados com o AWS SDK
- Enviar CloudWatch registros para o Firehose
- Enviar CloudWatch eventos para Firehose
- Configure AWS IoT para enviar dados para o Firehose

Configurar o agente do Kinesis para enviar dados

O agente do Amazon Kinesis é uma aplicação de software Java autônoma que serve como uma implementação de referência para mostrar como é possível coletar e enviar dados para o Firehose. O agente monitora continuamente um conjunto de arquivos e envia novos dados ao seu fluxo do Firehose. O agente mostra como é possível manipular a alternância de arquivos, os pontos de verificação e as novas tentativas após falhas. Ele mostra como é possível entregar seus dados de maneira confiável, imediata e simples. Também mostra como você pode emitir CloudWatch métricas

para melhor monitorar e solucionar problemas no processo de streaming. Para saber mais, awslabs/amazon-kinesis-agent.

Por padrão, os registros são analisados em cada arquivo com base no caractere de nova linha (`'\n'`). No entanto, o agente também pode ser configurado para analisar registros de várias linhas (consulte Especificação das definições de configuração do agente).

É possível instalar o agente em ambientes de servidor baseados no Linux, como servidores web, servidores de log e servidores de banco de dados. Após instalar o agente, configure-o especificando os arquivos a serem monitorados e o fluxo do Firehose para os dados. Depois que o agente é configurado, ele coleta dados dos arquivos de forma durável e os envia confiavelmente ao fluxo do Firehose.

Pré-requisitos

Antes começar a usar o agente do Kinesis, certifique-se de atender aos pré-requisitos a seguir.

- O sistema operacional deve ser o Amazon Linux, ou o Red Hat Enterprise Linux versão 7 ou posterior.
- O agente versão 2.0.0 ou posterior é executado usando o JRE versão 1.8 ou posterior. O agente versão1.1x é executado usando o JRE versão 1.7 ou posterior.
- Se você estiver usando EC2 a Amazon para executar seu agente, inicie sua EC2 instância.
- A função ou AWS as credenciais do IAM que você especificar devem ter permissão para realizar a operação do Amazon Data PutRecordBatchFirehose para que o agente envie dados para seu stream do Firehose. Se você ativar o CloudWatch monitoramento para o agente, a permissão para realizar a CloudWatch PutMetricDataoperação também será necessária. Para obter mais informações, consulte, Controle de acesso com o Amazon Data FirehoseMonitoramento da integridade do Kinesis Agent, e Autenticação e controle de acesso para a Amazon CloudWatch.

Gerenciar AWS credenciais

Gerencie suas AWS credenciais usando um dos seguintes métodos:

- Crie um provedor de credenciais personalizadas. Para obter detalhes, consulte the section called “Criação de provedores de credenciais personalizadas”.
- Especifique uma função do IAM ao iniciar sua EC2 instância.

- Especifique AWS as credenciais ao configurar o agente (veja as entradas para `awsAccessKeyId` e `awsSecretAccessKey` na tabela de configuração abaixothe section called “Especificação das definições de configuração do agente”).
- Edite `/etc/sysconfig/aws-kinesis-agent` para especificar sua AWS região e chaves de AWS acesso.
- Se sua EC2 instância estiver em uma AWS conta diferente, crie uma função do IAM para fornecer acesso ao serviço Amazon Data Firehose. Especifique essa função ao configurar o agente (consulte AssumeroLearn e IdassumeRoleExternal). Use um dos métodos anteriores para especificar AWS as credenciais de um usuário na outra conta que tenha permissão para assumir essa função.

Criação de provedores de credenciais personalizadas

É possível criar um provedor de credenciais personalizadas e fornecer seu nome de classe e caminho jar ao Kinesis Agent nas seguintes configurações: `userDefinedCredentialsProvider.classname` e `userDefinedCredentialsProvider.location`. Para obter as descrições dessas duas configurações, consulte the section called “Especificação das definições de configuração do agente”.

Para criar um provedor de credenciais personalizadas, defina uma classe que implemente a interface `AWS CredentialsProvider`, como a do exemplo a seguir.

```
import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.AWSCredentialsProvider;
import com.amazonaws.auth.BasicAWSCredentials;

public class YourClassName implements AWSCredentialsProvider {
    public YourClassName() {
    }

    public AWSCredentials getCredentials() {
        return new BasicAWSCredentials("key1", "key2");
    }

    public void refresh() {
    }
}
```

Sua classe deve ter um construtor que não aceite argumentos.

AWS invoca o método de atualização periodicamente para obter credenciais atualizadas. Se você quiser que seu provedor de credenciais forneça credenciais diferentes ao longo da vida útil, inclua código para atualizar as credenciais neste método. Também é possível deixar esse método vazio se quiser um provedor de credenciais que venda credenciais estáticas (sem alteração).

Download e instalação do agente

Primeiro, conecte-se à instância. Para obter mais informações, consulte Connect to Your Instance no Guia EC2 do usuário da Amazon. Se você tiver problemas para se conectar, consulte Solução de problemas de conexão com sua instância no Guia EC2 do usuário da Amazon.

Em seguida, instale o agente usando um dos métodos a seguir.

- Para configurar o agente a partir dos repositórios do Amazon Linux

  Esse método só funciona para instâncias do Amazon Linux. Use o seguinte comando:

  ```
  sudo yum install -y aws-kinesis-agent
  ```

  O Agent v 2.0.0 ou posterior está instalado em computadores com o sistema operacional Amazon Linux 2 (AL2). Essa versão do agente requer o Java versão 1.8 ou posterior. Se a versão Java requerida ainda não estiver presente, o processo de instalação do agente a instalará. Para obter mais informações sobre o Amazon Linux 2, consulte https://aws.amazon.com/amazon-linux-2/.

- Para configurar o agente a partir dos repositórios do Amazon S3

  Esse método funciona para o Red Hat Enterprise Linux e para instâncias do Amazon Linux 2, pois instala o agente a partir do repositório disponível publicamente. Use o comando a seguir para baixar e instalar a versão mais recente do agente versão 2.x.x:

  ```
  sudo yum install -y https://s3.amazonaws.com/streaming-data-agent/aws-kinesis-agent-latest.amzn2.noarch.rpm
  ```

  Para instalar uma versão específica do agente, especifique o número da versão no comando. Por exemplo, o comando a seguir instala o agente versão 2.0.1.

```
sudo yum install -y https://streaming-data-agent.s3.amazonaws.com/aws-kinesis-
agent-2.0.1-1.amzn1.noarch.rpm
```

Se você tiver o Java 1.7 e não quiser atualizá-lo, poderá baixar o agente versão 1.x.x que é compatível com o Java 1.7. Por exemplo, para baixar o agente v1.1.6, é possível usar o comando a seguir:

```
sudo yum install -y https://s3.amazonaws.com/streaming-data-agent/aws-kinesis-
agent-1.1.6-1.amzn1.noarch.rpm
```

É possível baixar o agente mais recente com o comando a seguir

```
sudo yum install -y https://s3.amazonaws.com/streaming-data-agent/aws-kinesis-agent-
latest.amzn2.noarch.rpm
```

- Para configurar o agente a partir do GitHub repositório

  1. Primeiro, certifique-se de que a versão do Java requerida esteja instalada, dependendo da versão do agente.
  2. Faça o download do agente do awslabs/ repo amazon-kinesis-agent GitHub .
  3. Instale o agente navegando até o diretório de download e executando o comando a seguir:

     ```
     sudo ./setup --install
     ```

- Como configurar o agente em um contêiner do Docker

  O Kinesis Agent também pode ser executado em um contêiner por meio da base de contêineres amazonlinux. Use o Dockerfile a seguir e depois execute o `docker build`.

```
FROM amazonlinux

RUN yum install -y aws-kinesis-agent which findutils
COPY agent.json /etc/aws-kinesis/agent.json
```

```
CMD ["start-aws-kinesis-agent"]
```

Configuração e inicialização do agente

Como configurar e iniciar o agente

1. Abra e edite o arquivo de configuração (como superusuário, se as permissões padrão de acesso a arquivos estiverem sendo usadas): `/etc/aws-kinesis/agent.json`

   Nesse arquivo de configuração, especifique os arquivos (`"filePattern"`) nos quais o agente coleta dados e o nome do fluxo do Firehose (`"deliveryStream"`) ao qual o agente envia dados. O nome do arquivo é um padrão, e o agente reconhece os rodízios de arquivos. Só é possível fazer o rodízio de arquivos ou criar novos arquivos uma vez por segundo, no máximo. O agente usa o carimbo de data e hora de criação de arquivo para determinar quais arquivos serão rastreados e colocados no final do fluxo do Firehose. A criação de novos arquivos ou o rodízio de arquivos em uma frequência superior a uma vez por segundo não permite que o agente faça a distinção entre eles corretamente.

   ```
   {
      "flows": [
           {
               "filePattern": "/tmp/app.log*",
               "deliveryStream": "yourdeliverystream"
           }
      ]
   }
   ```

   A AWS região padrão é`us-east-1`. Se você estiver usando outra região, adicione a configuração `firehose.endpoint` ao arquivo de configuração, especificando o endpoint para a sua região. Para obter mais informações, consulte Especificação das definições de configuração do agente.

2. Inicie o agente manualmente:

   ```
   sudo service aws-kinesis-agent start
   ```

3. (Opcional) Configure o agente para ser iniciado durante o startup do sistema:

```
sudo chkconfig aws-kinesis-agent on
```

Agora o agente está sendo executado como um serviço do sistema em segundo plano. Ele monitora continuamente os arquivos especificados e envia dados ao fluxo do Firehose especificado. A atividade do agent é registrada em `/var/log/aws-kinesis-agent/aws-kinesis-agent.log`.

Especificação das definições de configuração do agente

O agente oferece suporte a duas configurações obrigatórias, `filePattern` e `deliveryStream`, além das configurações opcionais de recursos adicionais. É possível especificar configurações obrigatórios e opcionais em `/etc/aws-kinesis/agent.json`.

Sempre que o arquivo de configuração for alterado, o agente deverá ser interrompido e iniciado, usando os seguintes comandos:

```
sudo service aws-kinesis-agent stop
sudo service aws-kinesis-agent start
```

Se desejar, é possível usar o comando a seguir:

```
sudo service aws-kinesis-agent restart
```

Estas são as configurações gerais.

Definição da configuração	Descrição
`assumeRoleARN`	O nome de recurso da Amazon (ARN) do perfil a ser assumido pelo usuário. Para obter mais informações, consulte Delegar acesso entre AWS contas usando funções do IAM no Guia do usuário do IAM.
`assumeRoleExternalId`	Um identificador opcional que determina quem pode assumir o perfil. Para obter mais informações, consulte Como usar um ID externo no Guia do usuário do IAM.

Definição da configuração	Descrição
`awsAccessKeyId`	AWS ID da chave de acesso que substitui as credenciais padrão. Essa configuração tem precedência sobre todos os outros provedores de credenciais.
`awsSecretAccessKey`	AWS chave secreta que substitui as credenciais padrão. Essa configuração tem precedência sobre todos os outros provedores de credenciais.
`cloudwatch.emitMetrics`	Permite que o agente emita métricas para, CloudWatch se definidas (verdadeiras). Padrão: verdadeiro
`cloudwatch.endpoint`	O endpoint regional para CloudWatch. Padrão: `monitoring.us-east-1.amazonaws.com`
`firehose.endpoint`	O endpoint regional para o Amazon Data Firehose. Padrão: `firehose.us-east-1.amazonaws.com`
`sts.endpoint`	O endpoint regional do AWS Security Token Service. Padrão: `https://sts.amazonaws.com`
`userDefinedCredentialsProvider.classname`	Se você definir um provedor de credenciais personalizadas, forneça seu nome de classe totalmente qualificado usando essa configuração. Não inclua `.class` no final do nome da classe.
`userDefinedCredentialsProvider.location`	Se você definir um provedor de credenciais personalizadas, use essa configuração para especificar o caminho absoluto do jar que contém o provedor de credenciais personalizadas. O agente também procura o arquivo jar no seguinte local: `/usr/share/aws-kinesis-agent/lib/`.

Estas são as configurações de fluxo.

Definição da configuração	Descrição
`aggregatedRecordSizeBytes`	Para fazer com que o agente agregue registros e, depois, coloque-os no fluxo do Firehose de uma operação, especifique essa configuração. Defina o tamanho desejado do registro agregado antes que o agente o coloque no fluxo do Firehose. Padrão: 0 (sem agregação)
`dataProcessingOptions`	A lista das opções de processamento aplicadas a cada registro analisado antes que ele seja enviado ao fluxo do Firehose. As opções de processamento são executadas na ordem especificada. Para obter mais informações, consulte Pré-processamento de dados com agentes.
`deliveryStream`	[Obrigatório] O nome do fluxo do Firehose.
`filePattern`	[Obrigatório] Um glob para os arquivos que precisam ser monitorados pelo agente. Qualquer arquivo que corresponda a esse padrão é selecionado pelo agente automaticamente e monitorado. Para todos os arquivos correspondentes a esse padrão, conceda permissão de leitura a `aws-kinesis-agent-user` . Para o diretório que contém os arquivos, conceda permissões de leitura e execução a `aws-kinesis-agent-user` . **Important** O agente seleciona qualquer arquivo que corresponda a esse padrão. Para garantir que o agente não selecione registros não intencionais, escolha esse padrão cuidadosamente.
`initialPosition`	A posição em que o arquivo começou a ser analisado. Os valores válidos são `START_OF_FILE` e `END_OF_FILE` . Padrão: `END_OF_FILE`
`maxBufferAgeMillis`	O tempo máximo, em milissegundos, durante o qual o agente armazena os dados em buffer antes de enviá-los ao fluxo do Firehose.

Definição da configuração	Descrição
	Intervalo de valores: 1.000 a 900.000 (1 segundo a 15 minutos) Padrão: 60.000 (1 minuto)
`maxBufferSizeBytes`	O tamanho máximo, em bytes, durante o qual o agente armazena os dados em buffer antes de enviá-los ao fluxo do Firehose. Intervalo de valores: 1 a 4.194.304 (4 MB) Padrão: 4.194.304 (4 MB)
`maxBufferSizeRecords`	O número máximo de registros para os quais o agente armazena os dados em buffer antes de enviá-los ao fluxo do Firehose. Intervalo de valores: 1 a 500 Padrão: 500
`minTimeBetweenFilePollsMillis`	O intervalo de tempo, em milissegundos, em que o agente consulta e analisa os arquivos monitorados em busca de novos dados. Intervalo de valores: 1 ou mais Padrão: 100
`multiLineStartPattern`	O padrão de identificação do início de um registro. Um registro é composto por uma linha que corresponde ao padrão e pelas linhas subsequentes que não correspondem ao padrão. Os valores válidos são expressões regulares. Por padrão, cada nova linha nos arquivos de log é analisada como um único registro.
`skipHeaderLines`	O número de linhas em que o agente ignorará a análise no início dos arquivos monitorados. Intervalo de valores: 0 ou mais Padrão: 0 (zero)

Definição da configuração	Descrição
`truncatedRecordTerminator`	A string que o agente usa para truncar um registro analisado quando o tamanho do registro excede o limite de tamanho de registro do Amazon Data Firehose. (1,000 KB) Padrão: `'\n'` (nova linha)

Configuração de vários fluxos e diretórios de arquivos

Ao especificar vários fluxos de configurações, é possível configurar o agente para monitorar vários diretórios de arquivos e enviar dados a vários streams. No exemplo de configuração a seguir, o agente monitora dois diretórios de arquivos e envia dados para um fluxo de dados do Kinesis para um fluxo do Firehose respectivamente. É possível especificar endpoints diferentes para o Kinesis Data Streams e o Amazon Data Firehose, de modo que seu fluxo de dados e o fluxo do Firehose não precisem estar na mesma região.

```
{
    "cloudwatch.emitMetrics": true,
    "kinesis.endpoint": "https://your/kinesis/endpoint",
    "firehose.endpoint": "https://your/firehose/endpoint",
    "flows": [
        {
            "filePattern": "/tmp/app1.log*",
            "kinesisStream": "yourkinesisstream"
        },
        {
            "filePattern": "/tmp/app2.log*",
            "deliveryStream": "yourfirehosedeliverystream"
        }
    ]
}
```

Para obter informações mais detalhadas sobre o uso do agente com o Amazon Kinesis Data Streams, consulte Writing to Amazon Kinesis Data Streams with Kinesis Agent.

Pré-processamento de dados com agentes

O agente pode pré-processar os registros analisados a partir dos arquivos monitorados antes de enviá-los ao fluxo do Firehose. É possível habilitar esse recurso adicionando a configuração `dataProcessingOptions` ao fluxo de arquivos. Um ou mais opções de processamento podem ser adicionadas e serão executadas na ordem especificada.

O agente oferece suporte às seguintes opções de processamento. Como o agente é de código aberto, é possível desenvolver e estender ainda mais suas opções de processamento. É possível baixar o agente em Kinesis Agent.

Opções de processamento

`SINGLELINE`

Converte um registro de várias linhas em um registro de única linha removendo caracteres de nova linha, e espaços à esquerda e à direita.

```
{
    "optionName": "SINGLELINE"
}
```

`CSVTOJSON`

Converte um registro com formato separado por delimitador em um registro com formato JSON.

```
{
    "optionName": "CSVTOJSON",
    "customFieldNames": [ "field1", "field2", ... ],
    "delimiter": "yourdelimiter"
}
```

`customFieldNames`

[Obrigatório] Os nomes de campo usados como chaves em cada par de valores de chave JSON. Por exemplo, se você especificar `["f1", "f2"]`, o registro "v1, v2" será convertido em `{"f1":"v1","f2":"v2"}`.

`delimiter`

A string usada como delimitador no registro. O padrão é uma vírgula (,).

LOGTOJSON

Converte um registro com formato de log em um registro com formato JSON. Os formatos de log com suporte são Apache Common Log, Apache Combined Log, Apache Error Log e RFC3164 Syslog.

```
{
    "optionName": "LOGTOJSON",
    "logFormat": "logformat",
    "matchPattern": "yourregexpattern",
    "customFieldNames": [ "field1", "field2", … ]
}
```

logFormat

[Obrigatório] O formato da entrada de log. Os valores possíveis são:

- `COMMONAPACHELOG`: o formato do Apache Common Log. Cada entrada de log tem o seguinte padrão: `"%{host} %{ident} %{authuser} [%{datetime}] \"%{request}\" %{response} %{bytes}"`.
- `COMBINEDAPACHELOG`: o formato do Apache Combined Log. Cada entrada de log tem o seguinte padrão: `"%{host} %{ident} %{authuser} [%{datetime}] \"%{request}\" %{response} %{bytes} %{referrer} %{agent}"`.
- `APACHEERRORLOG`: o formato do Apache Error Log. Cada entrada de log tem o seguinte padrão: `"[%{timestamp}] [%{module}:%{severity}] [pid %{processid}:tid %{threadid}] [client: %{client}] %{message}"`.
- `SYSLOG`— O formato RFC3164 Syslog. Cada entrada de log tem o seguinte padrão: `"%{timestamp} %{hostname} %{program}[%{processid}]: %{message}"`.

matchPattern

Substitui o padrão do formato de log especificado. Use esta configuração para extrair valores de entradas de log, caso elas tenham um formato personalizado. Se você especificar `matchPattern`, também deverá especificar `customFieldNames`.

customFieldNames

Os nomes de campo personalizados usados como chaves em cada par de valores de chave JSON. É possível usar essa configuração para definir nomes de campo para valores extraídos de `matchPattern` ou substituir os nomes de campo padrão de formatos de log predefinidos.

Example : Configuração LOGTOJSON

Este é um exemplo de uma configuração `LOGTOJSON` para uma entrada Apache Common Log convertida em formato JSON:

```
{
    "optionName": "LOGTOJSON",
    "logFormat": "COMMONAPACHELOG"
}
```

Antes da conversão:

```
64.242.88.10 - - [07/Mar/2004:16:10:02 -0800] "GET /mailman/listinfo/hsdivision
 HTTP/1.1" 200 6291
```

Depois da conversão:

```
{"host":"64.242.88.10","ident":null,"authuser":null,"datetime":"07/
Mar/2004:16:10:02 -0800","request":"GET /mailman/listinfo/hsdivision
 HTTP/1.1","response":"200","bytes":"6291"}
```

Example : Configuração LOGTOJSON com campos personalizados

Este é outro exemplo de configuração `LOGTOJSON`:

```
{
    "optionName": "LOGTOJSON",
    "logFormat": "COMMONAPACHELOG",
    "customFieldNames": ["f1", "f2", "f3", "f4", "f5", "f6", "f7"]
}
```

Com essa configuração, a mesma entrada Apache Common Log do exemplo anterior é convertida em formato JSON, da seguinte forma:

```
{"f1":"64.242.88.10","f2":null,"f3":null,"f4":"07/Mar/2004:16:10:02 -0800","f5":"GET /
mailman/listinfo/hsdivision HTTP/1.1","f6":"200","f7":"6291"}
```

Example : Conversão da entrada Apache Common Log

A configuração de fluxo a seguir converte uma entrada Apache Common Log em um registro de linha única no formato JSON:

```
{
    "flows": [
        {
            "filePattern": "/tmp/app.log*",
            "deliveryStream": "my-delivery-stream",
            "dataProcessingOptions": [
                {
                    "optionName": "LOGTOJSON",
                    "logFormat": "COMMONAPACHELOG"
                }
            ]
        }
    ]
}
```

Example : Conversão de registros de várias linhas

A configuração de fluxo a seguir analisa registros de várias linha cuja primeira linha começa com "`[SEQUENCE=`". Cada registro é convertido primeiro em um registro de única linha. Em seguida, os valores são extraídos do registro com base em um delimitador por tabulações. Os valores extraídos são mapeados para os valores `customFieldNames` especificados, a fim de formar um registro de linha única no formato JSON.

```
{
    "flows": [
        {
            "filePattern": "/tmp/app.log*",
            "deliveryStream": "my-delivery-stream",
            "multiLineStartPattern": "\\[SEQUENCE=",
            "dataProcessingOptions": [
                {
                    "optionName": "SINGLELINE"
                },
                {
                    "optionName": "CSVTOJSON",
                    "customFieldNames": [ "field1", "field2", "field3" ],
                    "delimiter": "\\t"
                }
            ]
        }
    ]
}
```

Example : Configuração LOGTOJSON com padrão de correspondência

este é um exemplo de configuração `LOGTOJSON` referente a uma entrada Apache Common Log convertida em formato JSON, com o último campo (bytes) omitido:

```
{
    "optionName": "LOGTOJSON",
    "logFormat": "COMMONAPACHELOG",
    "matchPattern": "^([\\d.]+) (\\S+) (\\S+) \\[([\\w:/]+\\s[+\\-]\\d{4})\\] \"(.
+?)\" (\\d{3})",
    "customFieldNames": ["host", "ident", "authuser", "datetime", "request",
 "response"]
}
```

Antes da conversão:

```
123.45.67.89 - - [27/Oct/2000:09:27:09 -0400] "GET /java/javaResources.html HTTP/1.0"
 200
```

Depois da conversão:

```
{"host":"123.45.67.89","ident":null,"authuser":null,"datetime":"27/Oct/2000:09:27:09
 -0400","request":"GET /java/javaResources.html HTTP/1.0","response":"200"}
```

Uso de comandos comuns da CLI do agente

A tabela a seguir fornece um conjunto de casos de uso comuns e comandos correspondentes para trabalhar com o agente AWS Kinesis.

Caso de uso	Command
Inicie automatic amente o agente durante a inicialização do sistema	`sudo chkconfig aws-kinesis-agent on`
Verifique o status do agente	`sudo service aws-kinesis-agent status`

Caso de uso	Command
Interrompa o agente	`sudo service aws-kinesis-agent stop`
Leia o arquivo de log do agente a partir deste local	`/var/log/aws-kinesis-agent/aws-kinesis-agent.log`
Desinstale o agente	`sudo yum remove aws-kinesis-agent`

Solução de problemas ao enviar do agente do Kinesis

Esta tabela fornece informações sobre solução de problemas e soluções para problemas comuns enfrentados ao usar o agente do Amazon Kinesis.

Problema	Solução
Por que o agente do Kinesis não funciona no Windows?	O Kinesis Agent para Windows é um software diferente das plataformas do Kinesis Agent para Linux.
Por que o Kinesis Agent está ficando mais lento e/ou aumentando os `RecordSendErrors` ?	Isso geralmente ocorre devido ao controle de utilização do Kinesis. Verifique a métrica `WriteProvisionedThroughputExceeded` do Kinesis Data Streams ou a métrica `ThrottledRecords` dos fluxos do Firehose. Qualquer aumento de 0 nessas métricas indica que os limites do fluxo precisam ser aumentados. Para obter mais informações, consulte Limites do Kinesis Data Stream e Fluxos do Firehose. Depois de descartar o controle de utilização como causa, verifique se o Kinesis Agent está configurado para seguir um número grande de arquivos pequenos. Há um atraso quando o Kinesis Agent exibe os dados do final de um arquivo novo, portanto, o Kinesis Agent deveria estar exibindo os dados do final de um pequeno número de

Problema	Solução
	arquivos maiores. Tente consolidar os arquivos de log em arquivos maiores.
Como resolver as exceções `java.lang.OutOfMemoryError` ?	Isso ocorre quando o agente do Kinesis não tem memória suficiente para lidar com a workload atual. Tente aumentar `JAVA_START_HEAP` e `JAVA_MAX_HEAP` no `/usr/bin/start-aws-kinesis-agent` e reiniciar o agente.
Como resolver as exceções `IllegalStateException : connection pool shut down` ?	O Kinesis Agent não tem conexões suficientes para lidar com a workload atual. Tente aumentar `maxConnections` e `maxSendingThreads` nas configurações gerais do agente em `/etc/aws-kinesis/agent.json` . O valor padrão para esses campos é 12 vezes o número de processadores de runtime disponíveis. Consulte AgentConfiguration.java para saber mais sobre as configurações avançadas do agente.
Como posso depurar outro problema com o Kinesis Agent?	Os logs do nível `DEBUG` podem ser habilitados em `/etc/aws-kinesis/log4j.xml` .
Como devo configurar o Kinesis Agent?	Quanto menor o `maxBufferSizeBytes` , mais frequentemente o Kinesis Agent enviará dados. Isso pode ser bom, pois diminui o tempo de entrega dos registros, mas também aumenta as solicitações por segundo feitas ao Kinesis.
Por que o Kinesis Agent está enviando registros duplicados?	Isso ocorre devido a uma configuração incorreta da exibição dos dados do final dos arquivos. Certifique-se de que cada `fileFlow's filePattern` corresponda a apenas um arquivo. Isso também pode ocorrer se o modo `logrotate` que está sendo usado estiver no modo `copytruncate` . Tente mudar o modo para o modo padrão ou criar para evitar duplicações. Para obter mais informações sobre como lidar com registros duplicados, consulte Handling Duplicate Records.

Envie dados com o AWS SDK

É possível usar a API do Amazon Data Firehose para enviar dados para um fluxo do Firehose usando o AWS SDK para Java, .NET, Node.js, Python ou Ruby. Se você estiver começando a usar o Amazon Data Firehose, dedique algum tempo para se familiarizar com os conceitos e a terminologia apresentados em O que é o Amazon Data Firehose?. Para obter mais informações, consulte Comece a desenvolver usando a Amazon Web Services.

Esses exemplos não representam um código pronto para produção, pois não verificam todas as exceções possíveis nem abrangem todas as considerações de segurança ou de performance possíveis.

A API Amazon Data Firehose oferece duas operações para enviar dados para seu stream do Firehose: e. PutRecordPutRecordBatch `PutRecord()`envia um registro de dados em uma chamada e `PutRecordBatch()` pode enviar vários registros de dados em uma chamada.

Operações de gravação única usando PutRecord

A inserção de dados exige apenas o nome do fluxo do Firehose e um buffer de bytes (<=1000 KB). Como o Amazon Data Firehose coloca vários registros em lote antes de carregar o arquivo no Amazon S3, talvez você queira adicionar um separador de registro. Para inserir dados em um registro por vez em um fluxo do Firehose, use o código a seguir:

```
PutRecordRequest putRecordRequest = new PutRecordRequest();
putRecordRequest.setDeliveryStreamName(deliveryStreamName);

String data = line + "\n";

Record record = new Record().withData(ByteBuffer.wrap(data.getBytes()));
putRecordRequest.setRecord(record);

// Put record into the DeliveryStream
firehoseClient.putRecord(putRecordRequest);
```

Para mais contexto de código, consulte o exemplo de código incluído no AWS SDK. Para obter informações sobre a sintaxe de solicitação e de resposta, consulte o tópico relevante em Operações de API do Firehose.

Operações de gravação em lote usando PutRecordBatch

A inserção de dados só exige o nome do fluxo do Firehose e uma lista de registros. Como o Amazon Data Firehose coloca vários registros em lote antes de carregar o arquivo no Amazon S3, talvez você queira adicionar um separador de registro. Para inserir registros de dados em lotes em um fluxo do Firehose, use o código a seguir:

```
PutRecordBatchRequest putRecordBatchRequest = new PutRecordBatchRequest();
putRecordBatchRequest.setDeliveryStreamName(deliveryStreamName);
putRecordBatchRequest.setRecords(recordList);

// Put Record Batch records. Max No.Of Records we can put in a
// single put record batch request is 500
firehoseClient.putRecordBatch(putRecordBatchRequest);

recordList.clear();
```

Para mais contexto de código, consulte o exemplo de código incluído no AWS SDK. Para obter informações sobre a sintaxe de solicitação e de resposta, consulte o tópico relevante em Operações de API do Firehose.

Enviar CloudWatch registros para o Firehose

CloudWatch Os eventos de registros podem ser enviados para o Firehose usando filtros de CloudWatch assinatura. Para obter mais informações, consulte Filtros de assinatura com o Amazon Data Firehose.

CloudWatch Os eventos de registros são enviados para o Firehose no formato gzip compactado. Se você quiser entregar eventos de log descompactados para destinos do Firehose, você pode usar o recurso de descompactação no Firehose para descompactar automaticamente os registros. CloudWatch

> **Important**
>
> Atualmente, o Firehose não suporta a entrega de CloudWatch registros para o destino do Amazon OpenSearch Service porque a Amazon CloudWatch combina vários eventos de log em um registro Firehose e o Amazon OpenSearch Service não pode aceitar vários eventos de log em um registro. Como alternativa, você pode considerar o uso do filtro de assinatura do Amazon OpenSearch Service em CloudWatch registros.

Descompactar registros CloudWatch

Se você estiver usando o Firehose para entregar CloudWatch registros e quiser entregar dados descompactados para o destino do stream do Firehose, use o Firehose Data Format Conversion (Parquet, ORC) ou o particionamento dinâmico. Você deve habilitar a descompactação para seu fluxo do Firehose.

Você pode ativar a descompressão usando o AWS Management Console, AWS Command Line Interface ou AWS SDKs.

> **Note**
>
> Se você ativar o recurso de descompressão em um stream, use esse stream exclusivamente para filtros de assinaturas do CloudWatch Logs, e não para Vended Logs. Se você ativar o recurso de descompressão em um stream usado para ingerir CloudWatch registros e registros vendidos, a ingestão de registros vendidos no Firehose falhará. Esse recurso de descompressão é somente para CloudWatch registros.

Extraia a mensagem após a descompressão dos registros CloudWatch

Ao habilitar a descompactação, você também tem a opção de habilitar a extração de mensagens. Ao usar a extração de mensagens, o Firehose filtra todos os metadados, como proprietário, grupo de registros, fluxo de registros e outros, dos registros descompactados do CloudWatch Logs e entrega somente o conteúdo dentro dos campos da mensagem. Se você estiver entregando dados para um destino do Splunk, deverá ativar a extração de mensagens para que o Splunk analise os dados. A seguir há exemplos de saídas após a descompactação com e sem extração de mensagens.

Figura 1: exemplo de saída após a descompactação sem extração de mensagens:

```
{
 "owner": "111111111111",
 "logGroup": "CloudTrail/logs",
 "logStream": "111111111111_CloudTrail/logs_us-east-1",
 "subscriptionFilters": [
 "Destination"
 ],
 "messageType": "DATA_MESSAGE",
 "logEvents": [
```

```
 {
 "id": "31953106606966983378809025079804211143289615424298221568",
 "timestamp": 1432826855000,
 "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root1\"}"
 },
 {
 "id": "31953106606966983378809025079804211143289615424298221569",
 "timestamp": 1432826855000,
 "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root2\"}"
 },
 {
 "id": "31953106606966983378809025079804211143289615424298221570",
 "timestamp": 1432826855000,
 "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root3\"}"
 }
 ]
}
```

Figura 2: exemplo de saída após a descompactação com extração de mensagens:

```
{"eventVersion":"1.03","userIdentity":{"type":"Root1"}
{"eventVersion":"1.03","userIdentity":{"type":"Root2"}
{"eventVersion":"1.03","userIdentity":{"type":"Root3"}
```

Habilitação da descompactação em um novo fluxo do Firehose a partir do console

Para habilitar a descompressão em um novo stream do Firehose usando o AWS Management Console

1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com /kinesis.
2. Escolha Amazon Data Firehose no painel de navegação.
3. Escolha Criar fluxo do Firehose.
4. Em Escolher fonte e destino

   Origem

   A fonte do seu fluxo do Firehose. Escolha uma das fontes a seguir:

- Direct PUT: escolha esta opção para criar um fluxo do Firehose no qual as aplicações de produção gravem diretamente. Para obter uma lista de serviços e agentes da AWS e dos serviços de código aberto integrados com o Direct PUT no Firehose, consulte esta seção.
- Fluxo do Kinesis: escolha esta opção para configurar um fluxo do Firehose que use um fluxo de dados do Kinesis como fonte de dados. Você então poderá usar o Firehose para ler dados com facilidade de um fluxo de dados do Kinesis existente e carregá-lo nos destinos. Para obter mais informações, consulte Gravação no Firehose usando o Kinesis Data Streams

Destination (Destino)

O destino do seu fluxo do Firehose. Escolha uma das seguintes opções:

- Amazon S3
- Splunk

5. Em Nome do fluxo do Firehose, insira um nome para seu fluxo.
6. (Opcional) Em Transformar registros:

- Na seção Descompactar registros de origem do Amazon CloudWatch Logs, escolha Ativar descompressão.
- Se você quiser usar a extração de mensagens após a descompactação, escolha Ativar extração de mensagens.

Ativar a descompactação em um fluxo do Firehose existente

Esta seção fornece instruções para ativar a descompressão em fluxos Firehose existentes. Ele abrange dois cenários: fluxos com processamento Lambda desativado e fluxos com processamento Lambda já habilitado. As seções a seguir descrevem step-by-step os procedimentos para cada caso, incluindo a criação ou modificação das funções do Lambda, a atualização das configurações do Firehose e as métricas de CloudWatch monitoramento para garantir a implementação bem-sucedida do recurso de descompressão integrado do Firehose.

Ativando a descompressão quando o processamento do Lambda está desativado

Para habilitar a descompressão em um stream Firehose existente com o processamento Lambda desativado, você deve primeiro habilitar o processamento Lambda. Essa condição só é válida para fluxos existentes. As etapas a seguir mostram como habilitar a descompressão em fluxos existentes que não têm o processamento Lambda ativado.

1. Crie uma função do Lambda. Você pode criar uma passagem de registro fictícia ou usar esse esquema para criar uma nova função Lambda.
2. Atualize seu stream atual do Firehose para habilitar o processamento do Lambda e usar a função Lambda que você criou para processamento.
3. Depois de atualizar o stream com a nova função Lambda, volte ao console Firehose e ative a descompressão.
4. Desative o processamento do Lambda que você ativou na etapa 1. Agora você pode excluir a função que você criou na etapa 1.

Habilitando a descompressão quando o processamento Lambda está ativado

Se você já tem um stream do Firehose com uma função Lambda, para realizar a descompressão, você pode substituí-lo pelo recurso de descompressão do Firehose. Antes de continuar, revise o código da função do Lambda para confirmar se ela só executa a descompactação ou a extração de mensagens. A saída da função Lambda deve ser semelhante aos exemplos mostrados na Figura 1 ou na Figura 2. Se a saída for semelhante, será possível substituir a função do Lambda usando as etapas a seguir.

1. Substitua sua função do Lambda atual por este esquema. A nova função do Lambda do esquema detecta automaticamente se os dados recebidos estão compactados ou descompactados. Ela só executará a descompactação se os dados de entrada estiverem compactados.
2. Ative a descompactação usando a opção integrada do Firehose para descompactação.
3. Ative CloudWatch as métricas para seu stream do Firehose, caso ainda não esteja ativado. Monitore a métrica `CloudWatchProcessorLambda_IncomingCompressedData` e espere até que essa métrica mude para zero. Isso confirma que todos os dados de entrada enviados para sua função do Lambda estão descompactados e que a função do Lambda não é mais necessária.
4. Remova a transformação de dados do Lambda, pois você não precisará mais dela para descompactar seu fluxo.

Desabilitação da descompactação no fluxo do Firehose

Para desativar a descompressão em um fluxo de dados usando o AWS Management Console

1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com /kinesis.
2. Escolha Amazon Data Firehose no painel de navegação.
3. Escolha o fluxo do Firehose que deseja editar.
4. Na página de Detalhes do fluxo do Firehose, escolha a guia Configuração.
5. Na seção Transformar e converter registros, escolha Editar.
6. Em Descompactar registros de origem do Amazon CloudWatch Logs, desmarque Ativar descompressão e escolha Salvar alterações.

Solução de problemas de descompactação no Firehose

A tabela a seguir mostra como o Firehose lida com erros durante a descompactação e o processamento de dados, incluindo a entrega de registros para um bucket de erros do S3, o registro em log de erros e a emissão de métricas. Ela também explica a mensagem de erro retornada em operações não autorizadas de colocação de dados.

Problema	Solução
O que acontece com os dados da fonte em caso de erro durante a descompactação?	Se o Amazon Data Firehose não conseguir descompactar o registro, o registro será entregue como está (em formato compactado) para o bucket de erros do S3 que você especificou durante a criação do fluxo do Firehose. Junto com o registro, o objeto entregue também inclui código de erro e mensagem de erro e esses objetos serão entregues a um prefixo de bucket do S3 chamado `decompression-failed` . O Firehose continuará processando outros registros após uma falha na descompactação de um registro.
O que acontece com os dados da fonte em caso de erro no pipeline de processamento após a descompactação com êxito?	Se o Amazon Data Firehose cometer erros nas etapas de processamento após a descompactação, como o particionamento dinâmico e a conversão de formato de dados, o registro será entregue em formato compactado para o bucket de erros do S3 que você especificou durante a criação do fluxo do Firehose. Junto com o registro,

Problema	Solução
	o objeto entregue também inclui o código de erro e a mensagem de erro.
Como você é informado em caso de erro ou exceção?	Em caso de erro ou exceção durante a descompac tação, se você configurar o Logs, o Firehose CloudWatc h registrará as mensagens CloudWatch de erro no Logs. Além disso, o Firehose envia métricas para CloudWatc h métricas que você pode monitorar. Opcionalmente, você também pode criar alarmes baseado nas métricas emitidas pelo Firehose.
O que acontece quando `put` as operações não vêm do CloudWatch Logs?	Quando `puts` o cliente não vem do CloudWatch Logs, a seguinte mensagem de erro é retornada: `Put to Firehose failed for AccountId: <accountI D>, FirehoseName: <firehosename> because the request is not originating from allowed source types.`
Quais métricas o Firehose emite para o atributo da descompactação?	O Firehose emite métricas para descompactação de todos os registros. Você deve selecionar o período (1 min), a estatística (soma), o intervalo de datas para obter o número de `DecompressedRecords` com falha ou êxito ou `DecompressedBytes` com falha ou êxito. Para obter mais informações, consulte CloudWatch Métricas de descompressão de logs.

Enviar CloudWatch eventos para Firehose

Você pode configurar CloudWatch a Amazon para enviar eventos para um stream do Firehose adicionando um destino a uma regra de CloudWatch eventos.

Para criar um destino para uma regra de CloudWatch eventos que envia eventos para um stream existente do Firehose

1. Faça login no AWS Management Console e abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. Escolha Criar regra.
3. Na página Etapa 1: criar regra, em Destinos, selecione Adicionar destino, e, em seguida, Fluxo do Firehose.
4. Escolha um fluxo do Firehose existente.

Para obter mais informações sobre a criação de regras de CloudWatch eventos, consulte Getting Started with Amazon CloudWatch Events.

Configure AWS IoT para enviar dados para o Firehose

Você pode configurar AWS IoT para enviar informações para um stream do Firehose adicionando uma ação.

Para criar uma ação que envie eventos para um fluxo do Firehose existente

1. Ao criar uma regra no AWS IoT console, na página Criar uma regra, em Definir uma ou mais ações, escolha Adicionar ação.
2. Escolha Enviar mensagens para um fluxo do Amazon Kinesis Firehose.
3. Escolha Configurar ação.
4. Em Nome do fluxo, escolha um fluxo do Firehose existente.
5. Em Separator, escolha um caractere separador a ser inserido entre os registros.
6. Em Nome do perfil do IAM, escolha um perfil do IAM ou escolha Criar um novo perfil.
7. Selecione Adicionar ação.

Para obter mais informações sobre a criação de AWS IoT regras, consulte Tutoriais de regras de AWS IoT.

Transformação de dados da fonte no Amazon Data Firehose

O Amazon Data Firehose pode invocar a função do Lambda para transformar os dados da fonte de entrada e entregar os dados transformados aos destinos. É possível habilitar a transformação de dados do Amazon Data Firehose ao criar o fluxo do Firehose.

Noções básicas sobre o fluxo de transformação de dados

Quando você habilita a transformação de dados do Firehose, ele armazena os dados recebidos em buffer. A sugestão de tamanho para armazenamento em buffer varia de 0,2 MB a 3 MB. A sugestão de tamanho para armazenamento em buffer padrão do Lambda é de 1 MB para todos os destinos, exceto o Splunk e o Snowflake. Para o Splunk e o Snowflake, a sugestão de armazenando em buffer padrão é de 256 KB. A sugestão de intervalo de armazenamento em buffer do Lambda varia entre 0 e 900 segundos. A sugestão de intervalo de armazenamento buffer padrão do Lambda é de sessenta segundos para todos os destinos, exceto o Snowflake. Para o Snowflake, a sugestão de intervalo de armazenando em buffer padrão é de 30 segundos. Para ajustar o tamanho do buffer, defina o ProcessingConfigurationparâmetro da UpdateDestinationAPI CreateDeliveryStreamou com o ProcessorParameterchamado e. `BufferSizeInMBs` `IntervalInSeconds` O Firehose então invoca a função Lambda especificada de forma síncrona com cada lote armazenado em buffer usando o modo de invocação síncrona. AWS Lambda Os dados transformados são enviados do Lambda para o Firehose. Em seguida, o Firehose os envia o para o destino quando o tamanho ou o intervalo de buffer de destino especificado é atingido, o que acontecer primeiro.

> **Important**
>
> O modo de invocação síncrona do Lambda tem um limite de tamanho de carga útil de 6 MB para ambas a solicitação e a resposta. Certifique-se de que o tamanho do armazenamento em buffer para envio da solicitação para a função seja menor que ou igual a 6 MB. Além disso, verifique se a resposta que sua função retorna não excede 6 MB.

Duração da invocação do Lambda

O Amazon Data Firehose oferece suporte a um tempo de invocação do Lambda de até 5 minutos. Se sua função do Lambda levar mais de 5 minutos para ser concluída, você receberá o seguinte erro: O Firehose encontrou erros de tempo limite ao chamar o Lambda. AWS O tempo limite máximo da função é de 5 minutos.

Para obter informações sobre o que o Amazon Data Firehose faz se esse erro ocorrer, consulte the section called “Como lidar com falhas na transformação de dados ”.

Parâmetros necessários para transformação de dados

Todos os registros transformados do Lambda devem conter os parâmetros a seguir. Caso contrário, o Amazon Data Firehose os rejeitará e tratará esse evento como uma falha na transformação de dados.

For Kinesis Data Streams and Direct PUT

Os parâmetros a seguir são necessários para todos os registros transformados do Lambda.

- `recordId`: o ID do registro é transmitido do Amazon Data Firehose para o Lambda durante a invocação. O registro transformado deve conter o mesmo ID de registro. Qualquer incompatibilidade entre o ID do registro original e o ID do registro transformado é considerada uma falha na transformação de dados.
- `result`: o status da transformação de dados do registro. Os valores possíveis são: `Ok` (o registro foi transformado com êxito), `Dropped` (o registro foi removido intencionalmente pela lógica de processamento), e `ProcessingFailed` (não foi possível transformar o registro). Se um registro tiver um status de `Ok` ou `Dropped`, o Amazon Data Firehose considerará o processamento com êxito. Caso contrário, o Amazon Data Firehose considerará que o processamento do registro não teve êxito.
- `data`: a carga útil dos dados transformados, após a codificação base64.

  Este é um exemplo de saída de resultados do Lambda:

```
 {
    "recordId": "<recordId from the Lambda input>",
    "result": "Ok",
    "data": "<Base64 encoded Transformed data>"
}
```

For Amazon MSK

Os parâmetros a seguir são necessários para todos os registros transformados do Lambda.

- `recordId`: o ID do registro é transmitido do Firehose para o Lambda durante a invocação. O registro transformado deve conter o mesmo ID de registro. Qualquer incompatibilidade entre o ID do registro original e o ID do registro transformado é considerada uma falha na transformação de dados.
- `result`: o status da transformação de dados do registro. Os valores possíveis são: `Ok` (o registro foi transformado com êxito), `Dropped` (o registro foi removido intencionalmente pela lógica de processamento), e `ProcessingFailed` (não foi possível transformar o registro). Se um registro tiver o status `Ok` ou `Dropped`, o Firehose considerará que processamento teve êxito. Caso contrário, o Firehose considerará que o processamento do registro não teve êxito.
- `KafkaRecordValue`: a carga útil dos dados transformados, após a codificação base64.

  Este é um exemplo de saída de resultados do Lambda:

```
 {
    "recordId": "<recordId from the Lambda input>",
    "result": "Ok",
    "kafkaRecordValue": "<Base64 encoded Transformed data>"
}
```

Esquemas do Lambda com suporte

Esses esquemas demonstram como você pode criar e usar funções AWS Lambda para transformar dados em seus fluxos de dados do Amazon Data Firehose.

Para ver as plantas que estão disponíveis no console AWS Lambda

1. Faça login no AWS Management Console e abra o AWS Lambda console em https://console.aws.amazon.com/lambda/.
2. Selecione Create function (Criar função) e Use a blueprint (Usar um esquema).
3. No campo Esquemas, procure a palavra-chave `firehose` para encontrar os esquemas do Lambda do Amazon Data Firehose.

Lista de esquemas:

- Processamento de registros enviados para o fluxo do Amazon Data Firehose (Node.js, Python)

Este esquema mostra um exemplo básico de como processar dados em seu stream de dados do Firehose usando AWS o Lambda.

Data da versão mais recente: novembro de 2016.

Notas da versão: nenhuma.

- CloudWatch Registros do processo enviados para o Firehose

  Esse blueprint está obsoleto. Não use esse modelo. Isso pode gerar altas cobranças quando os dados de CloudWatch registros descompactados tiverem mais de 6 MB (limite Lambda). Para obter informações sobre o processamento de CloudWatch registros enviados para o Firehose, consulte Gravando no Firehose usando registros. CloudWatch
- Conversão de registros de fluxo do Amazon Data Firehose no formato syslog para JSON (Node.js)

  Este esquema mostra como você pode converter registros de entrada no formato RFC3164 Syslog em JSON.

  Data da versão mais recente: novembro de 2016.

  Notas da versão: nenhuma.

Para ver as plantas que estão disponíveis no AWS Serverless Application Repository

1. Acesse AWS Serverless Application Repository.
2. Escolha Procurar todas as aplicações.
3. No campo Aplicações, procure a palavra-chave `firehose`.

Também é possível criar uma função do Lambda sem usar um esquema. Consulte Introdução ao AWS Lambda.

Como lidar com falhas na transformação de dados

Se a invocação da função do Lambda falhar devido ao tempo limite da rede ou porque você atingiu o limite de invocações do Lambda, o Amazon Data Firehose repetirá a invocação três vezes, por padrão. Se a invocação não tiver êxito, o Amazon Data Firehose ignorará esse lote de registros. Os registros ignorados são tratados como registros com falha no processamento. Você pode especificar ou substituir as opções de nova tentativa usando a API

CreateDeliveryStreamouUpdateDestination. Para esse tipo de falha, você pode registrar erros de invocação no Amazon CloudWatch Logs. Para obter mais informações, consulte Monitoramento do Amazon Data Firehose usando logs CloudWatch .

Se o status da transformação de dados de um registro for `ProcessingFailed`, o Amazon Data Firehose considerará que houve falha no processamento do registro. Para esse tipo de falha, você pode emitir registros de erro para o Amazon CloudWatch Logs a partir da sua função Lambda. Para obter mais informações, consulte Como acessar o Amazon CloudWatch Logs AWS Lambda no Guia do AWS Lambda desenvolvedor.

Se uma transformação de dados falhar, os registros processados sem sucesso serão entregues ao seu bucket do S3 na `processing-failed` pasta. Os registros têm o seguinte formato:

```
{
    "attemptsMade": "count",
    "arrivalTimestamp": "timestamp",
    "errorCode": "code",
    "errorMessage": "message",
    "attemptEndingTimestamp": "timestamp",
    "rawData": "data",
    "lambdaArn": "arn"
}
```

`attemptsMade`

O número de tentativas de solicitações de invocação.

`arrivalTimestamp`

A hora em que o registro foi recebido pelo Amazon Data Firehose.

`errorCode`

O código de erro de HTTP retornado pelo Lambda.

`errorMessage`

A mensagem de erro retornada pelo Lambda.

`attemptEndingTimestamp`

O momento em que o Amazon Data Firehose parou de tentar as invocações do Lambda.

`rawData`

Os dados de registro com codificação base64.

`lambdaArn`

O nome do recurso da Amazon (ARN) da função do Lambda.

Faça backup dos registros de origem

O Amazon Data Firehose pode fazer backup de todos os registros não transformados no bucket do S3 simultaneamente enquanto entrega os registros transformados ao destino. É possível habilitar o período de retenção de backup de registro da fonte ao criar ou atualizar seu fluxo do Firehose. Não é possível desabilitar o período de retenção de backup do registro de origem após habilitá-lo.

Partição de dados de streaming no Amazon Data Firehose

O particionamento dinâmico permite particionar continuamente os dados de streaming no Firehose usando chaves dentro dos dados (por exemplo, `customer_id` ou `transaction_id`) e depois entregando os dados agrupados por essas chaves nos prefixos correspondentes do Amazon Simple Storage Service (Amazon S3). Isso facilita a execução de análises econômicas e de alto desempenho em dados de streaming no Amazon S3 usando vários serviços, como Amazon Athena, Amazon EMR, Amazon Redshift Spectrum e Amazon. QuickSight Além disso, o AWS Glue pode realizar trabalhos mais sofisticados de extração, transformação e carregamento (ETL) depois que os dados de streaming particionados dinamicamente são entregues ao Amazon S3, em casos de uso em que é necessário processamento adicional.

Particionar os dados minimiza a quantidade de dados digitalizados, otimiza a performance e reduz os custos de consultas de análise no Amazon S3. Também aumenta o acesso granular aos dados. Os fluxos do Firehose são tradicionalmente usados para capturar e carregar dados no Amazon S3. Para particionar um conjunto de dados em streaming para análises baseadas no Amazon S3, você precisaria executar aplicações de particionamento entre buckets do Amazon S3 antes de disponibilizar os dados para análise, o que pode se tornar complicado ou caro.

Com o particionamento dinâmico, o Firehose agrupa continuamente os dados em trânsito usando chaves de dados definidas de forma dinâmica ou estática e entrega os dados a prefixos individuais do Amazon S3 por chave. Isso reduz time-to-insight em minutos ou horas. Também reduz os custos e simplifica as arquiteturas.

Tópicos

Habilitação do particionamento dinâmico no Amazon Data Firehose

Você pode configurar o particionamento dinâmico para seus streams do Firehose por meio do Amazon Data Firehose Management Console, da CLI ou do. APIs

> **Important**
>
> Você só pode habilitar o particionamento dinâmico ao criar um novo fluxo do Firehose. Você não pode habilitar o particionamento dinâmico em um fluxo do Firehose existente que não tenha o particionamento dinâmico já habilitado.

Para ver as etapas detalhadas de como habilitar e configurar o particionamento dinâmico usando o console de gerenciamento do Firehose ao criar um novo fluxo do Firehose, consulte Criação de um fluxo do Amazon Firehose. Ao concluir a tarefa de especificar o destino do seu stream do Firehose, certifique-se de seguir as etapas na Definição de configurações do destino seção, pois atualmente, o particionamento dinâmico só é suportado para streams do Firehose que usam o Amazon S3 como destino.

Depois que o particionamento dinâmico for habilitado em um fluxo do Firehose ativo, será possível atualizar a configuração adicionando, removendo ou atualizando chaves de particionamento e expressões de prefixo do S3. Depois de atualizado, o Firehose começará a usar as novas chaves e as novas expressões de prefixo do S3.

> **Important**
>
> Depois que você habilitar o particionamento dinâmico em um fluxo do Firehose, ele não poderá ser desabilitado nesse fluxo.

Noções básicas de chaves de particionamento

Com o particionamento dinâmico, você cria conjuntos de dados direcionados a partir dos dados do S3 em streaming particionando os dados com base em chaves de particionamento. As chaves de particionamento permitem que você filtre os dados em streaming com base em valores específicos. Por exemplo, se você precisar filtrar os dados com base no ID do cliente e no país, poderá especificar o campo de dados de `customer_id` como uma chave de particionamento e o campo de dados de `country` como outra chave de particionamento. Em seguida, você especifica as expressões (usando os formatos com suporte) para definir os prefixos de bucket do S3 aos quais os registros de dados particionados dinamicamente devem ser entregues.

É possível criar chaves de particionamento com os métodos a seguir.

- Análise em linha: esse método usa o mecanismo de suporte integrado do Firehose, um analisador jq, para extrair as chaves para particionamento dos registros de dados que estão no formato JSON. Atualmente, oferecemos suporte apenas à versão `jq 1.6`.
- AWS Função Lambda — esse método usa uma AWS Lambda função especificada para extrair e retornar os campos de dados necessários para o particionamento.

> **Important**
>
> Ao habilitar o particionamento dinâmico, você deve configurar pelo menos um desses métodos para particionar os dados. É possível configurar qualquer um desses métodos para especificar as chaves de particionamento ou ambos ao mesmo tempo.

Criação de chaves de particionamento com análise em linha

Para configurar a análise em linha como o método de particionamento dinâmico para os dados em streaming, você deve escolher os parâmetros de registro de dados a serem usados como chaves de particionamento e fornecer um valor para cada chave de particionamento especificada.

O exemplo de registro de dados a seguir mostra como é possível definir chaves de particionamento para ele com análise em linha. Observe que os dados devem ser codificados no formato Base64. Você também pode consultar o exemplo da CLI.

```
{
   "type": {
    "device": "mobile",
    "event": "user_clicked_submit_button"
  },
  "customer_id": "1234567890",
  "event_timestamp": 1565382027,   #epoch timestamp
  "region": "sample_region"
}
```

Por exemplo, é possível escolher particionar os dados com base no parâmetro `customer_id` ou no parâmetro `event_timestamp`. Isso significa que você deseja que o valor do parâmetro `customer_id` ou do parâmetro `event_timestamp` em cada registro seja usado para determinar o prefixo do S3 ao qual o registro deve ser entregue. Você também pode escolher um parâmetro

aninhado, como `device` com uma expressão `.type.device`. A lógica de particionamento dinâmico pode depender de vários parâmetros.

Depois de selecionar os parâmetros dos dados para as chaves de particionamento, você mapeia cada parâmetro para uma expressão jq válida. A tabela a seguir mostra esse mapeamento de parâmetros para expressões jq:

Parameter	Expressão jq
`customer_id`	.customer_id
`device`	.type.device
`year`	.event_timestamp| strftime("%Y")
`month`	.event_timestamp| strftime("%m")
`day`	.event_timestamp| strftime("%d")
`hour`	.event_timestamp| strftime("%H")

No runtime, o Firehose usa a coluna direita acima para avaliar os parâmetros com base nos dados de cada registro.

Criação de chaves de particionamento com uma função do AWS Lambda

Para registros de dados compactados ou criptografados, ou dados em qualquer formato de arquivo que não seja JSON, você pode usar a AWS Lambda função integrada com seu próprio código personalizado para descompactar, descriptografar ou transformar os registros a fim de extrair e retornar os campos de dados necessários para o particionamento. Essa é uma expansão da função do Lambda de transformação existente que está disponível atualmente com o Firehose. É possível transformar, analisar e retornar os campos de dados que podem ser usados para particionamento dinâmico usando a mesma função do Lambda.

Veja a seguir um exemplo de função do Lambda de processamento de fluxo do Firehose em Python que reproduz cada registro lido da entrada na saída e extrai as chaves de particionamento dos registros.

```
from __future__ import print_function
import base64
import json
import datetime

# Signature for all Lambda functions that user must implement
def lambda_handler(firehose_records_input, context):
    print("Received records for processing from DeliveryStream: " +
 firehose_records_input['deliveryStreamArn']
          + ", Region: " + firehose_records_input['region']
          + ", and InvocationId: " + firehose_records_input['invocationId'])

    # Create return value.
    firehose_records_output = {'records': []}

    # Create result object.
    # Go through records and process them

    for firehose_record_input in firehose_records_input['records']:
        # Get user payload
        payload = base64.b64decode(firehose_record_input['data'])
        json_value = json.loads(payload)

        print("Record that was received")
        print(json_value)
        print("\n")
        # Create output Firehose record and add modified payload and record ID to it.
        firehose_record_output = {}
        event_timestamp = datetime.datetime.fromtimestamp(json_value['eventTimestamp'])
        partition_keys = {"customerId": json_value['customerId'],
                          "year": event_timestamp.strftime('%Y'),
                          "month": event_timestamp.strftime('%m'),
                          "day": event_timestamp.strftime('%d'),
                          "hour": event_timestamp.strftime('%H'),
                          "minute": event_timestamp.strftime('%M')
                          }

        # Create output Firehose record and add modified payload and record ID to it.
        firehose_record_output = {'recordId': firehose_record_input['recordId'],
                                  'data': firehose_record_input['data'],
                                  'result': 'Ok',
                                  'metadata': { 'partitionKeys': partition_keys }}

        # Must set proper record ID
```

```
        # Add the record to the list of output records.

        firehose_records_output['records'].append(firehose_record_output)

    # At the end return processed records
    return firehose_records_output
```

Veja a seguir um exemplo de função do Lambda de processamento de fluxo do Firehose em Go que reproduz cada registro lido da entrada na saída e extrai as chaves de particionamento dos registros.

```
package main

import (
 "fmt"
 "encoding/json"
 "time"
 "strconv"

 "github.com/aws/aws-lambda-go/events"
 "github.com/aws/aws-lambda-go/lambda"
)

type DataFirehoseEventRecordData struct {
 CustomerId string `json:"customerId"`
}

func handleRequest(evnt events.DataFirehoseEvent) (events.DataFirehoseResponse, error)
 {

 fmt.Printf("InvocationID: %s\n", evnt.InvocationID)
 fmt.Printf("DeliveryStreamArn: %s\n", evnt.DeliveryStreamArn)
 fmt.Printf("Region: %s\n", evnt.Region)

 var response events.DataFirehoseResponse

 for _, record := range evnt.Records {
  fmt.Printf("RecordID: %s\n", record.RecordID)
  fmt.Printf("ApproximateArrivalTimestamp: %s\n", record.ApproximateArrivalTimestamp)

  var transformedRecord events.DataFirehoseResponseRecord
  transformedRecord.RecordID = record.RecordID
  transformedRecord.Result = events.DataFirehoseTransformedStateOk
```

```
  transformedRecord.Data = record.Data

  var metaData events.DataFirehoseResponseRecordMetadata
  var recordData DataFirehoseEventRecordData
  partitionKeys := make(map[string]string)

  currentTime := time.Now()
  json.Unmarshal(record.Data, &recordData)
  partitionKeys["customerId"] = recordData.CustomerId
  partitionKeys["year"] = strconv.Itoa(currentTime.Year())
  partitionKeys["month"] = strconv.Itoa(int(currentTime.Month()))
  partitionKeys["date"] = strconv.Itoa(currentTime.Day())
  partitionKeys["hour"] = strconv.Itoa(currentTime.Hour())
  partitionKeys["minute"] = strconv.Itoa(currentTime.Minute())
  metaData.PartitionKeys = partitionKeys
  transformedRecord.Metadata = metaData

  response.Records = append(response.Records, transformedRecord)
 }

 return response, nil
}

func main() {
 lambda.Start(handleRequest)
}
```

Uso do prefixo do bucket do Amazon S3 para entregar dados

Ao criar um fluxo do Firehose que use o Amazon S3 como destino, você deve especificar um bucket do Amazon S3 ao qual o Firehose deve entregar os dados. Os prefixos de bucket do Amazon S3 são usados para organizar os dados armazenados nos buckets do S3. Um prefixo de bucket do Amazon S3 é semelhante a um diretório que permite agrupar objetos semelhantes.

Com o particionamento dinâmico, os dados particionados são entregues nos prefixos especificados do Amazon S3. Se você não habilitar o particionamento dinâmico, a especificação de um prefixo de bucket do S3 para o fluxo do Firehose é opcional. Porém, se você escolher habilitar o particionamento dinâmico, deverá especificar os prefixos de bucket do S3 para os quais o Firehose fornecerá os dados particionados.

Em cada fluxo do Firehose em que você habilitar o particionamento dinâmico, o valor do prefixo de bucket do S3 consistirá em expressões com base nas chaves de particionamento especificadas para esse fluxo do Firehose. Usando novamente o exemplo de registro de dados acima, é possível criar o valor de prefixo do S3 a seguir, que consiste em expressões com base nas chaves de particionamento definidas acima:

```
"ExtendedS3DestinationConfiguration": {
"BucketARN": "arn:aws:s3:::my-logs-prod",
"Prefix": "customer_id=!{partitionKeyFromQuery:customer_id}/
    device=!{partitionKeyFromQuery:device}/
    year=!{partitionKeyFromQuery:year}/
    month=!{partitionKeyFromQuery:month}/
    day=!{partitionKeyFromQuery:day}/
    hour=!{partitionKeyFromQuery:hour}/"
}
```

O Firehose avalia a expressão acima no runtime. Ele agrupa os registros que correspondem à mesma expressão de prefixo S3 avaliada para um único conjunto de dados. Em seguida, o Firehose entrega cada conjunto de dados ao prefixo S3 avaliado. A frequência de entrega do conjunto de dados para o S3 é determinada pela configuração do buffer do fluxo do Firehose. Assim sendo, o registro neste exemplo é entregue à seguinte chave de objeto do S3:

```
s3://my-logs-prod/customer_id=1234567890/device=mobile/year=2019/month=08/day=09/
hour=20/my-delivery-stream-2019-08-09-23-55-09-a9fa96af-e4e4-409f-bac3-1f804714faaa
```

Para o particionamento dinâmico, você deve usar o seguinte formato de expressão no prefixo de bucket do S3: `!{namespace:value}`, em que o namespace pode ser `partitionKeyFromQuery`, `partitionKeyFromLambda` ou ambos. Se estiver usando análise em linha para criar as chaves de particionamento para os dados da fonte, você deverá especificar um valor de prefixo de bucket do S3 consistindo em expressões especificadas no seguinte formato: `"partitionKeyFromQuery:keyID"`. Se estiver usando função do AWS Lambda para criar as chaves de particionamento para os dados da fonte, você deverá especificar um valor de prefixo de bucket de S3 que consista em expressões especificadas no seguinte formato: `"partitionKeyFromLambda:keyID"`.

> **Note**
>
> Você também pode especificar o valor do prefixo do bucket do S3 usando o formato de estilo hive, por exemplo customer_id=! {partitionKeyFromConsulta: customer_ID}.

Para obter mais informações, consulte "Escolher o Amazon S3 como destino" em Criação de um fluxo do Firehose e Prefixos personalizados para objetos do Amazon S3.

Adição de um novo delimitador de linha ao entregar dados ao Amazon S3

É possível habilitar o Delimitador de nova linha para adicionar um novo delimitador de linha entre registros nos objetos que são entregues ao Amazon S3. Isso pode ser útil para analisar objetos no Amazon S3. Isso também é particularmente útil quando o particionamento dinâmico é aplicado a dados agregados, pois a desagregação de vários registros (que deve ser aplicada aos dados agregados antes que possam ser particionados dinamicamente) remove as linhas novas dos registros como parte do processo de análise.

Aplicação de particionamento dinâmico de dados agregados

É possível aplicar o particionamento dinâmico aos dados agregados (por exemplo, vários eventos, logs ou registros agregados em uma única chamada de API `PutRecord` e `PutRecordBatch`), mas esses dados devem primeiro ser desagregados. É possível desagregar os dados habilitando a desagregação de vários registros, ou seja, o processo de analisar os registros no fluxo do Firehose e separá-los.

A desagregação de vários registros pode ser do tipo `JSON`, o que significa que a separação dos registros é baseada em objetos JSON consecutivos. A desagregação também pode ser do tipo `Delimited`, o que significa que a separação dos registros é realizada com base em um delimitador personalizado especificado. Esse delimitador personalizado deve ser uma string codificada na base 64. Por exemplo, se quiser usar a string a seguir como seu delimitador personalizado `####`, você deverá especificá-la no formato codificado em base 64, ou seja, `IyMjIw==`. A desagregação de registros por JSON ou por delimitador é limitada a 500 por registro.

> **Note**
>
> Ao desagregar registros JSON, certifique-se de que sua entrada ainda seja apresentada no formato JSON com suporte. Os objetos JSON devem estar em uma única linha sem

delimitador ou somente delimitados por nova linha (JSONL). Uma matriz de objetos JSON não é uma entrada válida.
Estes são exemplos de entrada correta: `{"a":1}{"a":2}` and `{"a":1}\n{"a":2}`
Este é um exemplo da entrada incorreta: `[{"a":1}, {"a":2}]`

Com dados agregados, quando você habilita o particionamento dinâmico, o Firehose analisa os registros e procura objetos JSON válidos ou registros delimitados em cada chamada de API com base no tipo de desagregação de vários registros especificado.

Important

Se os dados forem agregados, o particionamento dinâmico só poderá ser aplicado se os dados primeiro forem desagregados.

Important

Quando você usa o atributo de transformação de dados no Firehose, a desagregação é aplicada antes da transformação de dados. Os dados que chegam ao Firehose são processados na ordem a seguir: Desagregação → Transformação de dados via Lambda → Chaves de particionamento.

Solução de problemas de particionamento dinâmico

Se o Amazon Data Firehose não puder analisar registros de dados no fluxo do Firehose ou não conseguir extrair as chaves de particionamento especificadas ou avaliar as expressões incluídas no valor do prefixo do S3, esses registros de dados serão entregues ao prefixo de bucket de erros do S3 que você deve especificar ao criar o fluxo do Firehose no qual habilita o particionamento dinâmico. O prefixo de bucket de erros do S3 contém todos os registros que o Firehose não consegue entregar ao destino especificado do S3. Esses registros são organizados de acordo com o tipo de erro. Junto com o registro, o objeto entregue também inclui informações sobre o erro para ajudar a entender e resolver esse erro.

Você deve especificar um prefixo de bucket de erros do S3 para um fluxo do Firehose se quiser habilitar o particionamento dinâmico para esse fluxo. Se você não quiser habilitar o particionamento

dinâmico para um fluxo do Firehose, a especificação de um prefixo de bucket de erros do S3 é opcional.

Dados de buffer para particionamento dinâmico

O Amazon Data Firehose armazena em buffer os dados de streaming recebidos até um determinado tamanho e por um determinado período antes de entregá-los aos destinos especificados. É possível configurar o tamanho do buffer e o intervalo do buffer ao criar novos fluxos do Firehose ou atualizar o tamanho do buffer e o intervalo do buffer nos fluxos do Firehose existentes. O tamanho do buffer é medido em MBs e o intervalo do buffer é medido em segundos.

> **Note**
>
> O atributo de armazenamento em buffer zero não está disponível para o particionamento dinâmico.

Quando o particionamento dinâmico está habilitado, o Firehose armazena internamente os registros que pertençam a uma determinada partição com base na sugestão de buffer configurada (tamanho e tempo) antes de entregar esses registros ao bucket do Amazon S3. Para entregar objetos com o tamanho máximo, o Firehose usa o armazenamento em buffer de vários estágios internamente. Portanto, o end-to-end atraso de um lote de registros pode ser 1,5 vezes o tempo de dica de buffer configurado. Isso afeta a atualização dos dados de um fluxo do Firehose.

A quantidade de partições ativas é o número total de partições ativas dentro do buffer de entrega. Por exemplo, se a consulta de particionamento dinâmico monta 3 partições por segundo e você tiver uma configuração de sugestão de buffer que aciona a entrega a cada 60 segundos, então, em média, você teria 180 partições ativas. Se o Firehose não puder entregar os dados em uma partição para um destino, essa partição será contada como ativa no buffer de entrega até que possa ser entregue.

Uma nova partição é criada quando um prefixo do S3 é avaliado como um novo valor com base nos campos de dados do registro e nas expressões do prefixo do S3. Um novo buffer é criado para cada partição ativa. Cada registro subsequente com o mesmo prefixo S3 avaliado é entregue a esse buffer.

Quando o buffer chega ao seu limite de tamanho ou ao fim de seu intervalo de tempo, o Firehose cria um objeto com os dados do buffer e o entrega ao prefixo especificado do Amazon S3. Depois que o

objeto é entregue, o buffer da partição e a própria partição são excluídos e removidos da contagem de partições ativas.

O Firehose entrega cada dado do buffer como um único objeto quando o tamanho ou o intervalo do buffer são atingidos para cada partição separadamente. Quando o número de partições ativas atinge o limite de 500 por stream do Firehose, o restante dos registros no stream do Firehose é entregue ao prefixo do bucket de erro do S3 especificado (). activePartitionExceeded É possível usar o formulário de limites do Amazon Data Firehose para solicitar um aumento dessa cota para até 5.000 partições ativas por cada fluxo do Firehose. Se você precisar de mais partições, será possível criar mais fluxos do Firehose e distribuir as partições ativas entre eles.

Conversão do formato de dados de entrada no Amazon Data Firehose

O Amazon Data Firehose pode converter o formato dos dados de entrada de JSON para Apache Parquet ou Apache ORC antes de armazenar os dados no Amazon S3. Parquet e ORC são formatos de dados em colunas que economizam espaço e permitem consultas mais rápidas do que com os formatos orientados a linhas, como JSON. Se você quiser converter um formato de entrada diferente de JSON, como valores separados por vírgula (CSV) ou texto estruturado, você pode usá-lo AWS Lambda para transformá-lo em JSON primeiro. Para obter mais informações, consulte Transformação de dados da fonte.

É possível converter o formato dos dados mesmo de agregar seus registros antes de enviá-los para o Amazon Data Firehose.

O Amazon Data Firehose exige estes três elementos para converter o formato de dados de registros:

Deserializer

O Amazon Data Firehose exige um desserializador para ler o JSON dos seus dados de entrada. Escolha um dos dois tipos a seguir de desserializador.

Ao combinar vários documentos JSON no mesmo registro, certifique-se de que sua entrada ainda seja apresentada no formato JSON compatível. Uma matriz de documentos JSON não é uma entrada válida.

Por exemplo, esta é a entrada correta: `{"a": 1}{"b": 1}` e esta é a entrada incorreta:`[{"a":1}, {"a":2}]`.

- Apache Hive JSON SerDe
- OpenX JSON SerDe

Escolha do desserializador JSON

Escolha o OpenX JSON SerDe se o JSON de entrada contiver carimbos de data e hora nos seguintes formatos:

- yyyy-MM-dd'T'hh:mm:ss [.S] 'Z', onde a fração pode ter até 9 dígitos — por exemplo,. `2017-02-07T15:13:01.39256Z`

- yyyy-[M]M-[d]d HH:mm:ss[.S], em que a fração pode ter até 9 dígitos, por exemplo, `2017-02-07 15:13:01.14`.
- Segundos a partir do ponto zero, por exemplo, `1518033528`.
- Milissegundos a partir do ponto zero, por exemplo, `1518033528123`.
- Segundos a partir do ponto zero com ponto flutuante, por exemplo, `1518033528.123`.

O OpenX JSON SerDe pode converter pontos (`.`) em sublinhados (). _ Ele também pode converter chaves JSON para minúsculas antes de desserializá-las. Para obter mais informações sobre as opções que estão disponíveis com esse desserializador por meio do Amazon Data Firehose, consulte Open. XJson SerDe

Se você não tiver certeza de qual desserializador escolher, use o OpenX JSON SerDe, a menos que tenha carimbos de data e hora que ele não suporta.

Se você tiver carimbos de data e hora em formatos diferentes dos listados anteriormente, use o Apache Hive JSON. SerDe Ao escolher esse desserializador, será possível especificar os formatos de carimbo de data/hora a serem usados. Para fazer isso, siga a sintaxe do padrão de string do formato Joda Time `DateTimeFormat`. Para obter mais informações, consulte Classe DateTimeFormat.

Você também pode usar o valor especial `millis` para analisar o time stamp em milissegundos de epoch. Se você não especificar um formato, o Amazon Data Firehose usará `java.sql.Timestamp::valueOf` por padrão.

O Hive JSON SerDe não permite o seguinte:

- Pontos (`.`) em nomes de coluna.
- Campos cujo tipo é `uniontype`.
- Campos que têm tipos numéricos no esquema, mas que são strings no JSON. Por exemplo, se o esquema for (um int) e o JSON for`{"a":"123"}`, o Hive apresentará um erro SerDe .

O Hive SerDe não converte JSON aninhado em strings. Por exemplo, se você tiver `{"a": {"inner":1}}`, ele não tratará `{"inner":1}` como uma string.

Schema

O Amazon Data Firehose exige um esquema para determinar como interpretar esses dados. Use o AWS Glue para criar um esquema no AWS Glue Data Catalog. Em seguida, o Amazon Data Firehose referencia esse esquema e usa-o para interpretar os dados de entrada. É possível usar o mesmo esquema para configurar o Amazon Data Firehose o software de análise. Para obter mais informações, consulte Preenchendo o catálogo de dados do AWS Glue no Guia do AWS Glue desenvolvedor.

> **Note**
>
> O esquema criado no Catálogo de AWS Glue Dados deve corresponder à estrutura de dados de entrada. Caso contrário, os dados convertidos não conterão atributos que não estejam especificados no esquema. Se você usar JSON aninhado, use um tipo STRUCT no esquema que espelha a estrutura dos dados JSON. Veja este exemplo para saber como lidar com JSON aninhado com um tipo STRUCT.

> **Important**
>
> Para tipos de dados que não especificam um limite de tamanho, há um limite prático de 32 MBs para todos os dados em uma única linha.
>
> Se você especificar o comprimento como `CHAR` ou `VARCHAR`, o Firehose truncará as cadeias de caracteres no comprimento especificado ao ler os dados de entrada. Se a string de dados subjacente for mais longa, ela permanecerá inalterada.

Serializer

O Firehose exige um serializador para converter dados para o formato de armazenamento em colunas de destino (Parquet ou ORC): escolha um dentre os dois tipos de serializadores a seguir.

- ORC SerDe
- Parquet SerDe

Escolha do serializador

O serializador que você escolhe depende de suas necessidades de negócios. Para saber mais sobre as duas opções de serializador, consulte ORC SerDe e Parquet. SerDe

Habilitar a conversão de formato do registro

Se você habilitar a conversão do formato de registro, não poderá definir o destino do Amazon Data Firehose como Amazon OpenSearch Service, Amazon Redshift ou Splunk. Com a conversão de formato habilitada, o Amazon S3 é o único destino que pode ser usado para o fluxo do Firehose. A seção a seguir mostra como ativar a conversão do formato de registro nas operações do console e da API do Firehose. Para obter um exemplo de como configurar a conversão do formato de registro com AWS CloudFormation, consulte AWS::DataFirehose:: DeliveryStream.

Habilitação da conversão de formato do registro a partir do console

É possível habilitar a conversão de formato de dados no console ao criar ou atualizar um fluxo do Firehose. Com a conversão de formato de dados habilitada, o Amazon S3 é o único destino que pode ser configurado para o fluxo do Firehose. Além disso, a compactação do Amazon S3 será desabilitada quando você habilitar a conversão de formato. No entanto, a compactação Snappy ocorre automaticamente como parte do processo de conversão. OK formato de enquadramento para o Snappy que o Amazon Data Firehose usa nesse caso é compatível com o Hadoop. Isso significa que é possível usar os resultados da compactação Snappy e executar consultas nesses dados no Athena. Para o formato de enquadramento Snappy no qual o Hadoop se baseia, consulte .java. BlockCompressorStream

Para ativar a conversão de formato de dados para um fluxo de dados do Firehose

1. Faça login no AWS Management Console e abra o console do Amazon Data Firehose em. https://console.aws.amazon.com/firehose/
2. Selecione um fluxo do Firehose para atualizar ou crie um novo fluxo do Firehose seguindo as etapas em Tutorial: Criação de um fluxo do Firehose a partir do console.
3. Em Convert record format (Converter formato do registro), defina Record format conversion (Conversão de formato do registro) como Enabled (Habilitado).
4. Selecione o formato de saída que você deseja. Para obter mais informações sobre as duas opções, consulte Apache Parquet e Apache ORC.

5. Escolha uma AWS Glue tabela para especificar um esquema para seus registros de origem. Defina a região, o banco de dados, a tabela e a versão da tabela.

Gerenciamento da conversão do formato de registro da API do Firehose

Se você quiser que o Amazon Data Firehose converta o formato dos seus dados de entrada de JSON para Parquet ou ORC, especifique o DataFormatConversionConfigurationelemento opcional em ExtendedS3 ou em ExtendedS3. DestinationConfiguration DestinationUpdate Se você especificar DataFormatConversionConfiguration, as seguintes restrições se aplicam.

- Em BufferingHints, você não pode `SizeInMBs` definir um valor menor que 64 se você habilitar a conversão do formato de registro. Além disso, quando a conversão de formato não está ativada, o valor padrão é 5. O valor se torna 128 quando você a habilita.
- Você deve definir `CompressionFormat` em ExtendedS3 DestinationConfiguration ou em ExtendedS3 como. DestinationUpdate `UNCOMPRESSED` O valor padrão para `CompressionFormat` é `UNCOMPRESSED`. Portanto, você também pode deixá-lo não especificado em DestinationConfigurationExtendedS3. Os dados ainda são compactados como parte do processo de serialização, usando a compactação Snappy, por padrão. OK formato de enquadramento para o Snappy que o Amazon Data Firehose usa nesse caso é compatível com o Hadoop. Isso significa que é possível usar os resultados da compactação Snappy e executar consultas nesses dados no Athena. Para o formato de enquadramento Snappy no qual o Hadoop se baseia, consulte .java. BlockCompressorStream Quando você configurar o serializador, será possível escolher outros tipos de compactação.

Tratamento de erros para conversão de formato de dados

Quando o Amazon Data Firehose não pode analisar ou desserializar um registro (por exemplo, quando os dados não correspondem ao esquema), ele o grava no Amazon S3 com um prefixo de erro. Se essa gravação falhar, o Amazon Data Firehose fará novas tentativas indefinidamente, bloqueando qualquer outra entrega. Para cada registro com falha, o Amazon Data Firehose grava um documento JSON com esquema a seguir:

```
{
  "attemptsMade": long,
  "arrivalTimestamp": long,
  "ErrorCode": string,
  "ErrorMessage": string,
```

```
  "attemptEndingTimestamp": long,
  "rawData": string,
  "sequenceNumber": string,
  "subSequenceNumber": long,
  "dataCatalogTable": {
    "catalogId": string,
    "databaseName": string,
    "tableName": string,
    "region": string,
    "versionId": string,
    "catalogArn": string
  }
}
```

Noções básicas sobre entrega de dados no Amazon Data Firehose

Quando você envia dados para o stream do Firehose, eles são automaticamente entregues ao destino escolhido. A tabela a seguir explica a entrega de dados para destinos diferentes.

Destino	Detalhes
Amazon S3	Para a entrega de dados ao Amazon S3, o Firehose concatena vários registros de entrada com base na configuração do armazenamento em buffer do seu fluxo do Firehose. Depois, entrega os ao Amazon S3; como um objeto do S3;. Por padrão, o Firehose concatena dados sem nenhum delimitador. Se quiser ter novos delimitadores de linha entre os registros, eles podem ser adicionados ativando o atributo na configuração do console Firehose ou no parâmetro da API. A entrega de dados entre o Firehose e o destino do Amazon S3 é criptografada com TLS (HTTPS).
Amazon Redshift	Para entrega de dados ao Amazon Redshift, o Firehose primeiro entrega os dados recebidos ao bucket do S3 no formato descrito anteriormente. Depois, o Firehose emite um comando COPY do Amazon Redshift para carregar os dados do bucket do S3 no cluster provisionado do Amazon Redshift ou no grupo de trabalho Amazon Redshift sem servidor. Certifique-se de que, após o Amazon Data Firehose concatenar vários registros recebidos em um objeto do Amazon S3, o objeto do Amazon S3 possa ser copiado para o cluster provisionado do Amazon Redshift ou para o grupo de trabalho do Amazon Redshift sem servidor. Para obter mais informações, consulte Parâmetros de formato de dados do comando COPY do Amazon Redshift.
OpenSearch Serviço e sem OpenSearch servidor	Para entrega de dados para OpenSearch Service e OpenSearch Serverless, o Amazon Data Firehose armazena registros de entrada com base na configuração de buffer do seu stream Firehose. Em seguida, ele gera uma solicitação em massa de OpenSearch serviço ou OpenSearch sem servidor para indexar vários registros em

Destino	Detalhes
	seu cluster de OpenSearch serviços ou coleção sem OpenSearch servidor. Certifique-se de que o registro esteja codificado em UTF-8 e reduzido a um objeto JSON de linha única antes de enviá-lo para o Amazon Data Firehose. Além disso, a `rest.action.multi.allow_explicit_index` opção para seu cluster de OpenSearch serviços deve ser definida como verdadeira (padrão) para receber solicitações em massa com um índice explícito definido por registro. Para obter mais informações, consulte OpenSearch Service Configure Advanced Options no Amazon OpenSearch Service Developer Guide.
Splunk	Para a entrega de dados ao Splunk, o Amazon Data Firehose concatena os bytes enviados por você. Se você quer delimitadores em seus dados, como um caractere de nova linha, deve inseri-los. Certifique-se de que o Splunk é configurado para analisar quaisquer delimitadores. Para redirecionar os dados que foram entregues ao bucket de erros do S3 (backup do S3) de volta ao Splunk, siga as etapas mencionadas na Documentação do Splunk.
Endpoint de HTTP	Para entrega de dados a um endpoint de HTTP de propriedade de um provedor de serviços terceirizado com suporte, é possível usar o serviço Amazon Lambda integrado para criar uma função para transformar os registros recebidos no formato que é esperado ela integração do provedor de serviços. Entre em contato com o provedor de serviços terceirizado cujo endpoint de HTTP você escolheu como destino para saber mais sobre o formato de registro que ele aceita.

Destino	Detalhes
Snowflake	Para entrega de dados ao Snowflake, o Amazon Data Firehose armazena internamente os dados em buffer por um segundo e usa as operações da API de streaming do Snowflake para inserir dados no Snowflake. Por padrão, os registros que você insere são liberados e confirmados na tabela do Snowflake a cada segundo. Depois de fazer a chamada de inserção, o Firehose emite uma CloudWatch métrica que mede quanto tempo levou para que os dados fossem confirmados no Snowflake. Atualmente, o Firehose oferece suporte a apenas um único item JSON como carga útil de registro, e não oferece suporte a matrizes JSON. Certifique-se de que sua carga útil de entrada seja um objeto JSON válido e esteja bem formada, sem aspas duplas, aspas ou caracteres de escape adicionais.

Cada destino do Firehose tem sua própria frequência de entrega de dados. Para obter mais informações, consulte Configuração de sugestões de armazenamento em buffer.

Registros duplicados

O Amazon Data Firehose usa at-least-once semântica para entrega de dados. Em algumas circunstâncias, como quando o tempo limite para entrega de dados é atingido, as novas tentativas de entrega feitas pelo Amazon Data Firehose poderão introduzir duplicatas se a solicitação de entrega de dados original acabar sendo atendida. Isso se aplica a todos os tipos de destino compatíveis com o Amazon Data Firehose, exceto destinos do Amazon S3, Apache Iceberg Tables e destinos do Snowflake.

Tópicos

- Entenda a entrega em todas AWS as contas e regiões
- Noções básicas das especificações de solicitação e resposta de entrega de endpoint de HTTP
- Como lidar com falhas de entrega de dados
- Configuração de formato de nome de objeto do Amazon S3
- Configurar a rotação do índice para o OpenSearch serviço
- Pausa e retomada da entrega de dados

Entenda a entrega em todas AWS as contas e regiões

O Amazon Data Firehose oferece suporte à entrega de dados para destinos de endpoints HTTP em todas as contas. AWS O stream do Firehose e o endpoint HTTP que você escolhe como destino podem pertencer a contas diferentes. AWS

O Amazon Data Firehose também oferece suporte à entrega de dados para destinos de endpoints HTTP em todas as regiões. AWS Você pode entregar dados de um stream do Firehose em uma AWS região para um endpoint HTTP em outra região. AWS Você também pode entregar dados de um stream do Firehose para um destino de endpoint HTTP fora das AWS regiões, por exemplo, para seu próprio servidor local, definindo a URL do endpoint HTTP como o destino desejado. Nesses cenários, taxas adicionais de transferência de dados são adicionadas aos seus custos de entrega. Para obter mais informações, consulte a seção Transferência de dados na página "Preços sob demanda".

Noções básicas das especificações de solicitação e resposta de entrega de endpoint de HTTP

Para que o Amazon Data Firehose entregue dados com êxito aos endpoints de HTTP personalizados, esses endpoints devem aceitar solicitações e enviar respostas usando determinados formatos de solicitação e resposta do Amazon Data Firehose. Esta seção descreve as especificações de formato das solicitações HTTP que o serviço Amazon Data Firehose envia para endpoints de HTTP personalizados, bem como as especificações de formato das respostas HTTP que o serviço Amazon Data Firehose espera. Os endpoints de HTTP têm 3 minutos para responder a uma solicitação antes que o Amazon Data Firehose atinja o tempo limite da solicitação. O Amazon Data Firehose trata as respostas que não seguem o formato adequado como falhas de entrega.

Formato de solicitação

Parâmetros de caminho e URL

Eles são configurados diretamente por você como parte de um único campo de URL. O Amazon Data Firehose os envia como foram configurados, sem modificação. Somente há suporte para destinos de https. As restrições de URL são aplicadas durante a configuração do fluxo de entrega.

> **Note**
>
> Atualmente, somente a porta 443 oferece suporte à entrega de dados de endpoint de HTTP.

Cabeçalhos HTTP - Versão X-Amz-Firehose-Protocol

Esse cabeçalho é usado para indicar a versão dos formatos de solicitação/resposta. Atualmente, a única versão é a 1.0.

Cabeçalhos HTTP - X-Amz-Firehose-Request -Id

O valor desse cabeçalho é um GUID opaco que pode ser usado para depuração e eliminação de duplicações. As implementações de endpoint devem registrar em log o valor desse cabeçalho, se possível, tanto para solicitações com êxito ou não. O ID da solicitação é mantido entre as várias tentativas da mesma solicitação.

Cabeçalhos HTTP: Content-Type

O valor do cabeçalho Content-Type é sempre `application/json`.

Cabeçalhos HTTP: Content-Encoding

Um fluxo do Firehose pode ser configurado para usar o GZIP para compactar o corpo das solicitações ao enviá-las. Quando essa compactação está habilitada, o valor do cabeçalho Content-Encoding é definido como gzip, de acordo com a prática padrão. Se a compactação não estiver habilitada, o cabeçalho Content-Encoding estará totalmente ausente.

Cabeçalhos HTTP: Content-Length

Isso é usado da maneira padrão.

Cabeçalhos HTTP - X-Amz-Firehose-Source -Arn:

O ARN do fluxo do Firehose representado no formato string ASCII. O ARN codifica a região, o ID da AWS conta e o nome do stream. Por exemplo, .`arn:aws:firehose:us-east-1:123456789:deliverystream/testStream`

Cabeçalhos HTTP - -Key X-Amz-Firehose-Access

Esse cabeçalho carrega uma chave de API ou outras credenciais. É possível criar ou atualizar a chave de API (também conhecida como token de autorização) ao criar ou atualizar seu fluxo de entrega. O Amazon Data Firehose restringe o tamanho da chave de acesso a 4.096 bytes.

O Amazon Data Firehose não tenta, de maneira nenhuma, interpretar essa chave. A chave configurada é copiada literalmente para o valor desse cabeçalho.

O conteúdo pode ser arbitrário e pode representar um token JWT ou uma ACCESS_KEY. Se um endpoint exigir credenciais com vários campos (por exemplo, nome de usuário e senha), os valores de todos os campos devem ser armazenados juntos em uma única chave de acesso em um formato que o endpoint entenda (JSON ou CSV). Esse campo pode ser codificado na base 64 se o conteúdo original for binário. O Amazon Data Firehose não modifica e/ou codifica o valor configurado e usa o conteúdo sem alterações.

Cabeçalhos HTTP - X-Amz-Firehose-Common -Atributos

Esse cabeçalho transporta os atributos comuns (metadados) relativos à solicitação inteira e/ou a todos os registros dentro da solicitação. Eles são configurados diretamente por você ao criar um fluxo do Firehose. O valor desse atributo é codificado como um objeto JSON com o seguinte esquema:

```
"$schema": http://json-schema.org/draft-07/schema#

properties:
  commonAttributes:
    type: object
    minProperties: 0
    maxProperties: 50
    patternProperties:
      "^.{1,256}$":
        type: string
        minLength: 0
        maxLength: 1024
```

Veja um exemplo abaixo:

```
"commonAttributes": {
    "deployment -context": "pre-prod-gamma",
    "device-types": ""
  }
```

Corpo: tamanho máximo

O tamanho máximo do corpo é configurado por você e pode ter até 64 MiB, antes de compactado.

Corpo: esquema

O corpo leva um único documento JSON com o seguinte esquema JSON (escrito em YAML):

```
"$schema": http://json-schema.org/draft-07/schema#

title: FirehoseCustomHttpsEndpointRequest
description: >
  The request body that the Firehose service sends to
  custom HTTPS endpoints.
type: object
properties:
  requestId:
    description: >
      Same as the value in the X-Amz-Firehose-Request-Id header,
      duplicated here for convenience.
    type: string
  timestamp:
    description: >
      The timestamp (milliseconds since epoch) at which the Firehose
      server generated this request.
    type: integer
  records:
    description: >
      The actual records of the Firehose stream, carrying
      the customer data.
    type: array
    minItems: 1
    maxItems: 10000
    items:
      type: object
      properties:
        data:
          description: >
            The data of this record, in Base64. Note that empty
            records are permitted in Firehose. The maximum allowed
            size of the data, before Base64 encoding, is 1024000
            bytes; the maximum length of this field is therefore
```

```
            1365336 chars.
          type: string
          minLength: 0
          maxLength: 1365336

required:
  - requestId
  - records
```

Veja um exemplo abaixo:

```
{
  "requestId": "ed4acda5-034f-9f42-bba1-f29aea6d7d8f",
  "timestamp": 1578090901599
  "records": [
    {
      "data": "aGVsbG8="
    },
    {
      "data": "aGVsbG8gd29ybGQ="
    }
  ]
}
```

Formato de resposta

Comportamento padrão em caso de erro

Se uma resposta não estiver em conformidade com os requisitos abaixo, o servidor do Firehose a tratará como se tivesse um código de status 500 sem corpo.

Código de status

O código de status do HTTP DEVE estar no intervalo 2XX, 4XX ou 5XX.

O servidor do Amazon Data Firehose NÃO segue redirecionamentos (códigos de status 3XX). Somente o código de resposta 200 é considerado uma entrega com êxito de registros para HTTP/EP. O código de resposta 413 (tamanho excedido) é considerado uma falha permanente, e o lote de registros não é enviado para o bucket de erros, se configurado. Todos os outros códigos de

resposta são considerados erros passíveis de novas tentativas e estão sujeitos ao algoritmo de novas tentativas de recuo que será explicado posteriormente.

Cabeçalhos HTTP: tipo de conteúdo

O único tipo de conteúdo aceitável é aplicação/json.

Cabeçalhos HTTP: Content-Encoding

A codificação de conteúdo NÃO DEVE ser usada. O corpo DEVE estar descompactado.

Cabeçalhos HTTP: Content-Length

O cabeçalho Content-Length DEVE estar presente se a resposta tiver um corpo.

Corpo: tamanho máximo

O corpo da resposta deve ter no máximo 1 MiB.

```
"$schema": http://json-schema.org/draft-07/schema#

title: FirehoseCustomHttpsEndpointResponse

description: >
  The response body that the Firehose service sends to
  custom HTTPS endpoints.
type: object
properties:
  requestId:
    description: >
      Must match the requestId in the request.
    type: string

  timestamp:
    description: >
      The timestamp (milliseconds since epoch) at which the
      server processed this request.
    type: integer

  errorMessage:
    description: >
      For failed requests, a message explaining the failure.
      If a request fails after exhausting all retries, the last
      Instance of the error message is copied to error output
      S3 bucket if configured.
```

```
    type: string
    minLength: 0
    maxLength: 8192
required:
  - requestId
  - timestamp
```

Veja um exemplo abaixo:

```
Failure Case (HTTP Response Code 4xx or 5xx)
{
  "requestId": "ed4acda5-034f-9f42-bba1-f29aea6d7d8f",
  "timestamp": "1578090903599",
  "errorMessage": "Unable to deliver records due to unknown error."
}
Success case (HTTP Response Code 200)
{
  "requestId": "ed4acda5-034f-9f42-bba1-f29aea6d7d8f",
  "timestamp": 1578090903599
}
```

Lidar com respostas de erro

Em todos os casos de erro, o servidor d Amazon Data Firehose faz uma nova tentativa de entrega do mesmo lote de registros usando um algoritmo de recuo exponencial. As novas tentativas são recuadas usando um tempo de recuo inicial (1 segundo) com um fator de instabilidade de (15%) e cada nova tentativa subsequente é recuada usando a fórmula (initial-backoff-time * (multiplicador (2) ^ retry_count)) com variação adicional. O tempo de recuo é limitado por um intervalo máximo de 2 minutos. Por exemplo, na 'n'-ésima tentativa, o tempo de recuo é = MAX(120, 2^n) * random(0,85, 1,15).

Os parâmetros especificados na equação anterior estão sujeitos a alterações. Consulte a documentação do AWS Firehose para ver o tempo exato de recuo inicial, o tempo máximo de recuo, o multiplicador e as porcentagens de instabilidade usadas no algoritmo de recuo exponencial.

Em cada nova tentativa subsequente, a chave de acesso e/ou o destino para o qual os registros são entregues podem mudar de acordo com a configuração atualizada do fluxo do Firehose.

O serviço Amazon Data Firehose usa, dentro do máximo possível, o mesmo ID de solicitação em todas as novas tentativas. Esse último atributo pode ser usado para eliminar duplicação pelo servidor de endpoint de HTTP. Se a solicitação ainda não for entregue após o tempo máximo permitido (com base na configuração do fluxo do Firehose), o lote de registros poderá, opcionalmente, ser entregue a um bucket de erros de acordo com a configuração do fluxo.

Exemplos

Exemplo de uma solicitação CWLog originada.

```
{
  "requestId": "ed4acda5-034f-9f42-bba1-f29aea6d7d8f",
  "timestamp": 1578090901599,
  "records": [
   {
    "data": {
      "messageType": "DATA_MESSAGE",
      "owner": "123456789012",
      "logGroup": "log_group_name",
      "logStream": "log_stream_name",
      "subscriptionFilters": [
        "subscription_filter_name"
      ],
      "logEvents": [
        {
          "id": "01234567890123456789012345678901234567890123456789012345",
          "timestamp": 1510109208016,
          "message": "log message 1"
        },
        {
          "id": "01234567890123456789012345678901234567890123456789012345",
          "timestamp": 1510109208017,
          "message": "log message 2"
        }
      ]
    }
   }
  ]
}
```

Como lidar com falhas de entrega de dados

Cada destino do Amazon Data Firehose tem seu próprio tratamento de falhas de entrega de dados.

Ao configurar um stream do Firehose, para muitos destinos, como OpenSearch Splunk e endpoints HTTP, você também configura um bucket S3 em que os dados que não foram entregues podem ser copiados. Para obter mais informações sobre como o Firehose faz backup dos dados em caso de falhas nas entregas, consulte as seções de destino relevantes nesta página. Para obter mais informações sobre como conceder acesso aos buckets do S3 nos quais os dados que não foram entregues podem ser copiados, consulte Concessão ao Firehose de acesso a um destino do Amazon S3. Quando o Firehose (a) falha em entregar dados ao destino do fluxo e (b) falha em gravar dados no bucket do S3 de backup devido a entregas com falha, ele pausa de fato a entrega do fluxo até que os dados possam ser entregues ao destino ou gravados no local do S3 para backup.

Amazon S3

A entrega de dados para o bucket do S3 pode apresentar falha por vários motivos. Por exemplo, o bucket pode não existir mais, o perfil do IAM que o Amazon Data Firehose assume pode não ter acesso ao bucket, a rede pode ter falhado ou outros eventos similares. Nessas condições, o Amazon Data Firehose continua a fazer novas tentativas por até 24 horas até que a entrega tenha êxito. O tempo máximo de armazenamento de dados do Amazon Data Firehose é de 24 horas. Se a entrega de dados apresentar falha por mais de 24 horas, os dados serão perdidos.

A entrega de dados para seu bucket do S3 pode falhar por vários motivos, como:

- O bucket não existe mais.
- A função do IAM assumida pelo Amazon Data Firehose não tem acesso ao bucket.
- Problemas de rede.
- Erros do S3, como HTTP 500s ou outras falhas de API.

Nesses casos, o Amazon Data Firehose tentará novamente a entrega:

- DirectPut fontes: as novas tentativas continuam por até 24 horas.
- Fontes do Kinesis Data Streams ou do Amazon MSK: as novas tentativas continuam indefinidamente, até a política de retenção definida no stream.

O Amazon Data Firehose entrega registros com falha para um bucket de erros do S3 somente quando o processamento do Lambda ou a conversão do parquet falham. Outros cenários de falha resultarão em tentativas contínuas de repetição do S3 até que o período de retenção seja atingido. Quando o Firehose entrega registros com sucesso ao S3, ele cria um arquivo de objeto do S3 e, em casos de falhas parciais no registro, ele tenta automaticamente a entrega e atualiza o mesmo arquivo de objeto do S3 com os registros processados com sucesso.

Amazon Redshift

Para um destino do Amazon Redshift, é possível especificar um período de novas tentativas (0 a 7.200 segundos) ao criar um fluxo do Firehose.

A entrega de dados ao cluster provisionado do Amazon Redshift ou ao grupo de trabalho do Amazon Redshift Sem Servidor pode falhar por vários motivos. Por exemplo, é possível ter uma configuração de cluster incorreta do fluxo do Firehose, um cluster ou grupo de trabalho em manutenção ou uma falha de rede. Nessas condições, o Amazon Data Firehose faz novas tentativas durante o período especificado e depois pula esse lote de objetos do Amazon S3. As informações dos objetos ignorados são entregues no bucket do S3 como um arquivo manifesto na pasta `errors/`, que pode ser usado para alocação manual. Para obter informações sobre como COPIAR manualmente os dados com arquivos de manifesto, consulte Uso de um manifesto para especificar arquivos de dados.

Amazon OpenSearch Service e OpenSearch Serverless

Para o destino OpenSearch Service e OpenSearch Serverless, você pode especificar uma duração de nova tentativa (0 a 7200 segundos) durante a criação do stream do Firehose.

A entrega de dados para seu cluster OpenSearch de serviços ou coleção OpenSearch sem servidor pode falhar por vários motivos. Por exemplo, você pode ter uma configuração incorreta de cluster de OpenSearch serviços ou coleção OpenSearch sem servidor do seu stream do Firehose, um cluster de OpenSearch serviços ou coleção OpenSearch sem servidor em manutenção, uma falha na rede ou eventos semelhantes. Nessas condições, o Amazon Data Firehose realiza novas tentativas durante período especificado e depois pula essa solicitação de índice. Os documentos ignorados são entregues no bucket do S3 na pasta `AmazonOpenSearchService_failed/`, que pode ser usada para alocação manual.

Para OpenSearch Serviço, cada documento tem o seguinte formato JSON:

```
{
```

```
    "attemptsMade": "(number of index requests attempted)",
    "arrivalTimestamp": "(the time when the document was received by Firehose)",
    "errorCode": "(http error code returned by OpenSearch Service)",
    "errorMessage": "(error message returned by OpenSearch Service)",
    "attemptEndingTimestamp": "(the time when Firehose stopped attempting index
 request)",
    "esDocumentId": "(intended OpenSearch Service document ID)",
    "esIndexName": "(intended OpenSearch Service index name)",
    "esTypeName": "(intended OpenSearch Service type name)",
    "rawData": "(base64-encoded document data)"
}
```

Para OpenSearch Serverless, cada documento tem o seguinte formato JSON:

```
{
    "attemptsMade": "(number of index requests attempted)",
    "arrivalTimestamp": "(the time when the document was received by Firehose)",
    "errorCode": "(http error code returned by OpenSearch Serverless)",
    "errorMessage": "(error message returned by OpenSearch Serverless)",
    "attemptEndingTimestamp": "(the time when Firehose stopped attempting index
 request)",
    "osDocumentId": "(intended OpenSearch Serverless document ID)",
    "osIndexName": "(intended OpenSearch Serverless index name)",
    "rawData": "(base64-encoded document data)"
}
```

Splunk

Quando o Amazon Data Firehose envia dados para o Splunk, ele aguarda uma confirmação do Splunk. Se ocorrer um erro ou a confirmação não chegar dentro do tempo limite para confirmação, o Amazon Data Firehose iniciará o contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Amazon Data Firehose considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Sempre que o Amazon Data Firehose envia dados para o Splunk, seja a tentativa inicial ou uma nova tentativa, ele reinicia o contador de tempo limite para confirmação. Em seguida, ele aguarda pela chegada de um reconhecimento do Splunk. Mesmo que o período de novas tentativas expire, o Amazon Data Firehose ainda aguardará a confirmação até recebê-la ou até que o tempo limite

para confirmação seja atingido. Se o tempo limite para confirmação expirar, o Amazon Data Firehose verificará se ainda resta algum tempo no contador de novas tentativas. Se houver tempo restante, ele tentará executar novamente e repetirá a lógica até receber um reconhecimento ou determinará que o tempo de tentar novamente expirou.

Uma falha em receber uma confirmação não é o único tipo de erro de entrega de dados que pode ocorrer. Para obter informações sobre outros tipos de erros de entrega de dados, consulte Erros de entrega de dados do Splunk. Qualquer erro de entrega de dados dispara a lógica de novas tentativas se a duração é maior que 0.

Veja a seguir um exemplo de registro de erro.


```
{
  "attemptsMade": 0,
  "arrivalTimestamp": 1506035354675,
  "errorCode": "Splunk.AckTimeout",
  "errorMessage": "Did not receive an acknowledgement from HEC before the HEC
 acknowledgement timeout expired. Despite the acknowledgement timeout, it's possible
 the data was indexed successfully in Splunk. Amazon Data Firehose backs up in Amazon
 S3 data for which the acknowledgement timeout expired.",
  "attemptEndingTimestamp": 13626284715507,
  "rawData":
 "MiAyNTE2MjAyNzIyMDkgZW5pLTA1ZjMyMmQ1IDIxOC45Mi4xODguMjE0IDE3Mi4xNi4xLjE2NyAyNTIzMyAxNDMzIDYgM
  "EventId": "49577193928114147339600778471082492393164139877200035842.0"
}
```


Destino do endpoint de HTTP

Quando o Amazon Data Firehose envia dados para um destino de endpoint de HTTP, ele espera por uma resposta desse destino. Se ocorrer um erro ou se a resposta não chegar dentro do tempo limite de resposta, o Amazon Data Firehose iniciará o contador do período de novas tentativas. Ele continuará tentando novamente até a duração da nova tentativa expirar. Depois disso, o Amazon Data Firehose considerará que houve uma falha de entrega de dados e fará o backup dos dados no bucket do Amazon S3.

Toda vez que o Amazon Data Firehose envia dados para um destino de endpoint de HTTP, seja a tentativa inicial ou uma nova tentativa, ele reinicia o contador de tempo limite para resposta. Depois, ele espera a chegada de uma resposta do destino de endpoint de HTTP. Mesmo que o período de novas tentativas expire, o Amazon Data Firehose ainda aguardará a resposta até recebê-la ou até que o tempo limite para confirmação seja atingido. Se o tempo limite para resposta expirar,

o Amazon Data Firehose verificará se ainda resta algum tempo no contador de novas tentativas. Se restar algum tempo, ele tentará novamente e repetirá a lógica até receber uma resposta ou determinar que o período de novas tentativas expirou.

Deixar de receber confirmação não é o único tipo de erro de entrega de dados que pode ocorrer. Para obter informações sobre outros tipos de erros de entrega de dados, consulte HTTP Endpoint Data Delivery Errors

Veja a seguir um exemplo de registro de erro.

```
{
 "attemptsMade":5,
 "arrivalTimestamp":1594265943615,
 "errorCode":"HttpEndpoint.DestinationException",
 "errorMessage":"Received the following response from the endpoint destination.
 {"requestId": "109777ac-8f9b-4082-8e8d-b4f12b5fc17b", "timestamp": 1594266081268,
 "errorMessage": "Unauthorized"}",
 "attemptEndingTimestamp":1594266081318,
 "rawData":"c2FtcGxlIHJhdyBkYXRh",
 "subsequenceNumber":0,
 "dataId":"49607357361271740811418664280693044274821622880012337186.0"
}
```

Snowflake

Para o destino Snowflake, ao criar um fluxo do Firehose, é possível especificar um período opcional de nova tentativa (0 a 7200 segundos). O valor padrão para a duração da nova tentativa é de 60 segundos.

A entrega de dados para sua tabela do Snowflake pode falhar por vários motivos, como configuração incorreta de destino do Snowflake, interrupção do Snowflake, falha na rede etc. A política de novas tentativas não se aplica a erros não recuperáveis. Por exemplo, se o Snowflake rejeitar sua carga útil de JSON porque ela tinha uma coluna extra que está faltando na tabela, o Firehose não tentará entregá-la novamente. Em vez disso, ele criará um backup para todas as falhas de inserção devido a problemas de carga útil de JSON em seu bucket de erros do S3.

Da mesma forma, se a entrega falhar devido a um perfil, tabela ou banco de dados incorretos, o Firehose não tentará novamente gravar os dados em seu bucket do S3. A duração da nova tentativa só se aplica a falhas devido a um problema no serviço Snowflake, falhas transitórias na rede, etc. Nessas condições, o Firehose faz novas tentativas durante o período especificado antes de entregá-

los ao S3. Os registros com falha são entregues na pasta snowflake-failed/, que pode ser usada para preenchimento manual.

Veja a seguir um exemplo de JSON para cada registro que você entrega ao S3.

```
{
    "attemptsMade": 3,
    "arrivalTimestamp": 1594265943615,
    "errorCode": "Snowflake.InvalidColumns",
    "errorMessage": "Snowpipe Streaming does not support columns of type AUTOINCREMENT,
 IDENTITY, GEO, or columns with a default value or collation",
    "attemptEndingTimestamp": 1712937865543,
    "rawData": "c2FtcGxlIHJhdyBkYXRh"
}
```

Configuração de formato de nome de objeto do Amazon S3

Quando o Firehose entrega dados para o Amazon S3, o nome da chave do objeto S3 segue o formato <prefixo avaliado><sufixo>, onde o sufixo tem o formato <nome do fluxo do Firehose><versão do fluxo do Firehose><anor><mês><dia><hora><minuto><segundo><uuid><extensão do arquivo> <versão do fluxo do Firehose>, começando com 1 e aumentando em 1 para cada alteração de configuração do fluxo do Firehose. É possível alterar as configurações do fluxo do Firehose (por exemplo, o nome do bucket do S3, as sugestões de armazenamento em buffer, a compactação e a criptografia). Você pode fazer isso usando o console Firehose ou a operação da UpdateDestinationAPI.

Para <prefixo avaliado>, o Firehose adiciona um prefixo de hora padrão no formato `YYYY/MM/dd/HH`. Esse prefixo cria uma hierarquia lógica no bucket, no qual cada barra (/) cria um nível na hierarquia. É possível modificar essa estrutura especificando um prefixo personalizado que inclua expressões que sejam avaliadas no runtime. Para obter informações sobre como especificar um prefixo personalizado, consulte Prefixos personalizados para objetos do Amazon Simple Storage Service.

Por padrão, o fuso horário usado para prefixo e sufixo de hora está em UTC, mas é possível alterá-lo para um fuso horário de sua preferência. Por exemplo, para usar o horário padrão do Japão em vez do UTC, você pode configurar o fuso horário para Ásia/Tóquio na configuração de parâmetros AWS Management Console da API (). CustomTimeZone A lista a seguir contém fusos horários com suporte no Firehose para a configuração do prefixo do S3.

Fusos horários suportados:

A seguir há uma lista de fusos horários com suporte no Firehose para a configuração do prefixo do S3.

Africa

```
Africa/Abidjan
Africa/Accra
Africa/Addis_Ababa
Africa/Algiers
Africa/Asmera
Africa/Bangui
Africa/Banjul
Africa/Bissau
Africa/Blantyre
Africa/Bujumbura
Africa/Cairo
Africa/Casablanca
Africa/Conakry
Africa/Dakar
Africa/Dar_es_Salaam
Africa/Djibouti
Africa/Douala
Africa/Freetown
Africa/Gaborone
Africa/Harare
Africa/Johannesburg
Africa/Kampala
Africa/Khartoum
Africa/Kigali
Africa/Kinshasa
Africa/Lagos
Africa/Libreville
Africa/Lome
Africa/Luanda
Africa/Lubumbashi
Africa/Lusaka
Africa/Malabo
Africa/Maputo
Africa/Maseru
Africa/Mbabane
Africa/Mogadishu
```

```
Africa/Monrovia
Africa/Nairobi
Africa/Ndjamena
Africa/Niamey
Africa/Nouakchott
Africa/Ouagadougou
Africa/Porto-Novo
Africa/Sao_Tome
Africa/Timbuktu
Africa/Tripoli
Africa/Tunis
Africa/Windhoek
```

America

```
America/Adak
America/Anchorage
America/Anguilla
America/Antigua
America/Aruba
America/Asuncion
America/Barbados
America/Belize
America/Bogota
America/Buenos_Aires
America/Caracas
America/Cayenne
America/Cayman
America/Chicago
America/Costa_Rica
America/Cuiaba
America/Curacao
America/Dawson_Creek
America/Denver
America/Dominica
America/Edmonton
America/El_Salvador
America/Fortaleza
America/Godthab
America/Grand_Turk
America/Grenada
America/Guadeloupe
America/Guatemala
```

```
America/Guayaquil
America/Guyana
America/Halifax
America/Havana
America/Indianapolis
America/Jamaica
America/La_Paz
America/Lima
America/Los_Angeles
America/Managua
America/Manaus
America/Martinique
America/Mazatlan
America/Mexico_City
America/Miquelon
America/Montevideo
America/Montreal
America/Montserrat
America/Nassau
America/New_York
America/Noronha
America/Panama
America/Paramaribo
America/Phoenix
America/Port_of_Spain
America/Port-au-Prince
America/Porto_Acre
America/Puerto_Rico
America/Regina
America/Rio_Branco
America/Santiago
America/Santo_Domingo
America/Sao_Paulo
America/Scoresbysund
America/St_Johns
America/St_Kitts
America/St_Lucia
America/St_Thomas
America/St_Vincent
America/Tegucigalpa
America/Thule
America/Tijuana
America/Tortola
America/Vancouver
```

```
America/Winnipeg
```

Antarctica

```
Antarctica/Casey
Antarctica/DumontDUrville
Antarctica/Mawson
Antarctica/McMurdo
Antarctica/Palmer
```

Asia

```
Asia/Aden
Asia/Almaty
Asia/Amman
Asia/Anadyr
Asia/Aqtau
Asia/Aqtobe
Asia/Ashgabat
Asia/Ashkhabad
Asia/Baghdad
Asia/Bahrain
Asia/Baku
Asia/Bangkok
Asia/Beirut
Asia/Bishkek
Asia/Brunei
Asia/Calcutta
Asia/Colombo
Asia/Dacca
Asia/Damascus
Asia/Dhaka
Asia/Dubai
Asia/Dushanbe
Asia/Hong_Kong
Asia/Irkutsk
Asia/Jakarta
Asia/Jayapura
Asia/Jerusalem
Asia/Kabul
Asia/Kamchatka
Asia/Karachi
Asia/Katmandu
```

```
Asia/Krasnoyarsk
Asia/Kuala_Lumpur
Asia/Kuwait
Asia/Macao
Asia/Magadan
Asia/Manila
Asia/Muscat
Asia/Nicosia
Asia/Novosibirsk
Asia/Phnom_Penh
Asia/Pyongyang
Asia/Qatar
Asia/Rangoon
Asia/Riyadh
Asia/Saigon
Asia/Seoul
Asia/Shanghai
Asia/Singapore
Asia/Taipei
Asia/Tashkent
Asia/Tbilisi
Asia/Tehran
Asia/Thimbu
Asia/Thimphu
Asia/Tokyo
Asia/Ujung_Pandang
Asia/Ulaanbaatar
Asia/Ulan_Bator
Asia/Vientiane
Asia/Vladivostok
Asia/Yakutsk
Asia/Yekaterinburg
Asia/Yerevan
```

Atlantic

```
Atlantic/Azores
Atlantic/Bermuda
Atlantic/Canary
Atlantic/Cape_Verde
Atlantic/Faeroe
Atlantic/Jan_Mayen
Atlantic/Reykjavik
```

```
Atlantic/South_Georgia
Atlantic/St_Helena
Atlantic/Stanley
```

Australia

```
Australia/Adelaide
Australia/Brisbane
Australia/Broken_Hill
Australia/Darwin
Australia/Hobart
Australia/Lord_Howe
Australia/Perth
Australia/Sydney
```

Europe

```
Europe/Amsterdam
Europe/Andorra
Europe/Athens
Europe/Belgrade
Europe/Berlin
Europe/Brussels
Europe/Bucharest
Europe/Budapest
Europe/Chisinau
Europe/Copenhagen
Europe/Dublin
Europe/Gibraltar
Europe/Helsinki
Europe/Istanbul
Europe/Kaliningrad
Europe/Kiev
Europe/Lisbon
Europe/London
Europe/Luxembourg
Europe/Madrid
Europe/Malta
Europe/Minsk
Europe/Monaco
Europe/Moscow
Europe/Oslo
Europe/Paris
```

```
Europe/Prague
Europe/Riga
Europe/Rome
Europe/Samara
Europe/Simferopol
Europe/Sofia
Europe/Stockholm
Europe/Tallinn
Europe/Tirane
Europe/Vaduz
Europe/Vienna
Europe/Vilnius
Europe/Warsaw
Europe/Zurich
```

Indian

```
Indian/Antananarivo
Indian/Chagos
Indian/Christmas
Indian/Cocos
Indian/Comoro
Indian/Kerguelen
Indian/Mahe
Indian/Maldives
Indian/Mauritius
Indian/Mayotte
Indian/Reunion
```

Pacific

```
Pacific/Apia
Pacific/Auckland
Pacific/Chatham
Pacific/Easter
Pacific/Efate
Pacific/Enderbury
Pacific/Fakaofo
Pacific/Fiji
Pacific/Funafuti
Pacific/Galapagos
Pacific/Gambier
Pacific/Guadalcanal
```

```
Pacific/Guam
Pacific/Honolulu
Pacific/Kiritimati
Pacific/Kosrae
Pacific/Majuro
Pacific/Marquesas
Pacific/Nauru
Pacific/Niue
Pacific/Norfolk
Pacific/Noumea
Pacific/Pago_Pago
Pacific/Palau
Pacific/Pitcairn
Pacific/Ponape
Pacific/Port_Moresby
Pacific/Rarotonga
Pacific/Saipan
Pacific/Tahiti
Pacific/Tarawa
Pacific/Tongatapu
Pacific/Truk
Pacific/Wake
Pacific/Wallis
```

Você não pode alterar o campo de sufixo, exceto <extensão do arquivo>. Quando você ativa a conversão ou a compactação do formato de dados, o Firehose anexa uma extensão de arquivo com base na configuração. A tabela a seguir explica a extensão de arquivo padrão anexada pelo Firehose:

Configuração	Extensão de arquivo
Conversão de formato de dados: Parquet	.parquet
Conversão de formato de dados: ORC	.orc
Compactação: Gzip	.gz
Compactação: Zip	.zip

Configuração	Extensão de arquivo
Compactação: Snappy	.snappy
Compactação: Hadoop-Snappy	.hsnappy

Você também pode especificar uma extensão de arquivo de sua preferência no console ou na API do Firehose. A extensão do arquivo deve começar com um ponto (.) e pode conter os caracteres permitidos: 0-9a-z!-_.*' (). A extensão do arquivo não pode exceder 128 caracteres.

> **Note**
>
> Quando você especifica uma extensão de arquivo, ela substitui a extensão de arquivo padrão que o Firehose adiciona quando a conversão de formato de dados ou a compactação estão habilitadas.

Noções básicas de prefixos personalizados para objetos do Amazon S3

Os objetos entregues ao Amazon S3 seguem o formato do nome <prefixo avaliado><sufixo>. É possível especificar seu prefixo personalizado que inclui expressões que são avaliadas no runtime. O prefixo personalizado que você especificar substituirá o prefixo padrão `yyyy/MM/dd/HH`.

É possível usar expressões nos seguintes formatos no seu prefixo personalizado: `!{namespace:value}`, em que `namespace` pode ser um dos que se seguem, como explicado nas próximas seções.

- `firehose`
- `timestamp`
- `partitionKeyFromQuery`
- `partitionKeyFromLambda`

Se um prefixo terminar com uma barra, ele aparecerá como uma pasta no bucket do Amazon S3. Para obter mais informações, consulte Formato de nome de objeto do Amazon S3 no Amazon Data FirehoseDeveloper Guide.

Namespace do **`timestamp`**

Os valores válidos para esse namespace são cadeias de caracteres Java válidas. DateTimeFormatter Como um exemplo, no ano 2018, a expressão `!{timestamp:yyyy}` é avaliada para `2018`.

Ao avaliar carimbos de data/hora, o Firehose usa o carimbo de data/hora aproximado da chegada do registro mais antigo contido no objeto do Amazon S3 que está sendo gravado.

Por padrão, o carimbo de data/hora está em UTC. Mas é possível especificar o fuso horário de sua preferência. Por exemplo, você pode configurar o fuso horário para Ásia/Tóquio na configuração de parâmetros da API () AWS Management Console ou na configuração de parâmetros da API (CustomTimeZone) se quiser usar o horário padrão do Japão em vez do UTC. Para ver a lista de fusos horários com suporte, consulte Formato de nome de objeto do Amazon S3.

Se você usar o namespace `timestamp` mais de uma vez na mesma expressão do prefixo, cada instância será avaliada no mesmo momento.

Namespace do **`firehose`**

Há dois valores que podem ser usados com esse namespace: `error-output-type` e `random-string`. A tabela a seguir explica como usá-los.

Os valores do namespace **`firehose`**.

Conversão	Descrição	Exemplo de entrada	Exemplo de saída	Observações
`error-output-type`	É avaliado como uma das seguintes sequências de caracteres, dependendo da configuração do stream do Firehose e do motivo da falha: {processing-failed,	`myPrefix/result=!{firehose:error-output-type}/!{timestamp:yyyy/MM/dd}`	`myPrefix/result=processing-failed/2018/08/03`	O error-output-type valor só pode ser usado no ErrorOutputPrefix campo.

Conversão	Descrição	Exemplo de entrada	Exemplo de saída	Observações
	AmazonOpenSearchService-failed, splunk-failed,,}. format-conversion-failed http-endpoint-failed Se você usá-lo mais de uma vez na mesma expressão, cada instância será avaliada para a mesma string de erro.			
`random-string`	Avalia para uma string aleatória de 11 caracteres. Se você usá-lo mais de uma vez na mesma expressão, cada instância será avaliada para uma nova string aleatória.	`myPrefix/!{firehose:random-string}/`	`myPrefix/046b6c7f-0b/`	É possível usá-lo com os dois tipos de prefixo. Pode ser colocado no início da string de formato para obter um prefixo aleatório, o que, às vezes, é necessário para atingir uma throughput extremamente alto com o Amazon S3.

Namespaces `partitionKeyFromLambda` e `partitionKeyFromQuery`

Para o particionamento dinâmico, você deve usar o seguinte formato de expressão no prefixo de bucket do S3: `!{namespace:value}`, em que o namespace pode ser `partitionKeyFromQuery`, `partitionKeyFromLambda` ou ambos. Se estiver usando análise em linha para criar as chaves de particionamento para os dados da fonte, você deverá especificar um valor de prefixo de bucket do S3 consistindo em expressões especificadas no seguinte formato: `"partitionKeyFromQuery:keyID"`. Se estiver usando função do AWS Lambda para criar as chaves de particionamento para os dados da fonte, você deverá especificar um valor de prefixo de bucket de S3 que consista em expressões especificadas no seguinte formato: `"partitionKeyFromLambda:keyID"`. Para obter mais informações, consulte "Escolha o Amazon S3 como destino" em Criação de um fluxo do Firehose.

Regras semânticas

As regras a seguir se aplicam às expressões `Prefix` e `ErrorOutputPrefix`.

- Para o namespace `timestamp`, qualquer caractere que não estiver em aspas simples é avaliado. Em outras palavras, qualquer string recuada com aspas simples no campo do valor é considerada literalmente.
- Se você especificar um prefixo que não contenha uma expressão de namespace de carimbo de data/hora, o Firehose acrescentará a expressão `!{timestamp:yyyy/MM/dd/HH/}` ao valor no campo `Prefix`.
- A sequência `!{` pode aparecer somente em expressões `!{namespace:value}`.
- `ErrorOutputPrefix` poderá ser nulo somente se `Prefix` não tiver expressões; Neste caso, `Prefix` é avaliado como `<specified-prefix>yyyy/MM/DDD/HH/`, e `ErrorOutputPrefix` é avaliado como `<specified-prefix><error-output-type>yyyy/MM/DDD/HH/`. `DDD` representa o dia do ano.
- Se você especificar uma expressão para `ErrorOutputPrefix`, deverá incluir pelo menos uma instância de `!{firehose:error-output-type}`.
- `Prefix` não pode conter `!{firehose:error-output-type}`.
- `Prefix` e `ErrorOutputPrefix` não podem ter mais de 512 caracteres após serem avaliados.
- Se o destino for o Amazon Redshift, o `Prefix` não deverá conter expressões e o `ErrorOutputPrefix` deverá ser nulo.
- Quando o destino é Amazon OpenSearch Service ou Splunk e não `ErrorOutputPrefix` é especificado, o Firehose usa `Prefix` o campo para registros com falha.

- Quando o destino é o Amazon S3, o `Prefix` e o `ErrorOutputPrefix` na configuração de destino do Amazon S3 são usados para registros com êxito e registros com falha, respectivamente. Se você usar a AWS CLI ou a API, poderá usar a `ExtendedS3DestinationConfiguration` para especificar uma configuração de backup do Amazon S3 com seu próprio `Prefix` e `ErrorOutputPrefix`.
- Quando você usa AWS Management Console e define o destino como Amazon S3, o Firehose usa `Prefix` e `ErrorOutputPrefix` na configuração de destino para registros bem-sucedidos e registros com falha, respectivamente. Se você especificar um prefixo usando expressões, deverá especificar o prefixo do erro, incluindo `!{firehose:error-output-type}`.
- Quando você usa `ExtendedS3DestinationConfiguration` com o AWS CLI, a API ou AWS CloudFormation, se você especificar um`S3BackupConfiguration`, o Firehose não fornece um padrão. `ErrorOutputPrefix`
- Você não pode usar `partitionKeyFromLambda` `partitionKeyFromQuery` namespaces ao criar ErrorOutputPrefix expressões.

Prefixos de exemplo

Exemplos de **`Prefix`** e **`ErrorOutputPrefix`**

Entrada	Prefixo avaliado (às 10:30 AM UTC em 27 de agosto de 2018)

`Prefix`: não especificado
 	
`ErrorOutputPrefix` : `myFirehoseFailures/!{firehose:error-output-type}/`
	`Prefix`: `2018/08/27/10`
	
	`ErrorOutputPrefix` : `myFirehoseFailures/processing-failed/`

`Prefix`: `!{timestamp:yyyy/MM/dd}` `ErrorOutputPrefix` : não especificado	Entrada inválida: `ErrorOutputPrefix` não poderá ser nulo quando o prefixo tiver expressões

`Prefix`: `myFirehose/DeliveredYear=!{timestamp:yyyy}/anyMonth/rand=!{firehose:random-string}`
 	
`ErrorOutputPrefix` : `myFirehoseFailures/!{firehose:error-`
	`Prefix`: `myFirehose/DeliveredYear=2018/anyMonth/rand=5abf82daaa5`

Entrada	Prefixo avaliado (às 10:30 AM UTC em 27 de agosto de 2018)
`output-type}/!{timestamp:yyyy}/anyMonth/!{timestamp:dd}`	`ErrorOutputPrefix : myFirehoseFailures/processing-failed/2018/anyMonth/10`

`Prefix: myPrefix/year=!{timestamp:yyyy}/month=!{timestamp:MM}/day=!{timestamp:dd}/hour=!{timestamp:HH}/`
 	
`ErrorOutputPrefix : myErrorPrefix/year=!{timestamp:yyyy}/month=!{timestamp:MM}/day=!{timestamp:dd}/hour=!{timestamp:HH}/!{firehose:error-output-type}`
	`Prefix: myPrefix/year=2018/month=07/day=06/hour=23/`
	
	`ErrorOutputPrefix : myErrorPrefix/year=2018/month=07/day=06/hour=23/processing-failed`

`Prefix: myFirehosePrefix/`
 	
`ErrorOutputPrefix` : não especificado
	`Prefix: myFirehosePrefix/2018/08/27/`
	
	`ErrorOutputPrefix : myFirehosePrefix/processing-failed/2018/08/27/`

Configurar a rotação do índice para o OpenSearch serviço

Para o destino do OpenSearch serviço, você pode especificar uma opção de rotação de índice com base no tempo a partir de uma das cinco opções a seguir: NoRotationOneHour,OneDay,OneWeek, ouOneMonth.

Dependendo da opção de rotação escolhida, o Amazon Data Firehose acrescenta uma parte do carimbo de data/hora em UTC ao nome do índice especificado. Ele alterna o time stamp anexado adequadamente. O exemplo a seguir mostra o nome do índice resultante em OpenSearch Serviço para cada opção de rotação do índice, onde está o nome do índice especificado myindex e a data e hora de chegada. `2016-02-25T13:00:00Z`

RotationPeriod	IndexName
`NoRotation`	`myindex`
`OneHour`	`myindex-2016-02-25-13`
`OneDay`	`myindex-2016-02-25`
`OneWeek`	`myindex-2016-w08`
`OneMonth`	`myindex-2016-02`

> **Note**
>
> Com a opção `OneWeek`, o Data Firehose cria índices automaticamente usando o formato <ANO>-w <NÚMERO DASEMANA>(por exemplo,`2020-w33`), em que o número da semana é calculado usando o horário UTC e de acordo com as seguintes convenções dos EUA:
>
> - A semana começa no domingo
> - A primeira semana do ano é a primeira semana que contém um sábado naquele ano

Pausa e retomada da entrega de dados

Depois que você configura um fluxo do Firehose, os dados disponíveis na fonte do fluxo são continuamente entregues ao destino. Se você se deparar com situações em que o destino do fluxo esteja temporariamente indisponível (por exemplo, durante operações de manutenção planejadas), pode ser que queira pausar temporariamente a entrega de dados e continuar quando o destino estiver disponível novamente.

> **Important**
>
> Ao usar a abordagem descrita abaixo para pausar e retomar um fluxo, depois de retomar o fluxo você verá que poucos registros são entregues ao bucket de erros no Amazon S3, enquanto o restante do fluxo continua sendo entregue ao destino. Essa é uma limitação conhecida da abordagem, e ocorre porque um pequeno número de registros que não

puderam ser entregues anteriormente ao destino após várias tentativas são rastreados como tendo falhado.

Pausa de um fluxo do Firehose

Para pausar a entrega de fluxo no Firehose, primeiro remova as permissões para o Firehose gravar no local de backup do S3 em caso de falhas nas entregas. Por exemplo, se quiser pausar o stream do Firehose com OpenSearch um destino, você pode fazer isso atualizando as permissões. Para obter mais informações, consulte Conceder acesso ao Firehose a um destino de OpenSearch serviço público.

Remova a permissão `"Effect": "Allow"` para a ação `s3:PutObject` e adicione explicitamente uma instrução que aplique a permissão `Effect": "Deny"` à ação `s3:PutObject` para o bucket do S3 usado para fazer backup de entregas com falha. Em seguida, desative o destino do stream (por exemplo, desative o OpenSearch domínio de destino) ou remova as permissões para que o Firehose grave no destino. Para atualizar as permissões para outros destinos, consulte a seção relativa ao destino em Controle de acesso com o Amazon Data Firehose. Depois de concluir essas duas ações, o Firehose deixará de fornecer streams e você poderá monitorar isso usando CloudWatch métricas do Firehose.

Important

Quando você pausa a entrega do fluxo no Firehose, precisa garantir que a fonte do fluxo (por exemplo, o Kinesis Data Streams ou o Managed Service for Kafka) esteja configurada para reter os dados até que a entrega do fluxo seja retomada e os dados sejam entregues ao destino. Se a fonte for DirectPUT, o Firehose reterá os dados por 24 horas. Poderá ocorrer uma perda de dados se você não retomar o fluxo de entregar os dados antes da expiração do período de retenção de dados.

Retomada do fluxo do Firehose

Para retomar a entrega, primeiro reverta a alteração feita anteriormente no destino do fluxo, ativando o destino e garantindo que o Firehose tenha permissões para entregar o fluxo ao destino. Depois, reverta as alterações feitas anteriormente nas permissões aplicadas ao bucket do S3 de backup de entregas com falha. Remova a permissão `"Effect": "Allow"` para a ação `s3:PutObject` e remova a permissão `"Effect": "Deny"` para a ação `s3:PutObject` para o bucket do S3 usado

para backup das entregas com falha. Por fim, monitore usando CloudWatch métricas do Firehose para confirmar se o stream está sendo entregue ao destino. Para visualizar e solucionar erros, use o monitoramento Amazon CloudWatch Logs para Firehose.

Entrega de dados às tabelas do Apache Iceberg com o Amazon Data Firehose

O Apache Iceberg é um formato de tabela de código aberto de alta performance para realizar análises de big data. O Apache Iceberg traz a confiabilidade e a simplicidade das tabelas SQL para os data lakes do Amazon S3 e possibilita que mecanismos de análise de código aberto como Spark, Flink, Trino, Hive e Impala trabalhem com os mesmos dados simultaneamente. Para obter mais informações sobre o Apache Iceberg, consulte https://iceberg.apache.org/.

É possível usar o Firehose para entregar dados de streaming às tabelas do Apache Iceberg no Amazon S3. Suas tabelas Apache Iceberg podem ser autogerenciadas no Amazon S3 ou hospedadas nas tabelas do Amazon S3. Nas tabelas autogerenciadas do Iceberg, você gerencia todas as otimizações da tabela, como compactação e expiração de instantâneos. A funcionalidade Tabelas do Amazon S3 fornece armazenamento otimizado para workloads de analytics em grande escala, com recursos que melhoram continuamente a performance das consultas e reduzem os custos de armazenamento de dados tabulares. Para obter mais informações sobre as tabelas do Amazon S3, consulte Tabelas do Amazon S3.

Esse recurso permite rotear registros de um único fluxo para diferentes tabelas Apache Iceberg. Você pode aplicar automaticamente as operações de inserção, atualização e exclusão aos registros nessas tabelas. Ele também suporta controle de acesso a dados refinado em tabelas Apache Iceberg no Amazon S3 com. AWS Lake Formation Você pode especificar controles de acesso centralmente AWS Lake Formation e fornecer permissões mais granulares em nível de tabela e coluna para o Firehose.

Considerações e limitações

> **Note**
>
> O Firehose oferece suporte ao Apache Iceberg Tables como destino em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia).

O suporte do Firehose para tabelas do Apache Iceberg tem as considerações e limitações a seguir.

- Taxa de transferência — Se você usar o Direct PUT como fonte para entregar dados às tabelas do Apache Iceberg, a taxa de transferência máxima por fluxo será 5 MiB/second in US East

(N. Virginia), US West (Oregon), and Europe (Ireland) Regions and 1 MiB/second em todos os outros. Regiões da AWS Se você quiser inserir dados nas tabelas do Iceberg sem atualizações e exclusões e quiser maior throughput para seu fluxo, use o formulário de limites do Firehose para solicitar um aumento do limite de throughput.

Você também pode definir o `AppendOnly` sinalizador como `True` se quiser apenas inserir dados e não realizar atualizações e exclusões. Ao definir a `AppendOnly` bandeira como`True`, o Firehose é escalado automaticamente para corresponder à sua taxa de transferência. Atualmente, você pode definir esse sinalizador somente com a operação CreateDeliveryStreamda API.

Se um stream Direct PUT sofrer limitação devido a maiores volumes de ingestão de dados que excedem a capacidade de taxa de transferência de um stream Firehose, o Firehose aumenta automaticamente o limite de taxa de transferência do stream até que a limitação seja contida. Dependendo do aumento da taxa de transferência e da limitação, pode levar mais tempo para que o Firehose aumente a taxa de transferência de um stream até os níveis desejados. Por esse motivo, continue tentando novamente os registros de ingestão de dados com falha. Se você espera que o volume de dados aumente em grandes rajadas repentinas ou se seu novo stream precisar de uma taxa de transferência maior do que o limite de taxa de transferência padrão, solicite o aumento do limite de taxa de transferência.

- Transação S3 por segundo (TPS) — Para otimizar o desempenho do S3, se você estiver usando o Kinesis Data Streams ou o Amazon MSK como fonte, recomendamos particionar o registro de origem usando uma chave de partição adequada. Dessa forma, os registros de dados que são roteados para a mesma tabela Iceberg são mapeados para uma ou algumas partições de origem conhecidas como fragmentos. Se possível, distribua registros de dados pertencentes a diferentes tabelas Iceberg de destino em diferentespartitions/shards, so that you can use all the aggregate throughput available across all the partitions/shards of the source topic/stream.
- Colunas: para nomes e valores de colunas, o Firehose usa somente o primeiro nível de nós em um JSON aninhado de vários níveis. Por exemplo, o Firehose seleciona os nós que estão disponíveis no primeiro nível, incluindo o campo de posição. Os nomes das colunas e os tipos de dados dos dados da fonte devem corresponder aos das tabelas de destino para que o Firehose entregue com êxito. Nesse caso, o Firehose espera que você tenha uma coluna de tipo de dados struct ou map em suas tabelas do Iceberg que corresponda ao campo de posição. O Firehose oferece suporte a 16 níveis de aninhamento. Veja a seguir um exemplo de JSON aninhado.

```
{
   "version":"2016-04-01",
   "deviceId":"<solution_unique_device_id>",
```

```
   "sensorId":"<device_sensor_id>",
   "timestamp":"2024-01-11T20:42:45.000Z",
   "value":"<actual_value>",
   "position":{
      "x":143.595901,
      "y":476.399628,
      "z":0.24234876
   }
}
```

Se os nomes das colunas ou os tipos de dados não corresponderem, o Firehose gerará um erro e entregará os dados ao bucket de erros do S3. Se todos os nomes de colunas e tipos de dados corresponderem nas tabelas do Apache Iceberg, mas você tiver um campo adicional presente no registro da fonte, o Firehose ignorará o novo campo.

- Um objeto JSON por registro: é possível enviar somente um objeto JSON em um registro do Firehose. Se você agregar e enviar vários objetos JSON dentro de um registro, o Firehose gerará um erro e entregará os dados ao bucket de erros do S3. Se você agregar registros com KPL e ingerir dados no Firehose com o Amazon Kinesis Data Streams como fonte, o Firehose automaticamente os desagregará e usará um objeto JSON por registro.
- Otimização de compactação e armazenamento: toda vez que você grava nas tabelas do Iceberg usando o Firehose, ele confirma e gera snapshots, arquivos de dados e arquivos de exclusão. Ter muitos arquivos de dados aumenta a sobrecarga de metadados e afeta a performance de leitura. Para obter um desempenho de consulta eficiente, talvez você queira considerar uma solução que, periodicamente, pegue pequenos arquivos de dados e os reescreva em um número menor de arquivos de dados maiores. Esse processo é chamado de compactação. AWS Glue Data Catalog oferece suporte à compactação automática de suas tabelas do Apache Iceberg. Para obter mais informações, consulte Gerenciamento de compactação no Guia do usuário do AWS Glue. Para obter informações adicionais, consulte Automatic compaction of Apache Iceberg Tables. Também é possível executar o comando do Athena Optimize para realizar a compactação manualmente. Para obter mais informações sobre o comando Otimizar, consulte Athena Optimize.

  Além da compactação dos arquivos de dados, você também pode otimizar o consumo de armazenamento com a instrução VACUUM, que realiza a manutenção das tabelas do Apache Iceberg, como expiração do snapshot e remoção de arquivos órfãos. Como alternativa, é possível usar o, AWS Glue Data Catalog que também oferece suporte à otimização gerenciada de tabelas do Apache Iceberg removendo automaticamente os arquivos de dados, arquivos órfãos e snapshots expirados que não são mais necessários. Para obter mais informações, consulte esta postagem no blog sobre otimização de armazenamento de tabelas do Apache Iceberg.

- Não oferecemos suporte à fonte do Amazon MSK Sem Servidor como destino para tabelas do Apache Iceberg como destino.
- Para uma operação de atualização, o Firehose coloca um arquivo de exclusão seguido por uma operação de inserção. A exclusão de arquivos incorre em cobranças de entrada do Amazon S3.

Pré-requisitos para uso das tabelas do Apache Iceberg como destino

Escolha entre as seguintes opções para preencher os pré-requisitos obrigatórios.

Tópicos

Pré-requisitos para entrega em tabelas Iceberg no Amazon S3

Antes de começar, conclua os pré-requisitos a seguir.

- Crie um bucket do Amazon S3: você deve criar um bucket do Amazon S3 para adicionar o caminho do arquivo de metadados durante a criação das tabelas. Para obter mais informações, consulte Criação de um bucket do S3.
- Crie um perfil do IAM com as permissões necessárias: o Firehose precisa de um perfil do IAM com permissões específicas para acessar as tabelas do AWS Glue e gravar dados no Amazon S3. A mesma função é usada para conceder ao AWS Glue acesso aos buckets do Amazon S3. Você precisará desse perfil do IAM ao criar uma tabela do Iceberg e um fluxo do Firehose. Para obter mais informações, consulte Conceda ao Firehose acesso às tabelas do Amazon S3.
- Crie tabelas do Apache Iceberg: se você estiver configurando chaves exclusivas no fluxo do Firehose para atualizações e exclusões, o Firehose as validará se a tabela e as chaves exclusivas existirem como parte da criação do fluxo. Para esse cenário, você deve criar tabelas antes de criar o fluxo do Firehose. É possível usar o AWS Glue para criar tabelas do Apache Iceberg. Para obter mais informações, consulte Criar tabelas do Apache Iceberg. Se você não estiver configurando chaves exclusivas no fluxo do Firehose, não precisará criar tabelas do Iceberg antes de criar um fluxo do Firehose.

> **Note**
>
> O Firehose oferece suporte à versão e formato de tabela a seguir para tabelas do Apache Iceberg.
>
> - Versão de formato de tabela: o Firehose oferece suporte apenas ao formato de tabela V2. Não crie tabelas no formato V1, caso contrário, você receberá um erro e, em vez disso, os dados serão entregues ao bucket de erros do S3.
> - Formato de armazenamento de dados: o Firehose grava dados nas tabelas do Apache Iceberg no formato Parquet.
> - Operação em nível de linha: o Firehose oferece suporte ao modo Merge-on-Read (MOR) de gravação de dados nas tabelas do Apache Iceberg.

Pré-requisitos para entrega às tabelas do Amazon S3

Para entregar dados aos buckets de tabela do Amazon S3, preencha os pré-requisitos a seguir.

- Crie um bucket do S3 Table, namespace, tabelas no bucket da tabela e outras etapas de integração descritas em Introdução às tabelas do Amazon S3. Os nomes das colunas devem estar em minúsculas devido às limitações impostas pela integração do catálogo do S3 Tables, conforme especificado nas limitações da integração do catálogo de tabelas do S3.
- Crie um perfil do IAM com as permissões necessárias: o Firehose precisa de um perfil do IAM com permissões específicas para acessar AWS Glue as tabelas em um bucket de tabela do Amazon S3. Para gravar em tabelas em um bucket de tabelas do S3, você também deve fornecer à função do IAM as permissões necessárias em AWS Lake Formation. Você configura esse perfil do IAM ao criar um fluxo do Firehose. Para obter mais informações, consulte Conceder ao Firehose acesso às tabelas do Amazon S3.
- Configurar AWS Lake Formation permissões — AWS Lake Formation gerencia o acesso aos recursos da sua tabela. O Lake Formation usa um modelo de permissões próprio que permite um controle de acesso detalhado aos recursos do Catálogo de Dados.

Para step-by-step integração, consulte o blog Crie um data lake para streaming de dados com o Amazon S3 Tables e o Amazon Data Firehose. Para obter informações adicionais, consulte também Usando tabelas do Amazon S3 com serviços de AWS análise.

Configuração do fluxo do Firehose

Para criar um fluxo do Firehose com tabelas do Apache Iceberg como destino, você precisa configurar os itens a seguir.

> **Note**
>
> A configuração de um stream do Firehose para entrega em tabelas em buckets de mesa do S3 é a mesma das tabelas Apache Iceberg no Amazon S3.

Configuração de fonte e destino

Para entregar dados para as tabelas do Apache Iceberg, escolha a fonte do seu fluxo.

Para configurar a fonte do seu fluxo, consulte Definição de configurações da fonte.

Em seguida, escolha Tabelas do Apache Iceberg como destino e forneça um nome de fluxo do Firehose.

Configuração da transformação de dados

Para realizar transformações personalizadas em seus dados, como adicionar ou modificar registros em seu fluxo de entrada, é possível adicionar uma função do Lambda ao seu fluxo do Firehose. Para obter mais informações sobre transformação de dados usando o Lambda em um fluxo do Firehose, consulte Transformação de dados da fonte no Amazon Data Firehose.

Para tabelas do Apache Iceberg, você precisa especificar como deseja encaminhamento os registros de entrada para diferentes tabelas de destino e as operações que deseja realizar. Uma das maneiras de fornecer as informações de encaminhamento necessárias para o Firehose é usando uma função do Lambda.

Para obter mais informações, consulte Encaminhamento de registros para diferentes tabelas do Iceberg.

Conexão de catálogo de dados

O Apache Iceberg requer um catálogo de dados para gravar nas tabelas do Apache Iceberg. O Firehose se integra ao AWS Glue Data Catalog para tabelas do Apache Iceberg.

É possível usar o AWS Glue Data Catalog na mesma conta do fluxo do Firehose ou em uma conta cruzada e na mesma região do fluxo do Firehose (padrão), ou em uma região diferente.

Se você estiver entregando para uma tabela do Amazon S3 e estiver usando o console para configurar seu stream do Firehose, selecione o catálogo que corresponde ao seu catálogo de tabelas do Amazon S3. Se você estiver usando a CLI para configurar seu stream do Firehose, então na `CatalogConfiguration` entrada, use `CatalogARN` com o formato:. `arn:aws:glue:<region>:<account-id>:catalog/s3tablescatalog/<s3 table bucket name>` Para obter mais informações, consulte Configurar um fluxo do Firehose para tabelas do Amazon S3.

Configuração de expressões JQ

Para tabelas do Apache Iceberg, você precisa especificar como deseja encaminhamento os registros de entrada para diferentes tabelas de destino e as operações, como inserir, atualizar e excluir, que deseja realizar. Isso pode ser feito configurando expressões JQ para que o Firehose analise e obtenha as informações necessárias. Para obter mais informações, consulte ???.

Configuração de chaves exclusivas

Atualizações e exclusões com mais de uma tabela: as chaves exclusivas são um ou mais campos em seu registro de fonte que identificam exclusivamente uma linha nas tabelas do Apache Iceberg. Se você inseriu somente um cenário com mais de uma tabela, não precisará configurar chaves exclusivas. Se você quiser fazer atualizações e exclusões em determinadas tabelas, deverá configurar chaves exclusivas para essas tabelas necessárias. Observe que a atualização inserirá automaticamente a linha se a linha nas tabelas estiver ausente. Se você tiver apenas uma única tabela, então poderá configurar chaves exclusivas. Para uma operação de atualização, o Firehose coloca um arquivo de exclusão seguido por uma inserção.

Você pode configurar chaves exclusivas por tabela como parte da criação do stream do Firehose ou definir de identifier-field-idsforma nativa no Iceberg durante a operação de criar tabela ou alterar tabela. Configurar chaves exclusivas por tabela durante a criação do fluxo é opcional. Se você não configurar chaves exclusivas por tabela durante a criação do fluxo, o Firehose verificará os `identifier-field-ids` para as tabelas necessárias e os usará como chaves exclusivas. Se ambos não estiverem configurados, a entrega de dados com operações de atualização e exclusão falhará.

Para configurar essa seção, forneça o nome do banco de dados, o nome da tabela e as chaves exclusivas das tabelas nas quais você deseja atualizar ou excluir dados. Você só pode ter uma

entrada para cada tabela na configuração. Opcionalmente, você também pode escolher fornecer um prefixo de bucket de erros se os dados da tabela falharem na entrega, conforme mostrado no exemplo a seguir.

```
[
  {
    "DestinationDatabaseName": "MySampleDatabase",
    "DestinationTableName": "MySampleTable",
    "UniqueKeys": [
      "COLUMN_PLACEHOLDER"
    ],
    "S3ErrorOutputPrefix": "OPTIONAL_PREFIX_PLACEHOLDER"
  }
]
```

Especificação da duração da repetição

É possível usar essa configuração para especificar a duração, em segundos, durante a qual o Firehose deve tentar novamente, caso encontre falhas na gravação nas tabelas do Apache Iceberg no Amazon S3. É possível definir qualquer valor de 0 a 7.200 segundos para realizar novas tentativas. Por padrão, o Firehose tenta novamente por 300 segundos.

Como lidar com falha na entrega ou no processamento

Você deve configurar o Firehose para entregar registros a um bucket de backup do S3 caso ele encontre falhas no processamento ou na entrega de um fluxo após a expiração da duração da nova tentativa. Para isso, configure o bucket de backup do S3 e o prefixo de saída de erro do bucket de backup do S3 nas Configurações de backup no console.

Configuração de sugestões de buffer

O Firehose armazena em buffer os dados de streaming recebidos na memória até um determinado tamanho (Tamanho do armazenamento em buffer) e por um determinado período (Intervalo de armazenamento em buffer) antes de entregá-los às tabelas do Apache Iceberg. É possível escolher um tamanho de buffer de 1 a 128 MiBs e um intervalo de buffer de 0 a 900 segundos. Sugestões de valores de buffer mais altos resultam em menor número de gravações no S3, menor custo de compactação devido a arquivos de dados maiores e tempo de execução de consulta mais rápido, mas com maior latência. Sugestões de valores buffer mais baixos entregam os dados com menor latência.

Definir as configurações avançadas

É possível configurar criptografia do lado do servidor, registro de erros em log, permissões e tags para suas tabelas do Apache Iceberg. Para obter mais informações, consulte Definir as configurações avançadas. É necessário adicionar o perfil do IAM que você criou como parte do???. O Firehose assumirá o perfil para acessar as tabelas do AWS Glue e gravar nos buckets do Amazon S3.

A criação do fluxo do Firehose pode demorar vários minutos para ser concluída. Depois de criar com êxito o fluxo do Firehose, será possível começar a ingerir dados nele e visualizar os dados nas tabelas do Apache Iceberg.

Encaminhamento dos registros recebidos para uma única tabela do Iceberg

Se você quiser que o Firehose insira dados em uma única tabela do Iceberg, basta configurar um único banco de dados e tabela na configuração do seu fluxo, conforme mostrado no exemplo de JSON a seguir. Para uma única tabela, você não precisa da expressão JQ e da função do Lambda para fornecer as informações de encaminhamento ao Firehose. Se você fornecer esses campos junto com o JQ ou o Lambda, o Firehose receberá informações do JQ ou do Lambda.

```
[
  {
    "DestinationDatabaseName": "UserEvents",
    "DestinationTableName": "customer_id",
    "UniqueKeys": [
      "COLUMN_PLACEHOLDER"
    ],
    "S3ErrorOutputPrefix": "OPTIONAL_PREFIX_PLACEHOLDER"
  }
]
```

Neste exemplo, o Firehose encaminha todos os registros de entrada para a tabela `customer_id` no banco de dados `UserEvents`. Se você quiser realizar operações de atualização ou exclusão em uma única tabela, precisará fornecer a operação para cada registro de entrada ao Firehose usando o método ou JSONQueryo método do Lambda.

Encaminhamento de registros recebidos para diferentes tabelas do Iceberg

O Firehose pode encaminhar registros recebidos em um fluxo para diferentes tabelas do Iceberg com base no conteúdo do registro. Por exemplo, considere o exemplo de registro de entrada a seguir.

```
{
  "deviceId": "Device1234",
  "timestamp": "2024-11-28T11:30:00Z",
  "data": {
    "temperature": 21.5,
    "location": {
      "latitude": 37.3324,
      "longitude": -122.0311
    }
  },
  "powerlevel": 84,
  "status": "online"
}
```

```
{
  "deviceId": "Device4567",
  "timestamp": "2023-11-28T10:40:00Z",
  "data": {
    "pressure": 1012.4,
    "location": {
      "zipcode": 24567
    }
  },
  "powerlevel": 82,
  "status": "online"
}
```

Neste exemplo, o campo **`deviceId`** tem dois valores possíveis: `Device1234` e `Device4567`. Quando um registro de entrada tem o **`deviceId`** campo como`Device1234`, queremos gravar o registro em uma tabela do Iceberg chamada`Device1234`, e quando um registro de entrada tem o **`deviceId`** campo como`Device4567`, queremos gravar o registro em uma tabela chamada. `Device4567`

Observe que os registros com `Device1234` e `Device4567` podem ter um conjunto diferente de campos mapeados para colunas diferentes na tabela do Iceberg correspondente. Os registros recebidos podem ter uma estrutura JSON aninhada, na qual o **`deviceId`** pode ser aninhado dentro do registro JSON. Nas próximas seções, discutiremos como é possível encaminhar registros para tabelas diferentes fornecendo as informações de encaminhamento apropriadas para o Firehose nesses cenários.

Fornecimento das informações de encaminhamento para o JSONQuery Firehose com expressão

A maneira mais simples e econômica de fornecer informações de encaminhamento de registros ao Firehose é fornecendo JSONQuery uma expressão. Com essa abordagem, você fornece JSONQuery expressões para três parâmetros — `Database NameTable Name`, e (opcionalmente)`Operation`. O Firehose usa a expressão que você fornece para extrair informações dos registros de fluxo de entrada para encaminhar os registros.

O `Database Name` parâmetro especifica o nome do banco de dados de destino. O `Table Name` parâmetro especifica o nome da tabela de destino. `Operation`é um parâmetro opcional que indica se o registro do fluxo de entrada como um novo registro na tabela de destino ou se deve modificar ou excluir um registro existente na tabela de destino. O campo Operação deve ter um dos valores a seguir: `insert`, `update` ou `delete`.

Para cada um desses três parâmetros, é possível fornecer um valor estático ou uma expressão dinâmica em que o valor é recuperado do registro de entrada. Por exemplo, se você quiser entregar todos os registros do fluxo de entrada para um único banco de dados chamado `IoTevents`, o nome do banco de dados teria um valor estático de `“IoTevents”`. Se o nome da tabela de destino precisar ser obtido a partir de um campo no registro de entrada, o Nome da tabela é uma expressão dinâmica que especifica o campo no registro de entrada do qual o nome da tabela de destino precisa ser recuperado.

No exemplo a seguir, usamos um valor estático para o nome do banco de dados, um valor dinâmico para o nome da tabela e um valor estático para a operação. Observe que especificar a operação é opcional. Se nenhuma operação for especificada, o Firehose inserirá os registros recebidos na tabela de destino como novos registros, por padrão.

```
Database Name : "IoTevents"
Table Name : .deviceId
Operation : "insert"
```

Se o `deviceId` campo estiver aninhado no registro JSON, especificamos o nome da tabela com as informações do campo aninhado como. `.event.deviceId`

> **Note**
>
> - Ao especificar a operação como `update` ou`delete`, você deve especificar chaves exclusivas para a tabela de destino ao configurar seu fluxo do Firehose, ou definir identifier-field-idsno Iceberg ao executar as operações create table ou alter table no Iceberg. Se você não especificar isso, o Firehose gerará um erro e enviará os dados para um bucket de erros do S3.
> - Os valores `Database Name` e `Table Name` devem corresponder exatamente aos nomes do banco de dados e da tabela de destino. Se eles não corresponderem, o Firehose gerará um erro e enviará os dados para o bucket de erros do S3.

Fornecimento de informações de encaminhamento usando uma função do AWS Lambda

Pode haver cenários em que você tenha regras complexas que determinem como encaminhar os registros recebidos para uma tabela de destino. Por exemplo, é possível ter uma regra que defina se um campo contém o valor A, B ou F, que deve ser encaminhado para uma tabela de destino chamada `TableX`, ou talvez você queira aumentar o registro do fluxo de entrada adicionando atributos adicionais. Por exemplo, se um registro contiver um campo `device_id` como 1, talvez você queira adicionar outro campo que seja `device_type` “modem” gravar o campo adicional na coluna da tabela de destino. Nesses casos, é possível transformar o fluxo da fonte usando uma AWS Lambda função do no Firehose e fornecer informações de encaminhamento como parte da saída da função de transformação do Lambda. Para entender como é possível transformar o fluxo da fonte usando uma AWS Lambda função do no Firehose, consulte Transformação de dados da fonte no Amazon Data Firehose.

Quando você usa o Lambda para a transformação de um stream de origem no Firehose, a saída deve conter parâmetros `recordId`, `result`, e ou. `data` `KafkaRecordValue` O parâmetro `recordId` contém o registro do fluxo de entrada, `result` indica se a transformação teve êxito e `data` contém a saída transformada codificada em Base64 da sua função do Lambda. Para obter mais informações, consulte ???.

```
{
```

```
  "recordId": "49655962066601463032522589543535113056108699331451682818000000",
  "result": "Ok",
  "data": "1IiwiI6ICJmYWxsIiwgImdgU21IiwiI6ICJmYWxsIiwg==tcHV0ZXIgU2NpZW5jZSIsICJzZW1"
}
```

Para especificar informações de encaminhamento para o Firehose sobre como encaminhar o registro de fluxo para uma tabela de destino como parte da sua função do Lambda, a saída da sua função do Lambda deve conter uma seção adicional para `metadata`. O exemplo a seguir mostra como a seção de metadados é adicionada à saída do Lambda para um fluxo do Firehose que usa o Kinesis Data Streams como fonte de dados para instruir o Firehose de que ele deve inserir o registro como um novo registro na tabela de nome `Device1234` do banco de dados `IoTevents`.

```
 {
"recordId": "49655962066601463032522589543535113056108699331451682818000000",
    "result": "Ok",
    "data":
 "1IiwiI6ICJmYWxsIiwgImdgU21IiwiI6ICJmYWxsIiwg==tcHV0ZXIgU2NpZW5jZSIsICJzZW1",

    "metadata":{
"otfMetadata":{
            "destinationTableName":"Device1234",
            "destinationDatabaseName":"IoTevents",
            "operation":"insert"
        }
    }
  }
```

Da mesma forma, o exemplo a seguir mostra como é possível adicionar a seção de metadados à saída do Lambda para um Firehose que use o Amazon Managed Streaming for Apache Kafka como fonte de dados para instruir o Firehose a inserir o registro como um novo registro em uma tabela chamada no banco de dados. `Device1234` `IoTevents`

```
{
"recordId": "49655962066601463032522589543535113056108699331451682818000000",
    "result": "Ok",
    "kafkaRecordValue":
 "1IiwiI6ICJmYWxsIiwgImdgU21IiwiI6ICJmYWxsIiwg==tcHV0ZXIgU2NpZW5jZSIsICJzZW1",

    "metadata":{
"otfMetadata":{
            "destinationTableName":"Device1234",
```

```
            "destinationDatabaseName":"IoTevents",
            "operation":"insert"
        }
    }
}
```

Para este exemplo,

- `destinationDatabaseName` refere-se ao nome do banco de dados de destino e é um campo obrigatório.
- `destinationTableName` refere-se ao nome da tabela de destino e é um campo obrigatório.
- `operation` é um campo opcional com valores possíveis de `insert`, `update` e `delete`. Se você não especificar nenhum valor, a operação padrão é `insert`.

> **Note**
>
> - Ao especificar a operação como `update` ou`delete`, você deve especificar chaves exclusivas para a tabela de destino ao configurar seu fluxo do Firehose, ou definir identifier-field-idsno Iceberg ao executar as operações create table ou alter table no Iceberg. Se você não especificar isso, o Firehose gerará um erro e enviará os dados para um bucket de erros do S3.
> - Os valores `Database Name` e `Table Name` devem corresponder exatamente aos nomes do banco de dados e da tabela de destino. Se eles não corresponderem, o Firehose gerará um erro e enviará os dados para o bucket de erros do S3.
> - Quando seu fluxo do Firehose tiver uma função de transformação e uma expressão do Lambda JSONQuery , o Firehose primeiro verificará o campo de metadados na saída do Lambda para determinar como encaminhar o registro para a tabela de destino apropriada e, em seguida, examinará a saída da expressão em busca de campos ausentes. JSONQuery
>
>   Se o Lambda ou a JSONQuery expressão não fornecerem as informações de encaminhamento necessárias, o Firehose assumirá isso como um cenário de tabela única e procurará informações de tabela única na configuração de chaves exclusivas.
>
>   Para obter mais informações, consulte os Encaminhamento de registros de entrada a uma tabela única do Iceberg. Se o Firehose não conseguir determinar as informações

de encaminhamento e corresponder o registro a uma tabela de destino especificada, ele entregará os dados ao seu bucket de erros do S3 especificado.

Amostra de função do Lambda

Essa função do Lambda é um exemplo de código Python que analisa os registros de fluxo de entrada e adiciona campos obrigatórios para especificar como os dados devem ser gravados em tabelas específicas. É possível usar esse código de exemplo para adicionar a seção de metadados para informações de encaminhamento.

```
import json
import base64


def lambda_handler(firehose_records_input, context):
    print("Received records for processing from DeliveryStream: " +
 firehose_records_input['deliveryStreamArn'])

    firehose_records_output = {}
    firehose_records_output['records'] = []


    for firehose_record_input in firehose_records_input['records']:

        # Get payload from Lambda input, it could be different with different sources
        if 'kafkaRecordValue' in firehose_record_input:
            payload_bytes =
 base64.b64decode(firehose_record_input['kafkaRecordValue']).decode('utf-8')
        else
            payload_bytes =
 base64.b64decode(firehose_record_input['data']).decode('utf-8')

        # perform data processing on customer payload bytes here

        # Create output with proper record ID, output data (may be different with
 different sources), result, and metadata
        firehose_record_output = {}

        if 'kafkaRecordValue' in firehose_record_input:
            firehose_record_output['kafkaRecordValue'] =
 base64.b64encode(payload_bytes.encode('utf-8'))
```

```
        else
            firehose_record_output['data'] =
 base64.b64encode(payload_bytes.encode('utf-8'))

        firehose_record_output['recordId'] = firehose_record_input['recordId']
        firehose_record_output['result'] =  'Ok'
        firehose_record_output['metadata'] = {
            'otfMetadata': {
                'destinationDatabaseName': 'your_destination_database',
                'destinationTableName': 'your_destination_table',
                'operation': 'insert'
                }
            }
        firehose_records_output['records'].append(firehose_record_output)
    return firehose_records_output
```

Monitorar métricas

Para entrega de dados às tabelas do Apache Iceberg, o Firehose emite as CloudWatch métricas a seguir em nível de fluxo.

Métrica	Descrição
`DeliveryToIceberg.Bytes`	O número de bytes entregues às tabelas do Apache Iceberg no período especificado. Unidades: bytes
`DeliveryToIceberg.IncomingRowCount`	Número de registros que o Firehose tenta entregar às tabelas do Apache Iceberg. Unidades: contagem
`DeliveryToIceberg.SuccessfulRowCount`	Número de linhas com êxito entregues às tabelas do Apache Iceberg. Unidades: contagem
`DeliveryToIceberg.FailedRowCount`	Número de linhas com falha entregues ao bucket de backup do S3.

Métrica	Descrição
	Unidades: contagem
`DeliveryToIceberg.DataFreshness`	A idade (da chegada ao Farehose até agora) do registro mais antigo no Firehose. Quaisquer registros anteriores a esse foram enviados para as tabelas do Apache Iceberg. Unidades: segundos
`DeliveryToIceberg.Success`	Soma das confirmações com êxito para as tabelas do Apache Iceberg.
`JQProcessing.Duration`	A quantidade de tempo necessária para executar a expressão JQ. Unidade: milissegundos

Noções básicas sobre os tipos de dados com suporte

O Firehose oferece suporte a todos os tipos de dados primitivos e complexos com suporte no Apache Iceberg. Para obter mais informações, consulte Esquemas e tipos de dados. Ao enviar dados binários como uma string, você deve usar os tipos de codificação com suporte no Firehose: Base64 básico, MIME Base64, Base64 seguro para URL e nome de arquivo e Hex. Para tipos de dados de carimbo de data/hora, você deve sempre enviar em microssegundos.

Exemplos de tipos de dados

A seção a seguir mostra exemplos de diferentes tipos de dados.

MapType

```
{
    "destination_column_0":
 {"WP5o0JOkuIQcDPcsvpJJygF1xzaOSq0wUlgTwuIeCEzgVneGxA":"PO3ReF3auyDqbfonx9Cd8NTmcQnqnw7JuZOCWwI
    "destination_column_1": "{\"{\\\"destination_nested_column_0\\\": \
\\"18:56:14.974\\\", \\\"destination_nested_column_1\\\": 241.86246}\":
\"M07kAvYdHvBh61F7RzfxtEd39YQI33LnM2NbGS67DOFFsRUyUUujKT5VnK7Wtfz1mHNeIix6FAY9cYpwTdedgr9XnFwG0
\",\"{\\\"destination_nested_column_0\\\": \\\"18:56:14.974\
\\", \\\"destination_nested_column_1\\\": 562.56384}\":
```

```
\"9GlxhDCt95LxBo51HybBZihqOqf6EU8jrDu7NMpxtGB2dY6q6kXpvxIrFuMdqHCJKIZIcDikwggLniUm8kgE4d
\",\"{\\\"destination_nested_column_0\\\": \\\"18:56:14.974\
\\", \\\"destination_nested_column_1\\\": 496.03268}\":
\"keTJZYLNvLRB50DMKzEI6M0AM4mueyNnA1m2YVnYdDwyxUpPqkb72Q6LiX0B9s8gCjZ6trW6C1PFk9KNBIpxYsj5Tc5Xs
\",\"{\\\"destination_nested_column_0\\\": \\\"18:56:14.974\\\", \\
\"destination_nested_column_1\\\": 559.0878}\":
\"mG0ZET84BUF28E312UCIWgmypyQFSUODH9NAMAnF3LJEutbooZWcBt97PP5AHaopNvC8pQZ4mGXB9hmVmjUNmuj5Qanyx
\",\"{\\\"destination_nested_column_0\\\": \\\"18:56:14.974\
\\", \\\"destination_nested_column_1\\\": 106.845245}\":
\"aidoVYrzu8gcLRkVVUyTKCN9gqTUFYi8uJQsrXEFEYl1f9ool7JhAtg9QKG5BBu67Ngb95ENsNKQyCHNImSu5x4hMnmHU
\"}"
}
```


DecimalType

```
{
    "destination_column_0": 9455262425851.1342772,
    "destination_column_1": "9455262425851.1342772",
    "destination_column_2": 9455262425852
}
```

BinaryType (base64-default, base64-mime, base64-url-safe, hex)


```
{
    "destination_column_0": "AsYhnHD\/Ra54hITl1daNV9glOjtWPEfopH
+PjgUKHYB6K7UcYi4K19b80wD4J\/93x5tyh+0y
+k5cMljVRlmfIkIuLxl9ERBiPPLhf4+yoJ2k70VavPnYWmNLs1hLDHlfeEMIfVhrqOGzJMoA
+CBAWXfIuiG420JSQP5iAx5xFG\/
mOfkM5zYothje8OGXltdthcCL6WYBiP0SlwXcE0uMeRfwclAc9fTOBz6RzdJlHhUDjoAXg
+4cvly27F82XpuGMNwpUj98AOrgbh2MoU9yvsM9ZrjD0eGVgOZP8Ky7Za4oE\/oK8j
+qABF6XV712iA6pVtTNJFvX6Ey3ssNYvno+LYF5ZsySs2rB5AbVM73RfOPqdS\/c\/
r3MEqoEqt+nPx6eGam4WSA+0swztt7aLdrlX6yK7xJeIJ0rTlIDBo0ZUaw011ykY
\/8Bvy+4byoPlmr4Z5yhN1z3ZTOkx7eDR6xMv+vDVsDBtItVazDwHgDy41r
\/hQNeNedPKrozc8TY9k7wZre\/6V2lCa3BmT8Uu9b9ydjR9z+fCSdG
+VRv35nz5kdqdKy8YIrynYs4eOcjh8jH3UwVYrYQcnWkBAfF7Xk9CoPVnL3ciHZtyiZOaTGIj9rO0xX\/
W5dGe9\/4YChs6LbD584kxLTxvHgSl4vadaTGNKci3SvNmZNsz8ducxtNXF\/Tv2DUub465hzgpaLPur3+MB
+kfdN2YXUfqb
+xJAgxThWfUe151nrH0EPow9lgSlp21rUBGznJAvPRl1ExGIAuc7JYAoUrJUkx5Hfl6PekPDhqt7+yJwCB8qxhTTryxo
+bjtai4ndRCGcuCaxT8KkOcXsS37urd3YGSDMinZdMNVc646s25415qK6nBRlqqAY8+EYmcUIVB9XcNdke4zoUfhVQoruwi
\/kFafoulo5DEoMOyaH1N2HCSxG5tZXNQocSZPaY8efZYMCpmDXsPAzkmgSkYRDSu\/r3wUqROa2tGK5\/
pQY24v+Jq0U\/jQ99GShlU283nZ85ot2ocbtMAgD\/WsrSEh6lNt9RaI3HfA7\/HcH\/
fgr9jsTtxDgZhabTBwwDwX0zjWGx1bCuTLKBN7byxg9ZvAVgqwPS4HERLer5T5UkKf74zn9Eq3HYH1Q5JpyDUx
+im7mte1sprf1+A24kksVU\/MD9aP9N8\/QDsQ13gkhOn5KwFMz3BC2Vw5gL
```

```
+gGNHFKDRL6wGIfhuYcx9LucolZ1yNy9Gbb3ioWSSufyFpyXqtndDLPI5QS1SJpJm2KDyqcH1SmRLIhd9MNRUC73EAEm
+NO5wxPzBRSjhCHZpf8SrYITWJl7K3XzGOfPFh2NgES3jMP9cvSXO6yyICcep2HBYGbFflni89+Rw==",
    "destination_column_1": "AsYhnHD\/Ra54hITl1daNV9glOjtWPEfopH
+PjgUKHYB6K7UcYi4K19b80wD4J\/93x5tyh+0y+k5c\r
\nMljVRlmfIkIuLxl9ERBiPPLhf4+yoJ2k70VavPnYWmNLs1hLDHlfeEMIfVhrqOGzJMoA+CBAWXfI\r
\nuiG420JSQP5iAx5xFG\/mOfkM5zYothje8OGXltdthcCL6WYBiP0SlwXcE0uMeRfwclAc9fTOBz6R\r
\nzdJlHhUDjoAXg+4cvly27F82XpuGMNwpUj98AOrgbh2MoU9yvsM9ZrjD0eGVgOZP8Ky7Za4oE\/oK\r
\n8j+qABF6XV712iA6pVtTNJFvX6Ey3ssNYvno+LYF5ZsySs2rB5AbVM73RfOPqdS\/c\/r3MEqoEqt+
\r\nnPx6eGam4WSA+0swztt7aLdrlX6yK7xJeIJ0rTlIDBo0ZUaw011ykY\/8Bvy+4byoPlmr4Z5yhN1z
\r\n3ZTOkx7eDR6xMv+vDVsDBtItVazDwHgDy41r\/hQNeNedPKrozc8TY9k7wZre\/6V2lCa3BmT8Uu9b
\r\n9ydjR9z+fCSdG+VRv35nz5kdqdKy8YIrynYs4eOcjh8jH3UwVYrYQcnWkBAfF7Xk9CoPVnL3ciHZ
\r\ntyiZOaTGIj9rO0xX\/W5dGe9\/4YChs6LbD584kxLTxvHgSl4vadaTGNKci3SvNmZNsz8ducxtNXF
\/\r\nTv2DUub465hzgpaLPur3+MB+kfdN2YXUfqb+xJAgxThWfUe151nrH0EPow9lgSlp21rUBGznJAvP
\r\nRl1ExGIAuc7JYAoUrJUkx5Hfl6PekPDhqt7+yJwCB8qxhTTryxo+bjtai4ndRCGcuCaxT8KkOcXs\r
\nS37urd3YGSDMinZdMNVc646s25415qK6nBRlqqAY8+EYmcUIVB9XcNdke4zoUfhVQoruwidzDU\/k\r
\nFafoulo5DEoMOyaH1N2HCSxG5tZXNQocSZPaY8efZYMCpmDXsPAzkmgSkYRDSu\/r3wUqROa2tGK5\r
\n\/pQY24v+Jq0U\/jQ99GShlU283nZ85ot2ocbtMAgD\/WsrSEh6lNt9RaI3HfA7\/HcH\/fgr9jsTtxDg
\r\nZhabTBwwDwX0zjWGx1bCuTLKBN7byxg9ZvAVgqwPS4HERLer5T5UkKf74zn9Eq3HYH1Q5JpyDUx+\r
\nim7mte1sprf1+A24kksVU\/MD9aP9N8\/QDsQ13gkhOn5KwFMz3BC2Vw5gL+gGNHFKDRL6wGIfhuYc
\r\nx9LucolZ1yNy9Gbb3ioWSSufyFpyXqtndDLPI5QS1SJpJm2KDyqcH1SmRLIhd9MNRUC73EAEm+NO\r
\n5wxPzBRSjhCHZpf8SrYITWJl7K3XzGOfPFh2NgES3jMP9cvSXO6yyICcep2HBYGbFflni89+Rw==",
    "destination_column_2": "AsYhnHD_Ra54hITl1daNV9glOjtWPEfopH-
PjgUKHYB6K7UcYi4K19b80wD4J_93x5tyh-0y-k5cMljVRlmfIkIuLxl9ERBiPPLhf4-
yoJ2k70VavPnYWmNLs1hLDHlfeEMIfVhrqOGzJMoA-
CBAWXfIuiG420JSQP5iAx5xFG_mOfkM5zYothje8OGXltdthcCL6WYBiP0SlwXcE0uMeRfwclAc9fTOBz6RzdJlHhUDjoAX
qABF6XV712iA6pVtTNJFvX6Ey3ssNYvno-LYF5ZsySs2rB5AbVM73RfOPqdS_c_r3MEqoEqt-
nPx6eGam4WSA-0swztt7aLdrlX6yK7xJeIJ0rTlIDBo0ZUaw011ykY_8Bvy-4byoPlmr4Z5yhN1z3ZTOkx7eDR6xMv-
vDVsDBtItVazDwHgDy41r_hQNeNedPKrozc8TY9k7wZre_6V2lCa3BmT8Uu9b9ydjR9z-fCSdG-
VRv35nz5kdqdKy8YIrynYs4eOcjh8jH3UwVYrYQcnWkBAfF7Xk9CoPVnL3ciHZtyiZOaTGIj9rO0xX_W5dGe9_4YChs6LbD
MB-kfdN2YXUfqb-
xJAgxThWfUe151nrH0EPow9lgSlp21rUBGznJAvPRl1ExGIAuc7JYAoUrJUkx5Hfl6PekPDhqt7-
yJwCB8qxhTTryxo-bjtai4ndRCGcuCaxT8KkOcXsS37urd3YGSDMinZdMNVc646s25415qK6nBRlqqAY8-
EYmcUIVB9XcNdke4zoUfhVQoruwidzDU_kFafoulo5DEoMOyaH1N2HCSxG5tZXNQocSZPaY8efZYMCpmDXsPAzkmgSkYRDS
Jq0U_jQ99GShlU283nZ85ot2ocbtMAgD_WsrSEh6lNt9RaI3HfA7_HcH_fgr9jsTtxDgZhabTBwwDwX0zjWGx1bCuTLKBN7
im7mte1sprf1-A24kksVU_MD9aP9N8_QDsQ13gkhOn5KwFMz3BC2Vw5gL-
gGNHFKDRL6wGIfhuYcx9LucolZ1yNy9Gbb3ioWSSufyFpyXqtndDLPI5QS1SJpJm2KDyqcH1SmRLIhd9MNRUC73EAEm-
NO5wxPzBRSjhCHZpf8SrYITWJl7K3XzGOfPFh2NgES3jMP9cvSXO6yyICcep2HBYGbFflni89-Rw==",
    "destination_column_3":
 "02c6219c70ff45ae788484e5d5d68d57d8253a3b563c47e8a47f8f8e050a1d807a2bb51c622e0ad7d6fcd300f827f
}
```


TimeType (Epoch em microssegundos, objeto LocalTime Java)

```
{
    "destination_column_0": 68175096000,
    "destination_column_1": "18:56:15.096"
}
```

TimestampType.withZone (Epoch em microssegundos, objeto OffsetDateTime Java, objeto Java) LocalDateTime

```
{
    "destination_column_0": 1725476175099000,
    "destination_column_1": "2024-09-04T18:56:15.099Z",
    "destination_column_2": "2024-09-04T18:56:15.099"
}
```

DoubleType

```
{
    "destination_column_0": 9.18477568715142,
    "destination_column_1": "9.18477568715142"
}
```

BooleanType

```
{
    "destination_column_0": true,
    "destination_column_1": "false",
    "destination_column_2": 1,
    "destination_column_3": 0
}
```

FloatType

```
{
    "destination_column_0": 0.6242226,
    "destination_column_1": "0.6242226"
}
```

IntegerType

```
{
```

```
    "destination_column_0": 7,
    "destination_column_1": "7"
}
```

TimestampType.withoutZone (Epoch em microssegundos, objeto LocalDateTime Java, objeto OffsetDateTime Java) ZonedDateTime

```
{
    "destination_column_0": 1725476175114000,
    "destination_column_1": "2024-09-04T18:56:15.114",
    "destination_column_2": "2024-09-04T18:56:15.114Z",
    "destination_column_3": "2024-09-04T18:56:15.114-07:00"
}
```

DateType

```
{
    "destination_column_0": 19970,
    "destination_column_1": "2024-09-04"
}
```

LongType

```
{
    "destination_column_0": 8,
    "destination_column_1": "8"
}
```

UUIDType (Objeto Java UUID)

```
{
    "destination_column_0": "21c5521c-a6d4-48d4-b2c8-7f6d842f72c3"
}
```

ListType

```
{
    "destination_column_0":
 ["s1FSrgb0lGDxfn2iYT0EtlP47aHSjwmLZgrdr1JqRs0dmbeCcQoaLr4Xhi2KIVvmus9ppFdpWIcOHnJ0omhAPhXH0yns
    "destination_column_1": "[{\"destination_nested_column_0\":\"bb00f8e6-
db82-4241-a5c5-0d9c0d2f71a4\",\"destination_nested_column_1\":907.35345},
```

```
{\"destination_nested_column_0\":\"2c77b702-d405-4fe1-beee-fb541d7ab833\",
\"destination_nested_column_1\":544.0026},{\"destination_nested_column_0\":
\"68389200-d6b1-413d-bcd9-fdb931708395\",\"destination_nested_column_1\":153.683},
{\"destination_nested_column_0\":\"bc31cbaa-39cd-4e2f-b357-9ea9ce75532b\",
\"destination_nested_column_1\":977.5165},{\"destination_nested_column_0\":
\"b7d627f9-0d5b-41b7-903a-525488259fba\",\"destination_nested_column_1\":434.17215},
{\"destination_nested_column_0\":\"06b6ec1e-1952-4582-b285-46aaf40064b8\",
\"destination_nested_column_1\":580.33124},{\"destination_nested_column_0\":
\"f04b3bbf-61ad-4c5c-8740-6f666f57c431\",\"destination_nested_column_1\":550.75793}]"
}
```


Recursos

Use os recursos a seguir para saber mais:

- Transmita dados em tempo real para tabelas do Apache Iceberg no Amazon S3 usando o Amazon Data Firehose
- Simplifique a análise de AWS WAF registros com o Apache Iceberg e o Amazon Data Firehose
- Crie um data lake para streaming de dados com o Amazon S3 Tables e o Amazon Data Firehose

Replique as alterações do banco de dados nas tabelas Apache Iceberg com o Amazon Data Firehose

> **Note**
>
> O Firehose oferece suporte ao banco de dados como fonte em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia). Esse recurso está em versão prévia e está sujeito a alterações. Não o use para suas cargas de trabalho de produção.

As organizações usam bancos de dados relacionais para armazenar e recuperar dados transacionais que são otimizados para interagir rapidamente com uma ou algumas linhas de dados por vez. Eles não são otimizados para consultar grandes conjuntos de dados agregados. As organizações transferem dados transacionais de bancos de dados relacionais para armazenamentos de dados analíticos, como lagos de dados, data warehouses e outras ferramentas para casos de uso de análise e aprendizado de máquina. Para manter os armazenamentos de dados analíticos sincronizados com bancos de dados relacionais, é usado um padrão de design chamado captura de dados de alteração (CDC) que permite capturar todas as alterações nos bancos de dados em tempo real. Quando os dados são alterados por meio de INSERT, UPDATE ou DELETE em um banco de dados de origem, essas alterações do CDC devem ser transmitidas continuamente sem afetar o desempenho dos bancos de dados.

O Firehose fornece uma easy-to-use end-to-end solução eficaz para replicar alterações dos bancos de dados MySQL e PostgreSQL em tabelas Apache Iceberg. Com esse recurso, o Firehose permite selecionar bancos de dados, tabelas e colunas específicos que você deseja que o Firehose capture em eventos do CDC. Se você ainda não tem Iceberg Tables, você pode optar pelo Firehose para criar Iceberg Tables. O Firehose cria bancos de dados e tabelas usando o mesmo esquema das tabelas do seu banco de dados relacional. Depois que o stream é criado, o Firehose faz uma cópia inicial dos dados nas tabelas e grava no Apache Iceberg Tables. Quando a cópia inicial é concluída, o Firehose inicia a captura quase contínua das alterações do CDC em tempo real em seus bancos de dados e as replica nas tabelas Apache Iceberg. Se você optar pela evolução do esquema, o Firehose evolui o esquema do Iceberg Table com base nas alterações do esquema nos bancos de dados relacionais.

O Firehose também pode replicar alterações dos bancos de dados MySQL e PostgreSQL para tabelas do Amazon S3. As tabelas do Amazon S3 fornecem armazenamento otimizado para cargas de trabalho de análise em grande escala, com recursos que melhoram continuamente o desempenho das consultas e reduzem os custos de armazenamento de dados tabulares. Com suporte integrado ao Apache Iceberg, você pode consultar dados tabulares no Amazon S3 com mecanismos de consulta populares, incluindo Amazon Athena, Amazon Redshift e Apache Spark. Para obter mais informações sobre as tabelas do Amazon S3, consulte Tabelas do Amazon S3.

Para tabelas do Amazon S3, o Firehose não suporta a criação automática de tabelas. Você deve criar tabelas do S3 antes de criar um stream do Firehose.

Considerações e limitações

> **Note**
>
> O Firehose oferece suporte ao banco de dados como fonte em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia). Esse recurso está em versão prévia e está sujeito a alterações. Não o use para suas cargas de trabalho de produção.

O suporte do Firehose para banco de dados como fonte para Apache Iceberg Tables tem as seguintes considerações e limitações:

- O Firehose é compatível com bancos de dados RDS e Aurora e bancos de dados executados em instâncias da Amazon. EC2
- O Firehose oferece suporte ao MySQL versão 8.0.x e 8.2 e às versões 12, 13, 14, 15 e 16 do PostgreSQL.
- Para MySQL e PostgreSQL, o Firehose é compatível com topologias autônomas, primárias e de réplica, clusters de alta disponibilidade e várias topologias primárias. O Firehose funciona somente com um endpoint de servidor primário.
- O Firehose suporta bancos de dados que estão dentro da Virtual Private Cloud (VPC).
- Como parte da evolução do esquema, o Firehose suporta novas alterações na adição de colunas.
- Durante a pré-visualização, o Firehose suporta no máximo 20 MBPS por stream do Firehose.
- O Firehose suporta um tamanho máximo de linha de 10 MB.
- O Firehose suporta o processamento em ordem de eventos do CDC por chave primária.

- O Firehose fornece exatamente uma replicação de eventos CDC em tabelas Apache Iceberg.
- Para tabelas do Amazon S3, o Firehose não suporta a criação automática de tabelas. Você deve criar tabelas do S3 antes de criar um stream do Firehose.

Pré-requisitos para usar o banco de dados como fonte

> **Note**
>
> O Firehose oferece suporte ao banco de dados como fonte em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia). Esse recurso está em versão prévia e está sujeito a alterações. Não o use para suas cargas de trabalho de produção.

Antes de começar, conclua os pré-requisitos a seguir.

- Configurações do banco de dados de origem — Você precisa das seguintes configurações do banco de dados de origem antes de poder usar o banco de dados como fonte para seu stream do Firehose.
  - Crie uma tabela de marca d'água instantânea com as permissões corretas — Para a cópia inicial (instantânea) dos dados nas tabelas, o Firehose usa uma abordagem de cópia incremental com marcas d'água para acompanhar o progresso. Essa abordagem de cópia incremental ajuda a retomar a cópia de onde ela parou e, em seguida, a recapturar a tabela se houver alguma interrupção. O Firehose usa uma tabela de marca d'água em seu banco de dados para armazenar as marcas d'água necessárias. O Firehose precisa de uma tabela de marcas d'água por fluxo do Firehose. Se a tabela ainda não tiver sido criada antes da criação do stream do Firehose, o Firehose criará essa tabela como parte da criação do stream. Você deve fornecer as permissões corretas para que o Firehose crie essa tabela.
  - Crie um usuário de banco de dados — O Firehose exige uma conta de usuário do banco de dados com as permissões corretas para fazer a cópia inicial das tabelas, ler eventos do CDC dos registros de transações, acessar a tabela de marcas d'água e criar uma tabela de marca d'água, caso ainda não tenha sido criada. Você usará esse nome de usuário e senha do banco de dados como parte das credenciais do Firehose para se conectar ao seu banco de dados durante a configuração do stream.
  - Ativar registros de transações — Os registros de transações registram todas as alterações do banco de dados, como INSERT, UPDATE e DELETE, na ordem em que são confirmadas no

banco de dados. O Firehose lê os registros de transações e replica as alterações nas tabelas Apache Iceberg. Você deve ativar os registros de transações se eles não estiverem ativados.

- Adicione uma regra de entrada e saída — Para permitir a conectividade privada com bancos de dados, você deve adicionar uma regra de entrada e uma regra de saída para tráfego HTTPS e uma regra de entrada para tráfego de banco de dados (MySQL ou PostgreSQL) no grupo de segurança da sua VPC de banco de dados. Para a coluna de origem, use o intervalo IPv4 CIDR da sua VPC.

Para criar uma tabela de marca d'água, um usuário do banco de dados e ativar os registros de transações, siga as etapas em???.

- Habilite a conectividade privada com bancos de dados — O Firehose oferece suporte à conexão com bancos de dados dentro da VPC usando tecnologia. AWS PrivateLink Para habilitar a conectividade privada com bancos de dados, consulte Acessar o Amazon RDS VPCs usando AWS PrivateLink e usando o Network Load Balancer. Aqui estão alguns pontos a serem observados para se conectar a bancos de dados.
  - Essas etapas também se aplicam aos bancos de dados em execução no EC2.
  - Você deve aumentar o tempo limite da função Lambda usada neste exemplo do padrão de 3 segundos para 5 minutos.
  - Antes de executar a função Lambda para atualizar o endereço IP da instância primária para o Network Load Balancer, você deve criar um VPC Endpoint com o nome do serviço como `com.amazonaws.us-east-1.elasticloadbalancing` na VPC do seu banco de dados, para que AWS o Lambda possa se comunicar com o serviço Elastic Load Balancing.
  - Você deve incluir o `firehose.amazonaws.com` diretor de serviço Firehose na lista de permissões para AWS PrivateLink criar em sua VPC. Para obter mais informações, consulte Gerenciar permissões. Não adicione o ARN dessa função de serviço. Adicione apenas `firehose.amazonaws.com` aos diretores de permissão.
  - Você deve permitir que seu serviço de endpoint aceite solicitações de conexão automaticamente, garantindo a desativação da opção Aceitação Obrigatória por meio da Amazon VPC. Isso permite que o Firehose crie a conexão de endpoint necessária sem qualquer intervenção manual. Para obter mais informações sobre como desabilitar a solicitação de conexão, consulte Aceitar ou rejeitar solicitações de conexão.
- Armazenar credenciais em AWS Secrets Manager — O Firehose AWS Secrets Manager usa para recuperar as credenciais usadas para se conectar aos bancos de dados. Adicione as credenciais de usuário do banco de dados que você criou no pré-requisito anterior, como segredos no. AWS

Secrets Manager Para obter mais informações, consulte Authenticate with AWS Secrets Manager in Amazon Data Firehose.

- Crie uma função do IAM com as permissões necessárias — O Firehose precisa de uma função do IAM com permissões específicas para acessar AWS Secrets Manager, AWS Glue tabelas e gravar dados no Amazon S3. A mesma função é usada para conceder AWS Glue acesso aos buckets do Amazon S3. Você precisa dessa função do IAM ao criar Apache Iceberg Tables e um Firehose. Para obter mais informações, consulte Conceda ao Firehose acesso para replicar as alterações do banco de dados nas tabelas Apache Iceberg.
- Crie tabelas Apache Iceberg — O Firehose pode criar tabelas Iceberg automaticamente se você ativar a configuração durante a criação do stream do Firehose. Se você não quiser que o Firehose crie tabelas Iceberg, crie tabelas Iceberg com o mesmo nome e esquema das tabelas do banco de dados de origem. Para obter mais informações sobre como criar tabelas Iceberg usando Glue, consulte Criando tabelas Iceberg. O Firehose não pode criar automaticamente tabelas do Amazon S3.

> **Note**
>
> Você deve criar tabelas Apache Iceberg com o mapeamento a seguir.
>
> - Para MySQL, o nome do banco de dados de origem é mapeado para o nome do AWS Glue banco de dados e o nome da tabela de origem é mapeado para o nome da AWS Glue tabela.
> - Para o PostgreSQL, o nome do banco de dados de origem é mapeado AWS Glue para o banco de dados e o nome do esquema de origem e o nome da tabela são mapeados AWS Glue para o nome da tabela no formato. `<SchemaName>_<TableName>` Se você criar uma tabela sozinho, os esquemas de origem e de destino devem corresponder exatamente.

Configuração do fluxo do Firehose

> **Note**
>
> O Firehose oferece suporte ao banco de dados como fonte em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia). Esse

recurso está em versão prévia e está sujeito a alterações. Não o use para suas cargas de trabalho de produção.

Para criar um stream do Firehose com bancos de dados como fonte, você deve configurar o seguinte:

Configuração de fonte e destino

Para obter dados do seu banco de dados, escolha a fonte do seu stream. O Firehose suporta bancos de dados MySQL e PostgreSQL como fontes de banco de dados. Em seguida, escolha Tabelas do Apache Iceberg como destino e forneça um nome de fluxo do Firehose.

Configurar conectividade do banco de dados

Para que o Firehose se conecte a uma instância de banco de dados, ele precisa de um endpoint de banco de dados, um endpoint de serviço VPC, uma porta e um usuário de banco de dados válido com as credenciais corretas.

- Ponto final do banco de dados — Ponto final do banco de dados do servidor primário do seu cluster de banco de dados. Por exemplo, o endpoint seria um banco de dados autogerenciado `xyz.amazonaws.com` ou um banco de dados RDS. `mydb.123456789012.us-east-1.rds.amazonaws.com`
- Nome do VPC Service Endpoint — O Firehose oferece suporte à conectividade privada com bancos de dados. Você deve fornecer o nome do serviço do VPC endpoint. Por exemplo, o endpoint do serviço pode ser`com.amazonaws.vpce.us-east-1.vpce-svc-XXXXXXXXXXXXX`.
- Porta — Para porta, você deve configurar 3306 para bancos de dados MySQL e 5432 para bancos de dados PostgreSQL.
- Modo SSL — Você pode optar por ativar ou desativar o modo SSL. Se ativado, o Firehose usa para **`verify_identity`** MySQL e o modo SSL para PostgreSQL**`verify-full`**. O certificado deve ser assinado por uma CA confiável. Para obter mais informações, consulte Como usar SSL/TLS para criptografar uma conexão com uma instância de banco de dados ou cluster. Observe que para RDS PostgreSQL e Aurora PostgreSQL, o parâmetro é definido como 1**`force_ssl`**, então você deve especificar o Modo SSL como habilitado na configuração do Firehose ou alterar o parâmetro para 0 no grupo de parâmetros do banco de dados. **`force_ssl`**

- Autenticar com AWS Secrets Manager — Selecione um segredo AWS Secrets Manager que contenha as credenciais para conexão com bancos de dados. Se você não tiver um segredo existente, crie um em AWS Secrets Manager. Para obter mais informações, consulte Authenticate with AWS Secrets Manager in Amazon Data Firehose.

Configurar a captura de dados

Se você quiser que o Firehose capture alterações de bancos de dados, tabelas e colunas específicos, você pode configurá-los como parte da criação do stream do Firehose. Você pode especificar bancos de dados, tabelas e colunas necessários fornecendo expressões regulares para incluí-los ou excluí-los ou fornecendo explicitamente nomes específicos de bancos de dados, tabelas e colunas separados por vírgula.

> **Note**
>
> Como no PostgreSQL o esquema em cada banco de dados contém objetos de banco de dados, como tabelas e visualizações, o nome totalmente qualificado ou a expressão regular devem levar os esquemas em consideração.
> Para o MySQL, o nome totalmente qualificado é **<Sampledatabase>.<SampleTable>** e para o PostgreSQL o nome totalmente qualificado é. <SampleSchema>.<SampleTable>

Aqui estão alguns exemplos de cada tipo.

Bancos de dados

```
Example of sample regular expression (for including databases): .*
Example of explicit naming of tables: <SampleDatabase>
```

Tabelas

```
Example of sample regular expression for excluding tables: <SampleDatabase>.*
Example of explicit naming of tables for MySQL : <SampleDatabase>.<SampleTable1>
Example of explicit naming of tables for PostgreSQL : <SampleSchema>.<SampleTable>
```

Columns

```
Example of sample regular expression (for excluding columns): <SampleDatabase>.*.*
```

```
Example of explicit naming of columns for
 MySQL : <SampleDatabase>.<SampleTable>.<SampleColumn>
Example of explicit naming of columns for
 PostgreSQL : <SampleSchema>.<SampleTable>.<SampleColumn>
```

Configurar chaves substitutas

O Firehose exige chaves exclusivas para que as tabelas configuradas obtenham a cópia inicial dos dados. Se você tiver tabelas sem uma chave primária em seus bancos de dados, deverá fornecer uma chave substituta para essas tabelas. Se todas as suas tabelas tiverem chaves primárias, você não precisará configurar esta seção. Se o Firehose encontrar chaves substitutas ausentes para tabelas sem chaves primárias, o processo de snapshot (cópia inicial) falhará. Nesses cenários, o Firehose gera um erro no Logs. CloudWatch Para chaves substitutas, você deve configurar explicitamente as chaves com um nome totalmente qualificado, conforme mostrado no exemplo a seguir.

Para MySQL

```
SampleDatabase.SampleTable:SampleColumn
```

Para PostgreSQL

```
SampleSchema.SampleTable:SampleColumn
```

Forneça uma tabela de marcas d'água instantâneas

O Firehose usa um mecanismo de marca d'água durante a captura instantânea incremental das tabelas. Você deve fornecer essa tabela de marca d'água de instantâneo que você criou como parte do pré-requisito. Insira a tabela de marca d'água do instantâneo no formato mostrado no exemplo a seguir.

```
For MySQL: DatabaseName.TableName
For PostgreSQL: SchemaName.TableName
```

> **Note**
>
> Não exclua a tabela de marcas d'água nem insira ou exclua manualmente registros da tabela de marcas d'água. Além disso, você não deve revogar a permissão de um usuário do banco de dados criado para o Firehose inserir ou excluir registros da tabela de marcas d'água.

Próxima etapa: ???

Definição de configurações do destino

> **Note**
>
> O Firehose oferece suporte ao banco de dados como fonte em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia). Esse recurso está em versão prévia e está sujeito a alterações. Não o use para suas cargas de trabalho de produção.

O Firehose suporta a entrega de alterações do banco de dados nas tabelas Apache Iceberg. Defina as seguintes configurações de destino para configurar o stream do Firehose com o banco de dados como sua fonte.

Conexão de catálogo de dados

O Apache Iceberg requer um catálogo de dados para gravar nas tabelas do Apache Iceberg. O Firehose se integra às tabelas Apache AWS Glue Data Catalog Iceberg. É possível usar o AWS Glue Data Catalog na mesma conta do fluxo do Firehose ou em uma conta cruzada e na mesma região do fluxo do Firehose (padrão), ou em uma região diferente.

Ativar a criação automática de tabelas

Se você ativar essa opção, o Firehose criará automaticamente os bancos de dados, tabelas e colunas necessários no seu destino de destino com o mesmo nome e esquema dos bancos de dados de origem. Se você habilitar essa opção e se o Firehose encontrar algumas tabelas com o mesmo nome e esquema já presentes, ele usará essas tabelas existentes e criará somente bancos de dados, tabelas e colunas ausentes.

Se você não habilitar essa opção, o Firehose tentará encontrar bancos de dados, tabelas e colunas necessários. Se o Firehose não conseguir encontrá-los, ele gerará um erro e enviará os dados para o bucket de erros do S3.

> **Note**
>
> Para que o Firehose entregue dados às tabelas Iceberg com sucesso, os nomes do banco de dados, da tabela e das colunas, juntamente com o esquema, devem corresponder completamente. Se os nomes dos objetos e esquemas do banco de dados não corresponderem, o Firehose gerará um erro e enviará os dados para um bucket de erros do S3.

Para bancos de dados MySQL, o banco de dados de origem mapeia para o Banco de AWS Glue dados e a tabela de origem é mapeada para a AWS Glue Tabela.

Para o PostgreSQL, o banco de dados de origem mapeia AWS Glue para o Banco de dados e a tabela de origem é mapeada AWS Glue para a Tabela com o nome de. `SchemaName_TableName`

> **Note**
>
> Para tabelas do Amazon S3, o Firehose não suporta a criação automática de tabelas. Você deve criar tabelas do S3 antes de criar um stream do Firehose.

Habilitar a evolução do esquema

Se você habilitar essa opção, o Firehose evoluirá automaticamente o esquema das tabelas Apache Iceberg quando o esquema de origem for alterado. Como parte da evolução do esquema, o Firehose atualmente oferece suporte à adição de novas colunas. Por exemplo, se uma nova coluna for adicionada a uma tabela no lado do banco de dados de origem, o Firehose automaticamente pega essas alterações e adiciona a nova coluna à tabela Apache Iceberg apropriada.

Especificação da duração da repetição

É possível usar essa configuração para especificar a duração, em segundos, durante a qual o Firehose deve tentar novamente, caso encontre falhas na gravação nas tabelas do Apache Iceberg no Amazon S3. É possível definir qualquer valor de 0 a 7.200 segundos para realizar novas tentativas. Por padrão, o Firehose tenta novamente por 300 segundos.

Como lidar com falha na entrega ou no processamento

Você deve configurar o Firehose para entregar registros a um bucket de backup do S3 caso ele falhe no processamento ou entrega de um stream após a expiração da duração da nova tentativa. Para isso, configure o bucket de backup do S3 e o prefixo de saída de erro do bucket de backup do S3.

Configuração de sugestões de buffer

O Firehose armazena em buffer os dados de streaming recebidos na memória até um determinado tamanho (Tamanho do armazenamento em buffer) e por um determinado período (Intervalo de armazenamento em buffer) antes de entregá-los às tabelas do Apache Iceberg. Você pode escolher um tamanho de buffer de 1 a 128 MiBs e um intervalo de buffer de 0 a 900 segundos. Maiores dicas de buffer resultam em menos gravações no S3, menor custo de compactação devido a arquivos de dados maiores e tempo de execução de consultas mais rápido, mas com maior latência. Sugestões de valores buffer mais baixos entregam os dados com menor latência.

Definir as configurações avançadas

Para configurações avançadas, você pode configurar criptografia do lado do servidor, registro de erros, permissões e tags para suas tabelas Apache Iceberg. Para obter mais informações, consulte the section called “Definir as configurações avançadas”. Você deve adicionar a função do IAM que você criou como parte do the section called “Conceder acesso ao Firehose” para usar o Apache Iceberg Tables como destino. O Firehose assumirá a função de acessar AWS Glue tabelas e gravar nos buckets do Amazon S3.

É altamente recomendável que você ative CloudWatch os registros. Se houver algum problema com a conexão do Firehose aos bancos de dados ou com a captura instantânea das tabelas, o Firehose gerará um erro e registrará os registros nos registros configurados. Esse é o único mecanismo que informa sobre os erros.

A criação do fluxo do Firehose pode demorar vários minutos para ser concluída. Depois de criar com êxito o fluxo do Firehose, será possível começar a ingerir dados nele e visualizar os dados nas tabelas do Apache Iceberg.

> **Note**
>
> Configure somente um stream do Firehose para um banco de dados. Ter vários streams Firehose para um banco de dados cria vários conectores para o banco de dados, o que afeta o desempenho do banco de dados.

Depois que um stream do Firehose for criado, o status inicial das tabelas existentes será um instantâneo IN_PROGRESS. Não altere o esquema da tabela de origem quando o status do snapshot estiver definido como IN_PROGRESS. Se você alterar o esquema da tabela quando o instantâneo estiver em andamento, o Firehose ignorará o instantâneo da tabela. Quando o processo de captura de imagem é concluído, seu status muda para CONCLUÍDO.

Monitorar métricas

> **Note**
>
> O Firehose oferece suporte ao banco de dados como fonte em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia). Esse recurso está em versão prévia e está sujeito a alterações. Não o use para suas cargas de trabalho de produção.

Para obter alterações do CDC a partir de bancos de dados, o Firehose emite as seguintes CloudWatch métricas em nível de tabela.

Métrica	Descrição
`DataReadFromDatabaseSource.Bytes`	O número de bytes [brutos] lidos do banco de dados de origem. Unidades: bytes
`DataReadFromDatabaseSource.Records`	O número de registros lidos do banco de dados de origem. Unidades: contagem
`BytesPerSecondLimit`	Limite atual de taxa de transferência em que o Firehose lê do banco de dados de origem. Unidades: bytes/segundo
`FailedValidation.Bytes`	O número de bytes [brutos] que falharam na validação do registro.

Métrica	Descrição
	Unidades: bytes
`FailedValidation.Records`	O número de registros que não foram aprovados na validação de registros. Unidades: contagem
`Dropped.Bytes`	O número de bytes que são descartados. Unidades: bytes
`Dropped.Records`	O número de registros que são descartados. Unidades: contagem

Conceda ao Firehose acesso para replicar as alterações do banco de dados nas tabelas Apache Iceberg

> **Note**
>
> O Firehose oferece suporte ao banco de dados como fonte em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia). Esse recurso está em versão prévia e está sujeito a alterações. Não o use para suas cargas de trabalho de produção.

Você deve ter um perfil do IAM antes de criar um fluxo do Firehose e tabelas do Apache Iceberg usando o AWS Glue. Use as etapas a seguir para criar uma política e um perfil do IAM. O Firehose assume esse perfil do IAM e executa as ações necessárias.

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Crie uma política e escolha JSON no editor de políticas.
3. Adicione a seguinte política em linha que concede permissões ao Amazon S3, como read/write permissões, permissões para atualizar a tabela no catálogo de dados e outras.

JSON

```
{
"Version": "2012-10-17",
    "Statement":
    [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetTable",
                "glue:GetDatabase",
                "glue:UpdateTable",
                "glue:CreateTable",
                "glue:CreateDatabase"
            ],
            "Resource": [
                "arn:aws:glue:<region>:<aws-account-id>:catalog",
                "arn:aws:glue:<region>:<aws-account-id>:database/*",
                "arn:aws:glue:<region>:<aws-account-id>:table/*/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        },
        {
           "Effect": "Allow",
           "Action": [
               "kms:Decrypt",
               "kms:GenerateDataKey"
           ],
```

```
            "Resource": [
                "arn:aws:kms:<region>:<aws-account-id>:key/<key-id>"
            ],
            "Condition": {
             "StringEquals": {
                 "kms:ViaService": "s3.region.amazonaws.com"
                },
                "StringLike": {
                 "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::amzn-s3-demo-
bucket/prefix*"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:<region>:<aws-account-id>:log-group:<log-group-
name>:log-stream:<log-stream-name>"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "<Secret ARN>"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVpcEndpointServices"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Noções básicas sobre os tipos de dados com suporte

> **Note**
>
> O Firehose oferece suporte ao banco de dados como fonte em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia). Esse recurso está em versão prévia e está sujeito a alterações. Não o use para suas cargas de trabalho de produção.

O Firehose oferece suporte a todos os tipos de dados primitivos e complexos com suporte no Apache Iceberg. Para obter mais informações, consulte Esquemas e tipos de dados.

Mapeamento de tipos de dados MySQL para Iceberg

Tipo MySQL	Tipo de dados Iceberg
BOOLEANO, BOOL	boolean
BIT (1)	boolean
BIT (>1)	binary
TINYINT	integer
PEQUENO [(M)]	integer
MÉDIO [(M)]	integer
INT, INTEIRO [(M)]	integer
BIGINT [(M)]	integer
REAL [(M, D)]	flutuação
FLUTUAR [(P)]	flutuação
DUPLO [(M, D)]	flutuação
CARACTERE (M)]	string

Tipo MySQL	Tipo de dados Iceberg
VARCHAR (M)]	string
BINÁRIO (M)]	binário ou string
VARBINÁRIO (M)]	binário ou string
TINYBLOB	binário ou string
TINYTEXT	string
BLOB	binário ou string
TEXT	string
MEDIUMBLOB	binário ou string
MEDIUMTEXT	string
LONGBLOB	binário ou string
LONGTEXT	string
JSON	string
ENUM	string
SET	string
ANO [(2|4)]	integer
TIMESTAMP [(M)]	string
DATE	integer
TEMPO [(M)]	integer
DATA HORA, DATA HORA (0), DATA HORA (1), DATA HORA (2), DATA HORA (3)	integer

Tipo MySQL	Tipo de dados Iceberg
DATA HORA (4), DATA HORA (5), DATA HORA (6)	integer
GEOMETRY	Struct
LINESTRING	Struct
POLYGON	Struct
MULTIPOINT	Struct
MULTILINESTRING	Struct
MULTIPOLYGON	Struct
GEOMETRYCOLLECTION	Struct

Mapeamento de tipos de dados do PostgreSQL para o Iceberg

Tipo PostgreSQL	Tipo de dados do Iceberg
BOOLEAN	boolean
BIT (1)	boolean
BIT (> 1)	binary
POUCO VARIANDO [(M)]	binary
PEQUENO, SÉRIE PEQUENA	integer
INTEIRO, SERIAL	integer
BIGBIT, BIGSERIAL, IDIOTA	integer
REAL	flutuação
DOUBLE PRECISION	flutuação

Tipo PostgreSQL	Tipo de dados do Iceberg
CARACTERE [(M)]	string
VARCHAR [(M)]	string
CARACTERE [(M)]	string
CARACTERE VARIANDO [(M)]	string
TIMESTAMPTZ, TIMESTAMP COM FUSO HORÁRIO	string
HORÁRIOS, HORA COM FUSO HORÁRIO	string
INTERVALO [P]	integer
INTERVALO [P]	string
BYTEA	binário ou string
JSON, JSONB	string
XML	string
UUID	string
POINT	string
LTREE	string
CITEXT	string
INET	string
INT4ALCANCE	string
INT8ALCANCE	string
NUMRANGE	string
ESTRANHO	string

Tipo PostgreSQL	Tipo de dados do Iceberg
GAMA TSTAR	string
INTERVALO DE DATAS	string
ENUM	string
DATE	integer
HORA (1), HORA (2), HORA (3)	integer
HORA (4), HORA (5), HORA (6)	integer
TIMESTAMP (1), TIMESTAMP (2), TIMESTAMP (3)	integer
TIMESTAMP (4), TIMESTAMP (5), TIMESTAMP (6), TIMESTAMP	integer
NUMÉRICO [(M [, D])]	binary
DECIMAL [(M [, D])]	binary
DINHEIRO [(M [, D])]	binary
INET	string
CIDR	string
MACADDR	string
MACADDR8	string
GEOMETRIA (plana)	Struct
GEOGRAFIA (esférica)	Struct

Configurar a conectividade do banco de dados

> **Note**
>
> O Firehose oferece suporte ao banco de dados como fonte em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia). Esse recurso está em versão prévia e está sujeito a alterações. Não o use para suas cargas de trabalho de produção.

Esta seção fornece instruções detalhadas para configurar bancos de dados para trabalhar com o Firehose. Ele abrange a criação de tabelas, funções e permissões necessárias para bancos de dados MySQL e PostgreSQL, incluindo RDS, Aurora e instâncias autogerenciadas em. EC2 O documento enfatiza a importância de criar uma tabela de marca d'água e conceder acesso adequado ao Firehose para replicação e streaming de dados e configuração de registros de transações.

Pontos-chave a serem observados

- O Firehose usa uma tabela de marcas d'água em seu banco de dados para armazenar as marcas d'água necessárias. O Firehose exige uma tabela de marcas d'água para cada stream.
- Procedimentos são fornecidos para MySQL e PostgreSQL para automatizar a configuração.
- São necessárias configurações diferentes para bancos de dados RDS/Aurora autogerenciados.
- Permissões e funções adequadas são cruciais para a funcionalidade do Firehose.

Tópicos

MySQL — RDS, Aurora e bancos de dados autogerenciados em execução na Amazon EC2

> **Note**
>
> O Firehose oferece suporte ao banco de dados como fonte em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia). Esse recurso está em versão prévia e está sujeito a alterações. Não o use para suas cargas de trabalho de produção.

Crie o procedimento SQL a seguir em seu banco de dados e, em seguida, chame o procedimento para criar a tabela de marca d'água, o usuário do banco de dados para que o Firehose acesse o banco de dados e forneça as permissões necessárias para o usuário do banco de dados Firehose. Você pode usar esse procedimento para bancos de dados MySQL, RDS e Aurora MySQL auto-hospedados.

> **Note**
>
> Algumas versões mais antigas do banco de dados podem não suportar a string `IF NOT EXISTS` na linha CREATE PROCEDURE. Nesses casos, remova `IF NOT EXISTS` do CREATE PROCEDURE e use o restante do procedimento.

```
DELIMITER //
CREATE PROCEDURE IF NOT EXISTS setupFirehose(IN databaseName TEXT, IN
 watermarkTableName TEXT, IN firehoseUserName TEXT, IN firehosePassword TEXT)
BEGIN

     -- Create watermark table
    SET @create_watermark_text := CONCAT('CREATE TABLE IF NOT EXISTS ', databaseName,
 '.', watermarkTableName, '(id varchar(64) PRIMARY KEY, type varchar(32), data
 varchar(2048))');
    PREPARE createWatermarkTable from @create_watermark_text;
    EXECUTE createWatermarkTable;
    DEALLOCATE PREPARE createWatermarkTable;

    SELECT CONCAT('Created watermark table with name ', databaseName, '.',
 watermarkTableName) as log;
```

```
     -- Create firehose user
    SET @create_user_text := CONCAT('CREATE USER IF NOT EXISTS ''', firehoseUserName,
 ''' IDENTIFIED BY ''', firehosePassword, '''');
    PREPARE createUser from @create_user_text;
    EXECUTE createUser;
    DEALLOCATE PREPARE createUser;

    SELECT CONCAT('Created user with name ', firehoseUserName) as log;

     -- Grant privileges to firehose user
    -- Edit *.* to database/tables you want to grant Firehose access to
    SET @grant_user_text := CONCAT('GRANT SELECT, RELOAD, SHOW DATABASES, REPLICATION
 SLAVE, REPLICATION CLIENT, LOCK TABLES
ON *.* TO ''', firehoseUserName, '''');
    PREPARE grantUser from @grant_user_text;
    EXECUTE grantUser;
    DEALLOCATE PREPARE grantUser;

    SET @grant_user_watermark_text := CONCAT('GRANT CREATE, INSERT, DELETE ON ',
 watermarkTableName, ' to ', firehoseUserName);
    PREPARE grantUserWatermark from @grant_user_watermark_text;
    EXECUTE grantUserWatermark;
    DEALLOCATE PREPARE grantUserWatermark;

    SELECT CONCAT('Granted necessary permissions to user ', firehoseUserName) AS log;

    FLUSH PRIVILEGES;

     -- Show if binlog enabled/disabled
    SELECT variable_value as "BINARY LOGGING STATUS (log-bin) ::" FROM
 performance_schema.global_variables WHERE variable_name='log_bin';

END //
DELIMITER ;
```

Uso

Chame esse procedimento usando um SQL Client.

```
CALL setupFirehose('database', 'watermark_test', 'new_user', 'Test123');
```

PostgreSQL — bancos de dados RDS e Aurora

> **Note**
>
> O Firehose oferece suporte ao banco de dados como fonte em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia). Esse recurso está em versão prévia e está sujeito a alterações. Não o use para suas cargas de trabalho de produção.

Crie o seguinte procedimento SQL em seu banco de dados para criar uma tabela de marca d'água, uma função para o acesso do Firehose ao banco de dados, fornecer as permissões necessárias para a função Firehose, criar uma função de propriedade do grupo e adicionar a função Firehose ao grupo. Você pode usar esse procedimento para bancos de dados PostgreSQL RDS e Aurora.

> **Note**
>
> Algumas versões mais antigas do banco de dados podem não suportar a string `IF NOT EXISTS` na linha CREATE PROCEDURE. Nesses casos, remova `IF NOT EXISTS` do CREATE PROCEDURE e use o restante do procedimento.

```
CREATE OR REPLACE PROCEDURE setupFirehose(
    p_schema_name TEXT,
    p_database_name TEXT,
    p_watermark_name TEXT,
    p_role_name TEXT,
    p_role_password TEXT,
    p_group_owner_name TEXT
)
LANGUAGE plpgsql
AS $$
BEGIN

    -- Create watermark table
    EXECUTE 'CREATE TABLE IF NOT EXISTS ' || quote_ident(p_database_name) || '.' ||
 quote_ident(p_schema_name) || '.' || quote_ident(p_watermark_name) || '(id varchar(64)
 PRIMARY KEY, type varchar(32), data varchar(2048))';

    RAISE NOTICE 'Created watermark table: %', p_watermark_name;
```

```
    -- Create the role with the given password
    IF EXISTS (
        SELECT FROM pg_catalog.pg_roles
        WHERE rolname = p_role_name)
    THEN
        RAISE NOTICE 'Role % already exists. Skipping creation', p_role_name;
    ELSE
        EXECUTE 'CREATE ROLE ' || p_role_name || ' WITH LOGIN INHERIT PASSWORD ' ||
 quote_literal(p_role_password);
        RAISE NOTICE 'Created role: %', p_role_name;
    END IF;

    -- Grant required privileges to the role
    EXECUTE 'GRANT CREATE ON SCHEMA ' || quote_ident(p_schema_name) || ' TO ' ||
 quote_ident(p_role_name);
    EXECUTE 'GRANT CREATE ON DATABASE ' || quote_ident(p_database_name) || ' TO ' ||
 quote_ident(p_role_name);
    EXECUTE 'GRANT rds_replication TO ' || quote_ident(p_role_name);
    EXECUTE 'ALTER TABLE ' || quote_ident(p_schema_name) || '.' ||
 quote_ident(p_watermark_name) || ' OWNER TO ' || quote_ident(p_role_name);

    -- Create shared ownership role
    IF EXISTS (
        SELECT FROM pg_catalog.pg_roles
        WHERE rolname = p_group_owner_name)
    THEN
        RAISE NOTICE 'Role % already exists. Skipping creation', p_group_owner_name;
    ELSE
        EXECUTE 'CREATE ROLE ' || quote_ident(p_group_owner_name);
        RAISE NOTICE 'Created role: %', p_group_owner_name;
    END IF;

    EXECUTE 'GRANT ' || quote_ident(p_group_owner_name) || ' TO ' ||
 quote_ident(p_role_name);

END;
$$;
```

Uso

Chame esse procedimento usando um SQL Client.

```
CALL
 setupFirehose('public', 'test_db', 'watermark', 'new_role', 'Test123', 'group_role');
```

PostgreSQL — bancos de dados autogerenciados em execução na Amazon EC2

> **Note**
>
> O Firehose oferece suporte ao banco de dados como fonte em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia). Esse recurso está em versão prévia e está sujeito a alterações. Não o use para suas cargas de trabalho de produção.

Crie o seguinte procedimento SQL em seu banco de dados para criar uma tabela de marca d'água, uma função para o acesso do Firehose ao banco de dados, fornecer as permissões necessárias para a função Firehose e criar a função de propriedade do grupo e a função Firehose para o grupo. Você pode usar esse procedimento para bancos de dados PostgreSQL executados em. EC2

> **Note**
>
> Algumas versões mais antigas do banco de dados podem não suportar a string `IF NOT EXISTS` na linha CREATE PROCEDURE. Nesses casos, remova `IF NOT EXISTS` do CREATE PROCEDURE e use o restante do procedimento.

```
CREATE OR REPLACE PROCEDURE setupFirehose(
    p_schema_name TEXT,
    p_database_name TEXT,
    p_watermark_name TEXT,
    p_role_name TEXT,
    p_role_password TEXT,
    p_group_owner_name TEXT
)
LANGUAGE plpgsql
AS $$
BEGIN

    -- Use logical decoding
```

```
    EXECUTE 'ALTER SYSTEM SET wal_level = logical';

    -- Create watermark table
    EXECUTE 'CREATE TABLE IF NOT EXISTS ' || quote_ident(p_database_name) || '.' ||
 quote_ident(p_schema_name) || '.' || quote_ident(p_watermark_name) || '(id varchar(64)
 PRIMARY KEY, type varchar(32), data varchar(2048))';

    RAISE NOTICE 'Created watermark table: %', p_watermark_name;

    -- Create the role with the given password
    IF EXISTS (
        SELECT FROM pg_catalog.pg_roles
        WHERE rolname = p_role_name)
    THEN
        RAISE NOTICE 'Role % already exists. Skipping creation', p_role_name;
    ELSE
        EXECUTE 'CREATE ROLE ' || p_role_name || ' WITH LOGIN INHERIT REPLICATION
 PASSWORD ' || quote_literal(p_role_password);
        RAISE NOTICE 'Created role: %', p_role_name;
    END IF;

    -- Grant required privileges to the role
    EXECUTE 'GRANT CREATE ON SCHEMA ' || quote_ident(p_schema_name) || ' TO ' ||
 quote_ident(p_role_name);
    EXECUTE 'GRANT CREATE ON DATABASE ' || quote_ident(p_database_name) || ' TO ' ||
 quote_ident(p_role_name);
    EXECUTE 'ALTER TABLE ' || quote_ident(p_schema_name) || '.' ||
 quote_ident(p_watermark_name) || ' OWNER TO ' || quote_ident(p_role_name);

    -- Create shared ownership role
    IF EXISTS (
        SELECT FROM pg_catalog.pg_roles
        WHERE rolname = p_group_owner_name)
    THEN
        RAISE NOTICE 'Role % already exists. Skipping creation', p_group_owner_name;
    ELSE
        EXECUTE 'CREATE ROLE ' || quote_ident(p_group_owner_name);
        RAISE NOTICE 'Created role: %', p_group_owner_name;
    END IF;

    EXECUTE 'GRANT ' || quote_ident(p_group_owner_name) || ' TO ' ||
 quote_ident(p_role_name);

END;
```

```
$$;
```

Uso

Chame esse procedimento usando um SQL Client.

```
CALL
 setupFirehose('public', 'test_db', 'watermark', 'new_role', 'Test123', 'group_role');
```

PostgreSQL — compartilhamento da propriedade de tabelas para bancos de dados RDS ou autogerenciados executados na Amazon EC2

> **Note**
>
> O Firehose oferece suporte ao banco de dados como fonte em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia). Esse recurso está em versão prévia e está sujeito a alterações. Não o use para suas cargas de trabalho de produção.

Esse procedimento atualiza as tabelas que você deseja usar com o Firehose para que a propriedade seja compartilhada entre o proprietário original e a função que está sendo usada pelo Firehose. Esse procedimento precisa ser chamado para cada tabela que você deseja usar com o Firehose. Esse procedimento usa a função de grupo que você criou com o procedimento anterior.

> **Note**
>
> Algumas versões mais antigas do banco de dados podem não suportar a string `IF NOT EXISTS` na linha CREATE PROCEDURE. Nesses casos, remova `IF NOT EXISTS` do CREATE PROCEDURE e use o restante do procedimento.

```
CREATE OR REPLACE PROCEDURE grant_shared_ownership(
     p_schema_name TEXT,
    p_table_name TEXT,
    p_group_owner_name TEXT
)
LANGUAGE plpgsql
```

```
AS $$
DECLARE
    l_table_owner TEXT;
BEGIN

    -- Get the owner of the specified table
    SELECT pg_catalog.pg_get_userbyid(c.relowner)
    INTO l_table_owner
    FROM pg_catalog.pg_class c
    WHERE c.relname = p_table_name;

    IF l_table_owner IS NOT NULL THEN

        -- Add table owner to the group
        EXECUTE 'GRANT ' || quote_ident(p_group_owner_name) || ' TO ' ||
 quote_ident(l_table_owner);

        -- Change ownership of table to group
        EXECUTE 'ALTER TABLE ' || quote_ident(p_schema_name) || '.' ||
 quote_ident(p_table_name) || ' OWNER TO ' || quote_ident(p_group_owner_name);
    ELSE
        RAISE EXCEPTION 'Table % not found', p_table_name;
    END IF;
END;
$$;
```

Uso

Chame esse procedimento usando um SQL Client.

```
CALL grant_shared_ownership('public', 'cx_table', 'group_role');
```

Ativar registros de transações

> **Note**
>
> O Firehose oferece suporte ao banco de dados como fonte em todas as regiões AWS GovCloud (US) Regions, exceto Regiões da AWSChina e Ásia-Pacífico (Malásia). Esse recurso está em versão prévia e está sujeito a alterações. Não o use para suas cargas de trabalho de produção.

Os registros de transações registram todas as alterações do banco de dados, como INSERT, UPDATE e DELETE, na ordem em que são confirmadas no banco de dados. O Firehose lê os registros de transações e replica as alterações nas tabelas Apache Iceberg. Você deve ativar os registros de transações, caso ainda não o tenha feito. As seções a seguir mostram como você pode habilitar registros de transações para vários bancos de dados MySQL e PostgreSQL.

MySQL

Self-managed MySQL running on EC2

- Verifique se a opção log-bin está ativada:

```
mysql> SELECT variable_value as "BINARY LOGGING STATUS (log-bin) ::"
FROM performance_schema.global_variables WHERE variable_name='log_bin';
```

- Para bancos de dados em execução EC2, se o log binário estiver DESATIVADO, adicione as propriedades na tabela a seguir ao arquivo de configuração do servidor MySQL. Para obter mais informações sobre como definir os parâmetros, consulte a documentação do MySQL no binlog.

```
server-id         = 223344 # Querying variable is called server_id, e.g.
 SELECT variable_value FROM information_schema.global_variables WHERE
 variable_name='server_id';
log_bin                    = mysql-bin
binlog_format              = ROW
binlog_row_image           = FULL
binlog_expire_logs_seconds  = 864000
```

RDS MySQL

- Se o registro binário não estiver ativado, ative-o com as etapas descritas em Configuração do RDS para registro binário do MySQL.
- Defina o formato de registro binário do MySQL para o formato ROW.
- Defina o período de retenção do log binário de pelo menos 72 horas. Para aumentar o período de retenção do log binário, consulte a documentação do RDS. Por padrão, o período de retenção éNULL, portanto, você deve definir o período de retenção para um valor diferente de zero.

Aurora MySQL

- Se o registro binário não estiver ativado, ative-o para o Aurora MySQL com as etapas de configuração do Aurora para registro binário do MySQL.
- Defina o formato de registro binário do MySQL para o formato ROW.
- Defina o período de retenção do log binário de pelo menos 72 horas. Para aumentar o período de retenção do log binário, consulte Definindo e mostrando a configuração do log binário. Por padrão, o período de retenção éNULL, portanto, você deve definir o período de retenção para um valor diferente de zero.

PostgreSQL

Self-managed PostgreSQL running on EC2

- O script acima para PostgreSQL autogerenciado define o wal_level como. `logical`
- Defina configurações adicionais de retenção de WAL em `postgresql.conf`
  - PostgreSQL 12 — `wal_keep_segments = <int>`
  - PostgreSQL 13+ — `wal_keep_size = <int>`

RDS and Aurora PostgreSQL

- Você deve habilitar a replicação lógica (registro antecipado de gravação) por meio do RDS junto com as configurações de retenção do WAL. Para obter mais informações, consulte Decodificação lógica em uma réplica de leitura.

Aplicação de tags a um fluxo do Firehose

É possível atribuir seus próprios metadados aos fluxos do Firehose que criar no Amazon Data Firehose na forma de tags. Tag é um par de chave-valor definida para um fluxo. Usar tags é uma maneira simples, porém poderosa, de gerenciar AWS recursos e organizar dados, incluindo dados de faturamento.

Você pode especificar tags ao invocar CreateDeliveryStreampara criar um novo stream do Firehose. Para fluxos do Firehose existentes, é possível adicionar, listar e remover tags usando estas três operações:

- TagDeliveryStream
- ListTagsForDeliveryStream
- UntagDeliveryStream

Noções básicas sobre tags

É possível usar as operações da API do Amazon Data Firehose para realizar as tarefas a seguir:

- Adicionar tags a um fluxo do Firehose.
- Listar as tags dos seus fluxos do Firehose.
- Remover tags de um fluxo do Firehose.

É possível usar tags para categorizar seus fluxos do Firehose. Por exemplo, é possível categorizar os fluxos do Firehose por finalidade, proprietário ou ambiente. Como você define a chave e o valor para cada marca, é possível criar um conjunto de categorias personalizado para atender às suas necessidades específicas. Por exemplo, é possível definir um conjunto de tags que o ajude a monitorar fluxos do Firehose por proprietário e aplicação associada.

Estes são diversos exemplos de tags:

- `Project: Project name`
- `Owner: Name`
- `Purpose: Load testing`
- `Application: Application name`

- `Environment: Production`

Se tags forem especificadas na ação `CreateDeliveryStream`, o Amazon Data Firehose realizará uma autorização adicional na ação `firehose:TagDeliveryStream` para verificar se os usuários têm permissões para criar tags. Se essa permissão não for fornecida, as solicitações para criar novos fluxos do Firehose com tags de recursos do IAM falharão com `AccessDeniedException`, conforme a seguir.

```
AccessDeniedException
User: arn:aws:sts::x:assumed-role/x/x is not authorized to perform:
 firehose:TagDeliveryStream on resource: arn:aws:firehose:us-east-1:x:deliverystream/x
 with an explicit deny in an identity-based policy.
```

O exemplo a seguir demonstra uma política que permite aos usuários criar um fluxo do Firehose e aplicar tags.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "firehose:CreateDeliveryStream",
            "Resource": "*",
            }
        },
        {
            "Effect": "Allow",
            "Action": "firehose:TagDeliveryStream",
            "Resource": "*",
            }
        }
    ]
}
```

Monitoramento de custos com o uso de tags

Você pode usar tags para categorizar e monitorar seus AWS custos. Quando você aplica tags aos seus AWS recursos, incluindo streams do Firehose, seu relatório de alocação de AWS custos inclui o uso e os custos agregados por tags. É possível organizar seus custos de vários serviços aplicando

tags que representem categorias de negócios (como centros de custos, nomes de aplicações ou proprietários). Para obter mais informações, consulte Usar tags de alocação de custos para relatórios de faturamento personalizados no Manual do usuário do AWS Billing .

Conheça as restrições das tags

As restrições a seguir se aplicam às tags do Amazon Data Firehose.

Restrições básicas

- O número máximo de tags por recurso (fluxo) é 50.
- As chaves e os valores de tags diferenciam maiúsculas de minúsculas.
- Não é possível alterar nem editar as tags de um fluxo excluído.

Restrições de chaves de marcas

- Cada chave de marca deve ser exclusiva. Se uma tag for adicionada com uma chave que já estiver em uso, a nova tag substituirá o par de chave-valor existente.
- 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 não é possível editá-las ou excluí-las.
- As chaves de tag devem ter entre 1 e 128 caracteres Unicode.
- As chaves de tag devem conter os seguintes caracteres: letras Unicode, dígitos, espaço em branco e os seguintes caracteres especiais: `_ . / = + - @`.

Restrições de valor das tags

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

Segurança no Amazon Data Firehose

A segurança na nuvem AWS é a maior prioridade. Como AWS cliente, você se beneficiará 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 isto 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. A eficácia da nossa segurança é regularmente testada e verificada por auditores de terceiros como parte dos Programas de conformidade da AWS. Para saber mais sobre os programas de conformidade que se aplicam ao Data Firehose, consulte Serviços da AWS em escopo por programa de conformidade.
- Segurança na nuvem — Sua responsabilidade é determinada pelo AWS serviço que você usa. Também existe a responsabilidade por outros fatores, incluindo a confidencialidade de dados, os requisitos da organização e as leis e regulamentos aplicáveis.

Esta documentação ajuda você a entender como aplicar o modelo de responsabilidade compartilhada ao usar o Data Firehose. Os tópicos a seguir mostram como configurar o Data Firehose para atender aos seus objetivos de segurança e conformidade. Você também aprenderá a usar outros AWS serviços que podem ajudá-lo a monitorar e proteger seus recursos do Data Firehose.

Tópicos

Proteção de dados no Amazon Data Firehose

O Amazon Data Firehose criptografa todos os dados em trânsito usando o protocolo TLS. Além disso, quando os dados são armazenados em armazenamento provisório durante o processamento, o Amazon Data Firehose os criptografa usando o AWS Key Management Service e verifica sua integridade usando a soma de verificação.

Se você tiver dados confidenciais, poderá habilitar a criptografia de dados no lado do servidor ao usar o Amazon Data Firehose. Como fazer isso depende da fonte dos seus dados.

> **Note**
>
> 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.

Criptografia no lado do servidor com o Kinesis Data Streams

Quando você envia dados de seus produtores de dados para seu stream de dados, o Kinesis Data Streams criptografa seus dados AWS Key Management Service usando AWS KMS uma chave () antes de armazená-los em repouso. Quando seu fluxo do Firehose lê os dados do fluxo de dados, o Kinesis Data Streams primeiro descriptografa os dados e depois os envia ao Amazon Data Firehose. O Amazon Data Firehose armazena os dados na memória com base nas sugestões de armazenamento em buffer que você especifica. Em seguida, entrega-o aos destinos sem armazenar os dados não criptografados em repouso.

Para obter informações sobre como habilitar a criptografia no lado do servidor para o Kinesis Data Streams, consulte Using Server-Side Encryption no Amazon Kinesis Data Streams Developer Guide.

Criptografia do lado do servidor com Direct PUT ou outras fontes de dados

Se você enviar dados para seu stream do Firehose usando PutRecordou PutRecordBatch, ou se você enviar os dados usando AWS IoT Amazon CloudWatch Logs ou CloudWatch Events, você pode ativar a criptografia do lado do servidor usando a operação. StartDeliveryStreamEncryption

Para parar server-side-encryption, use a StopDeliveryStreamEncryptionoperação.

Também é possível habilitar a SSE ao criar o fluxo do Firehose. Para fazer isso, especifique DeliveryStreamEncryptionConfigurationInputquando você invoca. CreateDeliveryStream

Quando a CMK é do tipo `CUSTOMER_MANAGED_CMK`, se o serviço Amazon Data Firehose não conseguir descriptografar os registros devido a uma `KMSNotFoundException`, `KMSInvalidStateException`, `KMSDisabledException` ou `KMSAccessDeniedException`, o serviço aguardará até 24 horas (o período de retenção) para você resolver o problema. Se o problema persistir depois do período de retenção, o serviço ignorará os registros que passaram pelo período de retenção e não puderam ser descriptografados, e descartará os dados. O Amazon Data Firehose fornece as quatro CloudWatch métricas a seguir que você pode usar para rastrear as quatro AWS KMS exceções:

- `KMSKeyAccessDenied`
- `KMSKeyDisabled`
- `KMSKeyInvalidState`
- `KMSKeyNotFound`

Para obter mais informações sobre essas quatro métricas, consulte the section called "Monitoramento com CloudWatch métricas".

Important

Para criptografar seu stream do Firehose, use symmetric. CMKs O Amazon Data Firehose não oferece suporte a assimetrias. CMKs Para obter informações sobre simétrico e assimétrico CMKs, consulte Sobre simétrico e CMKs assimétrico no guia do desenvolvedor. AWS Key Management Service

Note

Quando você usa uma chave gerenciada pelo cliente (CUSTOMER_MANAGED_CMK) para ativar a criptografia do lado do servidor (SSE) no seu fluxo do Firehose, o serviço do Firehose define um contexto de criptografia sempre que usa sua chave. Como esse contexto de criptografia representa uma ocorrência em que uma chave pertencente à sua AWS conta foi usada, ela é registrada como parte dos registros de AWS CloudTrail eventos da sua AWS conta. Esse contexto de criptografia é gerado pelo sistema pelo serviço do Firehose. Sua

aplicação não deve fazer nenhuma suposição sobre o formato ou o conteúdo do contexto de criptografia definido pelo serviço do Firehose.

Controle de acesso com o Amazon Data Firehose

As seções a seguir abordam como controlar o acesso de entrada e saída dos recursos do Amazon Data Firehose. As informações abordadas incluem como conceder acesso à sua aplicação para que ela possa enviar dados para seu fluxo do Firehose. Eles também descrevem como você pode conceder ao Amazon Data Firehose acesso ao seu bucket do Amazon Simple Storage Service (Amazon S3), ao cluster do Amazon Redshift ou ao cluster do OpenSearch Amazon Service, bem como às permissões de acesso necessárias se você usar Datadog, Dynatrace, MongoDB, New Relic, Splunk ou Sumo LogicMonitor Logic como seu destino. Por fim, neste tópico, você encontrará orientações sobre como configurar o Amazon Data Firehose para que ele possa entregar dados a um destino que pertence a outra conta da AWS . A tecnologia para gerenciar todas essas formas de acesso é AWS Identity and Access Management (IAM). Para obter mais informações sobre o IAM, consulte O que é o IAM?.

Conteúdo

- Ingestão de logs de fluxo da VPC no Splunk usando o Amazon Data Firehose
- Acesso ao Snowflake ou ao endpoint de HTTP
- Concessão ao Firehose de acesso a um destino do Snowflake
- Acesso ao Snowflake na VPC
- Concessão ao Firehose de acesso a um destino de endpoint de HTTP
- Entrega entre contas do Amazon MSK
- Entrega entre contas a um destino do Amazon S3
- Entrega entre contas para um destino OpenSearch de serviço
- Uso de tags para controlar o acesso

Concessão de acesso a seus recursos do Firehose

Para conceder à sua aplicação acesso ao fluxo do Firehose, use uma política semelhante a este exemplo. É possível ajustar as operações de API individuais às quais concede acesso modificando a seção `Action` ou conceder acesso a todas as operações com `"firehose:*"`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "firehose:DeleteDeliveryStream",
                "firehose:PutRecord",
                "firehose:PutRecordBatch",
                "firehose:UpdateDestination"
            ],
            "Resource": [
                "arn:aws:firehose:region:account-id:deliverystream/delivery-
stream-name"
            ]
        }
    ]
}
```

Concessão ao Firehose de acesso ao seu cluster privado do Amazon MSK

Se a fonte do seu fluxo do Firehose for um cluster privado do Amazon MSK, use uma política similar a este exemplo.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Principal": {
        "Service": [
          "firehose.amazonaws.com"
        ]
    },
    "Effect": "Allow",
    "Action": [
      "kafka:CreateVpcConnection"
    ],
    "Resource": "cluster-arn"
    }
  ]
}
```

Você deve adicionar uma política como essa à política baseada em recursos do cluster para conceder à entidade principal do serviço do Firehose permissão de invocar a operação de API `CreateVpcConnection` do Amazon MSK.

Permissão para o Firehose assumir um perfil do IAM

Esta seção descreve as permissões e as políticas que concedem ao Amazon Data Firehose acesso para ingerir, processar e entregar dados da fonte ao destino.

> **Note**
>
> Se você usar o console para criar um stream do Firehose e escolher a opção de criar uma nova função, AWS anexará a política de confiança necessária à função. Por outro lado, se você quiser que o Amazon Data Firehose use um perfil do IAM existente ou se criar um perfil, anexe a política de confiança a seguir a esse perfil para que o Amazon Data Firehose possa

assumi-lo. Edite as políticas para *account-id* substituí-las pelo ID AWS da sua conta. Para obter informações sobre como modificar a relação de confiança de um perfil, consulte Modificação de um perfil.

O Amazon Data Firehose usa um perfil do IAM para todas as permissões de que o fluxo do Firehose precisa para processar e entregar os dados. Certifique-se de que as políticas de confiança a seguir estejam anexadas a esse perfil para que o Amazon Data Firehose possa assumi-lo.

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

Se você escolher o Amazon MSK como fonte do fluxo do Firehose, deverá especificar outro perfil do IAM que conceda ao Amazon Data Firehose permissões para ingerir dados da fonte do cluster do Amazon MSK especificado. Certifique-se de que as políticas de confiança a seguir estejam anexadas a esse perfil para que o Amazon Data Firehose possa assumi-lo.

JSON

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

```
    }
  ]
}
```

Certifique-se de que esse perfil que concede permissões ao Amazon Data Firehose para ingerir dados da fonte do cluster do Amazon MSK especificado conceda as permissões a seguir:

JSON

```
{
   "Version": "2012-10-17",
   "Statement": [{
        "Effect":"Allow",
        "Action": [
           "kafka:GetBootstrapBrokers",
           "kafka:DescribeCluster",
           "kafka:DescribeClusterV2",
           "kafka-cluster:Connect"
         ],
         "Resource": "CLUSTER-ARN"
       },
       {
         "Effect":"Allow",
         "Action": [
           "kafka-cluster:DescribeTopic",
           "kafka-cluster:DescribeTopicDynamicConfiguration",
           "kafka-cluster:ReadData"
         ],
         "Resource": "TOPIC-ARN"
       }]
}
```

Conceda acesso ao Firehose AWS Glue para conversão de formato de dados

Se o seu fluxo do Firehose fizer conversão de formato de dados, o Amazon Data Firehose referenciará as definições de tabela armazenadas no AWS Glue. Para dar ao Amazon Data Firehose o acesso necessário AWS Glue, adicione a seguinte declaração à sua política. Para obter informações sobre como encontrar o ARN da tabela, consulte Specifying AWS Glue Resource. ARNs

```
[{
    "Effect": "Allow",
    "Action": [
        "glue:GetTable",
        "glue:GetTableVersion",
        "glue:GetTableVersions"
    ],
    "Resource": "table-arn"
}, {
    "Sid": "GetSchemaVersion",
    "Effect": "Allow",
    "Action": [
        "glue:GetSchemaVersion"
    ],
    "Resource": ["*"]
}]
```

A política recomendada para obter esquemas do registro do esquema não tem restrições de recursos. Para obter mais informações, consulte exemplos de IAM para desserializadores no Guia do AWS Glue desenvolvedor.

Concessão ao Firehose de acesso a um destino do Amazon S3

Quando você está usando um destino do Amazon S3, o Amazon Data Firehose entrega dados para seu bucket do S3 e, opcionalmente, pode usar uma AWS KMS chave que você possui para criptografia de dados. Se o registro de erros estiver ativado, o Amazon Data Firehose também enviará erros de entrega de dados para seu grupo de CloudWatch registros e streams. Você precisa ter um perfil do IAM ao criar um fluxo do Firehose. O Amazon Data Firehose assume essa função do IAM e obtém acesso ao bucket, à chave, ao grupo de CloudWatch registros e aos fluxos especificados.

Use a política de acesso padrão a seguir para permitir que o Amazon Data Firehose acesse o bucket do S3 e a chave do AWS KMS . Se você não tiver o bucket do S3, adicione `s3:PutObjectAcl` à lista de ações do Amazon S3. Isso concede ao proprietário do bucket acesso total aos objetos entregues pelo Amazon Data Firehose.

JSON

```
{
    "Version": "2012-10-17",
    "Statement":
    [
        {
            "Effect": "Allow",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStream",
                "kinesis:GetShardIterator",
                "kinesis:GetRecords",
                "kinesis:ListShards"
            ],
            "Resource": "arn:aws:kinesis:region:account-id:stream/stream-name"
        },
        {
           "Effect": "Allow",
           "Action": [
               "kms:Decrypt",
               "kms:GenerateDataKey"
           ],
```

```
            "Resource": [
                "arn:aws:kms:region:account-id:key/key-id"
            ],
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "s3.region.amazonaws.com"
                },
                "StringLike": {
                    "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::amzn-s3-
demo-bucket/prefix*"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:region:account-id:log-group:log-group-name:log-
stream:log-stream-name"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction",
                "lambda:GetFunctionConfiguration"
            ],
            "Resource": [
                "arn:aws:lambda:region:account-id:function:function-name:function-
version"
            ]
        }
    ]
}
```

A política acima também tem uma declaração que permite o acesso ao Amazon Kinesis Data Streams. Se você não usar o Kinesis Data Streams como fonte de dados, poderá remover essa instrução. Se você usar o Amazon MSK como sua fonte, poderá substituir essa declaração pela a seguir:

```
{
   "Sid":"",
   "Effect":"Allow",
   "Action":[
      "kafka:GetBootstrapBrokers",
      "kafka:DescribeCluster",
      "kafka:DescribeClusterV2",
      "kafka-cluster:Connect"
   ],
   "Resource":"arn:aws:kafka:{{mskClusterRegion}}:{{mskClusterAccount}}:cluster/
{{mskClusterName}}/{{clusterUUID}}"
},
{
   "Sid":"",
   "Effect":"Allow",
   "Action":[
      "kafka-cluster:DescribeTopic",
      "kafka-cluster:DescribeTopicDynamicConfiguration",
      "kafka-cluster:ReadData"
   ],
   "Resource":"arn:aws:kafka:{{mskClusterRegion}}:{{mskClusterAccount}}:topic/
{{mskClusterName}}/{{clusterUUID}}/{{mskTopicName}}"
},
{
   "Sid":"",
   "Effect":"Allow",
   "Action":[
      "kafka-cluster:DescribeGroup"
   ],
   "Resource":"arn:aws:kafka:{{mskClusterRegion}}:{{mskClusterAccount}}:group/
{{mskClusterName}}/{{clusterUUID}}/*"
}
```

Para obter mais informações sobre como permitir que outros AWS serviços acessem seus AWS recursos, consulte Como criar uma função para delegar permissões a um AWS serviço no Guia do usuário do IAM.

Para saber como conceder ao Amazon Data Firehose acesso a um destino do Amazon S3 em outra conta, consulte the section called “Entrega entre contas a um destino do Amazon S3”.

Conceda ao Firehose acesso às tabelas do Amazon S3

É necessário ter um perfil do IAM para criar um fluxo do Firehose. Use as etapas a seguir para criar uma política e um perfil do IAM. O Firehose assume esse perfil do IAM e executa as ações necessárias.

Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.

Crie uma política e escolha JSON no editor de políticas. Adicione a seguinte política em linha que concede permissões ao Amazon S3, read/write como permissões, permissões para atualizar a tabela no catálogo de dados e outras.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "S3TableAccessViaGlueFederation",
      "Effect": "Allow",
      "Action": [
        "glue:GetTable",
        "glue:GetDatabase",
        "glue:UpdateTable"
      ],
      "Resource": [
        "arn:aws:glue:<region>:<account-id>:catalog/s3tablescatalog/*",
        "arn:aws:glue:<region>:<account-id>:catalog/s3tablescatalog",
        "arn:aws:glue:<region>:<account-id>:catalog",
        "arn:aws:glue:<region>:<account-id>:database/*",
        "arn:aws:glue:<region>:<account-id>:table/*/*"
      ]
    },
    {
      "Sid": "S3DeliveryErrorBucketPermission",
      "Effect": "Allow",
      "Action": [
        "s3:AbortMultipartUpload",
        "s3:GetBucketLocation",
        "s3:GetObject",
        "s3:ListBucket",
```

```
        "s3:ListBucketMultipartUploads",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::<error delivery bucket>",
        "arn:aws:s3:::<error delivery bucket>/*"
      ]
    },
    {
      "Sid": "RequiredWhenUsingKinesisDataStreamsAsSource",
      "Effect": "Allow",
      "Action": [
        "kinesis:DescribeStream",
        "kinesis:GetShardIterator",
        "kinesis:GetRecords",
        "kinesis:ListShards"
      ],
      "Resource": "arn:aws:kinesis:<region>:<account-id>:stream/<stream-name>"
    },
    {
      "Sid":
"RequiredWhenDoingMetadataReadsANDDataAndMetadataWriteViaLakeformation",
      "Effect": "Allow",
      "Action": [
        "lakeformation:GetDataAccess"
      ],
      "Resource": "*"
    },
    {
      "Sid": "RequiredWhenUsingKMSEncryptionForS3ErrorBucketDelivery",
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
      ],
      "Resource": [
        "arn:aws:kms:<region>:<account-id>:key/<KMS-key-id>"
      ],
      "Condition": {
        "StringEquals": {
          "kms:ViaService": "s3.<region>.amazonaws.com"
        },
        "StringLike": {
```

```
          "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::<error delivery
 bucket>/prefix*"
        }
      }
    },
    {
      "Sid": "LoggingInCloudWatch",
      "Effect": "Allow",
      "Action": [
        "logs:PutLogEvents"
      ],
      "Resource": [
        "arn:aws:logs:<region>:<account-id>:log-group:<log-group-name>:log-
stream:<log-stream-name>"
      ]
    },
    {
      "Sid": "RequiredWhenAttachingLambdaToFirehose",
      "Effect": "Allow",
      "Action": [
        "lambda:InvokeFunction",
        "lambda:GetFunctionConfiguration"
      ],
      "Resource": [
        "arn:aws:lambda:<region>:<account-id>:function:<function-name>:<function-
version>"
      ]
    }
  ]
}
```

A política tem declarações que permitem o acesso ao Amazon Kinesis Data Streams, a invocação de funções do Lambda e o acesso às chaves. AWS KMS Se você não usar nenhum desses recursos, poderá remover as respectivas declarações. Se o registro de erros estiver ativado, o Amazon Data Firehose também enviará erros de entrega de dados para seu grupo de CloudWatch registros e streams. Você deve configurar os nomes do grupo de registros e do fluxo de registros para usar essa opção. Para os nomes de grupo de logs e fluxo de logs, consulte Monitoramento do Amazon Data Firehose usando logs CloudWatch .

Nas políticas em linha, `<error delivery bucket>` substitua pelo nome `aws-account-id` do bucket do Amazon S3 e a região por um número Conta da AWS válido e uma região do recurso.

Depois de criar a política, abra o console do IAM em https://console.aws.amazon.com/iam/e crie uma função do IAM com AWS service (Serviço da AWS)o tipo de entidade confiável.

Em Serviço ou Caso de Uso, escolha Kinesis. Em Caso de uso, escolha Kinesis Firehose.

Na próxima página, escolha a política criada na etapa anterior para anexar a esse perfil. Na página de análise, você encontrará uma política de confiança já anexada a essa função, dando permissões ao serviço Firehose para assumir essa função. Quando você cria a função, o Amazon Data Firehose pode assumir que ela executa as operações necessárias nos buckets do AWS Glue S3. Adicione o principal do serviço Firehose à política de confiança da função criada. Para obter mais informações, consulte Permissão para o Firehose assumir um perfil do IAM.

Concessão ao Firehose de acesso a um destino de tabelas do Apache Iceberg

Você deve ter um perfil do IAM antes de criar um fluxo do Firehose e tabelas do Apache Iceberg usando o AWS Glue. Use as etapas a seguir para criar uma política e um perfil do IAM. O Firehose assume esse perfil do IAM e executa as ações necessárias.

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Crie uma política e escolha JSON no editor de políticas.
3. Adicione a seguinte política em linha que concede permissões ao Amazon S3, como read/write permissões, permissões para atualizar a tabela no catálogo de dados, etc.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement":
    [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetTable",
                "glue:GetDatabase",
                "glue:UpdateTable"
            ],
            "Resource": [
```

```
                "arn:aws:glue:<region>:<aws-account-id>:catalog",
                "arn:aws:glue:<region>:<aws-account-id>:database/*",
                "arn:aws:glue:<region>:<aws-account-id>:table/*/*"

            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStream",
                "kinesis:GetShardIterator",
                "kinesis:GetRecords",
                "kinesis:ListShards"
            ],
            "Resource": "arn:aws:kinesis:<region>:<aws-account-
id>:stream/<stream-name>"
        },
        {
           "Effect": "Allow",
           "Action": [
               "kms:Decrypt",
               "kms:GenerateDataKey"
           ],
           "Resource": [
               "arn:aws:kms:<region>:<aws-account-id>:key/<key-id>"

           ],
           "Condition": {
```

```
                "StringEquals": {
                    "kms:ViaService": "s3.region.amazonaws.com"
                },
                "StringLike": {
                    "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::amzn-s3-
demo-bucket/prefix*"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:<region>:<aws-account-id>:log-group:<log-group-
name>:log-stream:<log-stream-name>"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction",
                "lambda:GetFunctionConfiguration"
            ],
            "Resource": [
                "arn:aws:lambda:<region>:<aws-account-id>:function:<function-
name>:<function-version>"
            ]
        }
    ]
}
```

A política acima tem uma declaração que permite o acesso ao Amazon Kinesis Data Streams, invocando funções do Lambda e acesso a chaves do KMS. Se você não usar nenhum desses recursos, poderá remover as respectivas declarações.

Se o registro de erros estiver ativado, o Firehose também enviará erros de entrega de dados para seu grupo de CloudWatch registros e streams. Para isso, é necessário configurar os nomes do grupo de logs e do fluxo de logs. Para os nomes de grupo de logs e fluxo de logs, consulte Monitoramento do Amazon Data Firehose usando logs CloudWatch .

4. Nas políticas em linha, *amzn-s3-demo-bucket* substitua pelo nome aws-account-id do bucket do Amazon S3 e a região por um número Conta da AWS válido e uma região dos recursos.

> **Note**
>
> Esse perfil dá permissão a todos os bancos de dados e tabelas em seu catálogo de dados. Se você quiser, poderá conceder permissões somente para tabelas e bancos de dados específicos.

5. Depois de criar a política, abra o console do IAM e crie um perfil do IAM com o AWS service (Serviço da AWS) como Tipo de entidade confiável.
6. Em Serviço ou Caso de Uso, escolha Kinesis. Em Caso de uso, escolha Kinesis Firehose.
7. Na próxima página, escolha a política criada na etapa anterior para anexar a esse perfil. Na página de análise, você encontrará uma política de confiança já anexada a esse perfil, dando permissões ao serviço do Firehose para assumir esse perfil. Quando você cria o perfil, o Amazon Data Firehose pode assumi-lo para executar as operações necessárias no AWS Glue e nos buckets do S3.

Conceder ao Firehose acesso a um destino do Amazon Redshift

Consulte os pontos a seguir ao conceder acesso ao Amazon Data Firehose ao usar um destino do Amazon Redshift.

Tópicos

- Perfil do IAM e política de acesso
- Acesso da VPC a um cluster provisionado pelo Amazon Redshift ou a um grupo de trabalho do Amazon Redshift sem servidor

Perfil do IAM e política de acesso

Quando você usa um destino do Amazon Redshift, o Amazon Data Firehose entrega os dados ao bucket do S3 como um local intermediário. Opcionalmente, ele pode usar qualquer AWS KMS chave que você possua para criptografia de dados. Em seguida, o Amazon Data Firehose emite carrega os dados do bucket do S3 ao cluster provisionado do Amazon Redshift ou no grupo de trabalho do Amazon Redshift sem servidor. Se o registro de erros estiver ativado, o Amazon Data Firehose também enviará erros de entrega de dados para seu grupo de CloudWatch registros e streams.

O Amazon Data Firehose usa o nome de usuário e a senha especificados do Amazon Redshift para acessar seu cluster provisionado ou grupo de trabalho Amazon Redshift Serverless e usa uma função do IAM para acessar o bucket, a chave, o grupo de logs e os streams especificados. CloudWatch Você precisa ter um perfil do IAM ao criar um fluxo do Firehose.

Use a política de acesso padrão a seguir para permitir que o Amazon Data Firehose acesse o bucket do S3 e a chave do AWS KMS . Se você não é o proprietário do bucket do S3, adicione `s3:PutObjectAcl` à lista de ações do Amazon S3, o que concede ao proprietário do bucket acesso total aos objetos entregues pelo Amazon Data Firehose. A política acima também tem uma instrução que concede acesso ao Amazon Kinesis Data Streams. Se você não usar o Kinesis Data Streams como fonte de dados, poderá remover essa instrução.

JSON

```
{
"Version": "2012-10-17",
    "Statement":
    [
        {
            "Effect": "Allow",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        },
        {
           "Effect": "Allow",
           "Action": [
               "kms:Decrypt",
               "kms:GenerateDataKey"
           ],
           "Resource": [
               "arn:aws:kms:region:account-id:key/key-id"
```

```
            ],
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "s3.region.amazonaws.com"
                },
                "StringLike": {
                    "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::amzn-s3-
demo-bucket/prefix*"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStream",
                "kinesis:GetShardIterator",
                "kinesis:GetRecords",
                "kinesis:ListShards"
            ],
            "Resource": "arn:aws:kinesis:region:account-id:stream/stream-name"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:region:account-id:log-group:log-group-name:log-
stream:log-stream-name"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction",
                "lambda:GetFunctionConfiguration"
            ],
            "Resource": [
                "arn:aws:lambda:region:account-id:function:function-name:function-
version"
            ]
        }
    ]
```

```
}
```

Para obter mais informações sobre como permitir que outros AWS serviços acessem seus AWS recursos, consulte Como criar uma função para delegar permissões a um AWS serviço no Guia do usuário do IAM.

Acesso da VPC a um cluster provisionado pelo Amazon Redshift ou a um grupo de trabalho do Amazon Redshift sem servidor

Se o cluster provisionado do Amazon Redshift ou um grupo de trabalho do Amazon Redshift sem servidor estiver em uma nuvem privada virtual (VPC), ele deve ser acessível publicamente com um endereço IP público. Além disso, conceda ao Amazon Data Firehose acesso ao cluster provisionado do Amazon Redshift ou ao grupo de trabalho do Amazon Redshift sem servidor desbloqueando os endereços IP do Amazon Data Firehose. O Amazon Data Firehose atualmente usa um bloco CIDR para cada região disponível.

Região	Blocos CIDR
Leste dos EUA (Ohio)	`13.58.135.96/27`
Leste dos EUA (Norte da Virgínia)	`52.70.63.192/27`
Oeste dos EUA (Norte da Califórnia)	`13.57.135.192/27`
Oeste dos EUA (Oregon)	`52.89.255.224/27`
AWS GovCloud (Leste dos EUA)	`18.253.138.96/27`
AWS GovCloud (Oeste dos EUA)	`52.61.204.160/27`
Canadá (Central)	`35.183.92.128/27`
Oeste do Canadá (Calgary)	`40.176.98.192/27`

Região	Blocos CIDR
Ásia-Pacífico (Hong Kong)	18.162.221.32/27
Asia Pacific (Mumbai)	13.232.67.32/27
Ásia-Pacífico (Hyderabad)	18.60.192.128/27
Ásia-Pacífico (Seul)	13.209.1.64/27
Ásia-Pacífico (Singapura)	13.228.64.192/27
Ásia-Pacífico (Sydney)	13.210.67.224/27
Ásia-Pacífico (Jacarta)	108.136.221.64/27
Ásia-Pacífico (Tóquio)	13.113.196.224/27
Ásia-Pacífico (Osaka)	13.208.177.192/27
Ásia-Pacífico (Tailândia)	43.208.112.96/27
Ásia-Pacífico (Taipei)	43.212.53.160/27
China (Pequim)	52.81.151.32/27
China (Ningxia)	161.189.23.64/27
Europa (Zurique)	16.62.183.32/27
Europa (Frankfurt)	35.158.127.160/27
Europa (Irlanda)	52.19.239.192/27
Europa (Londres)	18.130.1.96/27
Europa (Paris)	35.180.1.96/27
Europa (Estocolmo)	13.53.63.224/27

Região	Blocos CIDR
Europa (Espanha)	18.100.71.96/27
Oriente Médio (Bahrein)	15.185.91.0/27
México (Central)	78.12.207.32/27
América do Sul (São Paulo)	18.228.1.128/27
Europa (Milão)	15.161.135.128/27
África (Cidade do Cabo)	13.244.121.224/27
Oriente Médio (Emirados Árabes Unidos)	3.28.159.32/27
Israel (Tel Aviv)	51.16.102.0/27
Ásia-Pacífico (Melbourne)	16.50.161.128/27
Ásia-Pacífico (Malásia)	43.216.58.0/27

Para obter mais informações sobre como desbloquear endereços IP, consulte a etapa Autorizar o acesso ao cluster no Guia de conceitos básicos do Amazon Redshift.

Conceder ao Firehose acesso a um destino de serviço público OpenSearch

Quando você está usando um destino de OpenSearch serviço, o Amazon Data Firehose entrega dados para seu cluster de OpenSearch serviços e, ao mesmo tempo, faz backup de todos os documentos falhados ou de todos os documentos em seu bucket do S3. Se o registro de erros estiver ativado, o Amazon Data Firehose também enviará erros de entrega de dados para seu grupo de CloudWatch registros e streams. O Amazon Data Firehose usa uma função do IAM para acessar o domínio de OpenSearch serviço, o bucket do S3, a AWS KMS chave, o grupo de CloudWatch registros e os fluxos especificados. Você precisa ter um perfil do IAM ao criar um fluxo do Firehose.

Use a seguinte política de acesso para permitir que o Amazon Data Firehose acesse seu bucket, domínio de OpenSearch serviço e chave do S3. AWS KMS Se você não for o proprietário do bucket

do S3, adicione `s3:PutObjectAcl` à lista de ações do Amazon S3, o que concede ao proprietário do bucket acesso total aos objetos entregues pelo Amazon Data Firehose. A política acima também tem uma instrução que concede acesso ao Amazon Kinesis Data Streams. Se você não usar o Kinesis Data Streams como fonte de dados, poderá remover essa instrução.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        },
        {
           "Effect": "Allow",
           "Action": [
               "kms:Decrypt",
               "kms:GenerateDataKey"
           ],
           "Resource": [
               "arn:aws:kms:region:account-id:key/key-id"
           ],
           "Condition": {
               "StringEquals": {
                   "kms:ViaService": "s3.region.amazonaws.com"
               },
               "StringLike": {
                   "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::amzn-s3-
demo-bucket/prefix*"
               }
```

```
            }
         },
         {
            "Effect": "Allow",
            "Action": [
                "es:DescribeDomain",
                "es:DescribeDomains",
                "es:DescribeDomainConfig",
                "es:ESHttpPost",
                "es:ESHttpPut"
            ],
            "Resource": [
                "arn:aws:es:region:account-id:domain/domain-name",
                "arn:aws:es:region:account-id:domain/domain-name/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "es:ESHttpGet"
            ],
            "Resource": [
                "arn:aws:es:region:account-id:domain/domain-name/_all/_settings",
                "arn:aws:es:region:account-id:domain/domain-name/_cluster/stats",
                "arn:aws:es:region:account-id:domain/domain-name/index-name*/
_mapping/type-name",
                "arn:aws:es:region:account-id:domain/domain-name/_nodes",
                "arn:aws:es:region:account-id:domain/domain-name/_nodes/stats",
                "arn:aws:es:region:account-id:domain/domain-name/_nodes/*/stats",
                "arn:aws:es:region:account-id:domain/domain-name/_stats",
                "arn:aws:es:region:account-id:domain/domain-name/index-name*/
_stats",
                "arn:aws:es:region:account-id:domain/domain-name/"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStream",
                "kinesis:GetShardIterator",
                "kinesis:GetRecords",
                "kinesis:ListShards"
            ],
            "Resource": "arn:aws:kinesis:region:account-id:stream/stream-name"
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:region:account-id:log-group:log-group-name:log-
stream:log-stream-name"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction",
                "lambda:GetFunctionConfiguration"
            ],
            "Resource": [
                "arn:aws:lambda:region:account-id:function:function-name:function-
version"
            ]
        }
    ]
}
```

Para obter mais informações sobre como permitir que outros AWS serviços acessem seus AWS recursos, consulte Como criar uma função para delegar permissões a um AWS serviço no Guia do usuário do IAM.

Para saber como conceder ao Amazon Data Firehose acesso a um cluster de OpenSearch serviços em outra conta, consulte. the section called “Entrega entre contas para um destino OpenSearch de serviço”

Conceder ao Firehose acesso a um destino de OpenSearch serviço em uma VPC

Se o seu domínio de OpenSearch serviço estiver em uma VPC, certifique-se de conceder ao Amazon Data Firehose as permissões descritas na seção anterior. Além disso, você precisa conceder ao Amazon Data Firehose as seguintes permissões para permitir que ele acesse a VPC do seu domínio OpenSearch de serviço.

- `ec2:DescribeVpcs`
- `ec2:DescribeVpcAttribute`
- `ec2:DescribeSubnets`
- `ec2:DescribeSecurityGroups`
- `ec2:DescribeNetworkInterfaces`
- `ec2:CreateNetworkInterface`
- `ec2:CreateNetworkInterfacePermission`
- `ec2:DeleteNetworkInterface`

> **Important**
>
> Não revogue essas permissões depois de criar o fluxo do Firehose. Se você revogar essas permissões, seu stream do Firehose será degradado ou deixará de fornecer dados ao OpenSearch seu domínio de serviço sempre que o serviço tentar consultar ou atualizar. ENIs

> **Important**
>
> Ao especificar sub-redes para entregar dados ao destino em uma VPC privada, verifique se você tem um número suficiente de endereços IP livres nas sub-redes escolhidas. Se não houver um endereço IP gratuito disponível em uma sub-rede especificada, o Firehose não poderá criar ou ENIs adicionar dados para a entrega de dados na VPC privada, e a entrega será degradada ou falhará.

Ao criar ou atualizar seu stream do Firehose, você especifica um grupo de segurança para o Firehose usar ao enviar dados para seu domínio de serviço. OpenSearch Você pode usar o mesmo grupo de segurança usado pelo domínio do OpenSearch Serviço ou um diferente. Se você especificar um grupo de segurança diferente, certifique-se de que ele permita tráfego HTTPS de saída para o grupo de segurança do domínio do OpenSearch Serviço. Além disso, certifique-se de que o grupo de segurança do domínio OpenSearch Service permita tráfego HTTPS do grupo de segurança que você especificou ao configurar seu stream do Firehose. Se você usa o mesmo grupo de segurança para o stream do Firehose e para o domínio OpenSearch Service, verifique se a regra de entrada do grupo de segurança permite tráfego HTTPS. Para obter mais informações

sobre regras de grupo de segurança, consulte Regras de grupo de segurança na documentação da Amazon VPC.

Conceda ao Firehose acesso a um destino público OpenSearch sem servidor

Quando você está usando um destino OpenSearch sem servidor, o Amazon Data Firehose entrega dados para sua coleção OpenSearch sem servidor e, ao mesmo tempo, faz backup de todos os documentos falhados ou de todos os documentos em seu bucket do S3. Se o registro de erros estiver ativado, o Amazon Data Firehose também enviará erros de entrega de dados para seu grupo de CloudWatch registros e streams. O Amazon Data Firehose usa uma função do IAM para acessar a coleção OpenSearch Serverless, o bucket S3, o grupo e os fluxos de AWS KMS chaves e CloudWatch logs especificados. Você precisa ter um perfil do IAM ao criar um fluxo do Firehose.

Use a seguinte política de acesso para permitir que o Amazon Data Firehose acesse seu bucket S3, domínio OpenSearch sem servidor e chave. AWS KMS Se você não for o proprietário do bucket do S3, adicione `s3:PutObjectAcl` à lista de ações do Amazon S3, o que concede ao proprietário do bucket acesso total aos objetos entregues pelo Amazon Data Firehose. A política acima também tem uma instrução que concede acesso ao Amazon Kinesis Data Streams. Se você não usar o Kinesis Data Streams como fonte de dados, poderá remover essa instrução.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
```

```
        },
        {
           "Effect": "Allow",
           "Action": [
               "kms:Decrypt",
               "kms:GenerateDataKey"
           ],
           "Resource": [
               "arn:aws:kms:region:account-id:key/key-id"
           ],
           "Condition": {
               "StringEquals": {
                   "kms:ViaService": "s3.region.amazonaws.com"
               },
               "StringLike": {
                   "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::amzn-s3-
demo-bucket/prefix*"
               }
           }
        },
       {
          "Effect": "Allow",
          "Action": [
              "kinesis:DescribeStream",
              "kinesis:GetShardIterator",
              "kinesis:GetRecords",
              "kinesis:ListShards"
          ],
          "Resource": "arn:aws:kinesis:region:account-id:stream/stream-name"
       },
       {
          "Effect": "Allow",
          "Action": [
              "logs:PutLogEvents"
          ],
          "Resource": [
              "arn:aws:logs:region:account-id:log-group:log-group-name:log-
stream:log-stream-name"
          ]
       },
       {
          "Effect": "Allow",
          "Action": [
              "lambda:InvokeFunction",
```

```
                "lambda:GetFunctionConfiguration"
            ],
            "Resource": [
                "arn:aws:lambda:region:account-id:function:function-name:function-
version"
            ]
        },
        {
         "Effect": "Allow",
         "Action": "aoss:APIAccessAll",
         "Resource": "arn:aws:aoss:region:account-id:collection/collection-id"
       }
     ]
}
```

Além da política acima, você também deve configurar o Amazon Data Firehose para ter as permissões mínimas a seguir atribuídas em uma política de acesso a dados:

```
[
   {
      "Rules":[
         {
            "ResourceType":"index",
            "Resource":[
               "index/target-collection/target-index"
            ],
            "Permission":[
               "aoss:WriteDocument",
               "aoss:UpdateIndex",
               "aoss:CreateIndex"
            ]
         }
      ],
      "Principal":[
         "arn:aws:sts::account-id:assumed-role/firehose-delivery-role-name/*"
      ]
   }
]
```

Para obter mais informações sobre como permitir que outros AWS serviços acessem seus AWS recursos, consulte Como criar uma função para delegar permissões a um AWS serviço no Guia do usuário do IAM.

Conceda ao Firehose acesso a um destino OpenSearch sem servidor em uma VPC

Se sua coleção OpenSearch Serverless estiver em uma VPC, certifique-se de conceder ao Amazon Data Firehose as permissões descritas na seção anterior. Além disso, você precisa conceder ao Amazon Data Firehose as seguintes permissões para permitir que ele acesse a VPC da sua OpenSearch coleção Serverless.

- `ec2:DescribeVpcs`
- `ec2:DescribeVpcAttribute`
- `ec2:DescribeSubnets`
- `ec2:DescribeSecurityGroups`
- `ec2:DescribeNetworkInterfaces`
- `ec2:CreateNetworkInterface`
- `ec2:CreateNetworkInterfacePermission`
- `ec2:DeleteNetworkInterface`

> **Important**
>
> Não revogue essas permissões depois de criar o fluxo do Firehose. Se você revogar essas permissões, seu stream do Firehose será degradado ou deixará de fornecer dados ao OpenSearch seu domínio de serviço sempre que o serviço tentar consultar ou atualizar. ENIs

> **Important**
>
> Ao especificar sub-redes para entregar dados ao destino em uma VPC privada, verifique se você tem um número suficiente de endereços IP livres nas sub-redes escolhidas. Se não houver um endereço IP gratuito disponível em uma sub-rede especificada, o Firehose não poderá criar ou ENIs adicionar dados para a entrega de dados na VPC privada, e a entrega será degradada ou falhará.

Ao criar ou atualizar seu stream do Firehose, você especifica um grupo de segurança para o Firehose usar ao enviar dados para sua coleção Serverless. OpenSearch Você pode usar o mesmo grupo de segurança que a coleção OpenSearch Serverless usa ou um diferente. Se você especificar um grupo de segurança diferente, certifique-se de que ele permita tráfego HTTPS de saída para o grupo de segurança da coleção OpenSearch Serverless. Além disso, certifique-se de que o grupo de segurança da coleção OpenSearch Serverless permita tráfego HTTPS do grupo de segurança que você especificou ao configurar seu stream do Firehose. Se você usa o mesmo grupo de segurança para o stream do Firehose e para a coleção OpenSearch Serverless, verifique se a regra de entrada do grupo de segurança permite tráfego HTTPS. Para obter mais informações sobre regras de grupo de segurança, consulte Regras de grupo de segurança na documentação da Amazon VPC.

Concessão ao Firehose de acesso a um destino do Splunk

Quando você está usando um destino Splunk, o Amazon Data Firehose entrega os dados ao endpoint do Coletor de eventos de HTTP (HEC) do Splunk. Ele também faz backup desses dados no bucket do Amazon S3 que você especificar e, opcionalmente, você pode usar uma AWS KMS chave que você possui para a criptografia do lado do servidor do Amazon S3. Se o registro de erros estiver ativado, o Firehose enviará erros de entrega de dados para seus fluxos de CloudWatch registro. Você também pode usar AWS Lambda para transformação de dados.

Se você usa um balanceador de AWS carga, certifique-se de que seja um Classic Load Balancer ou um Application Load Balancer. Além disso, habilite sessões persistentes com a expiração de cookies desabilitada para o Classic Load Balancer, e a expiração definida como máxima (7 dias) para o Application Load Balancer. Para obter informações sobre como fazer isso, consulte Persistência de sessão baseada na duração para Classic Load Balancer ou Application Load Balancer.

É necessário ter um perfil do IAM ao criar um fluxo do Firehose. O Firehose assume essa função do IAM e obtém acesso ao bucket, à chave, ao grupo de CloudWatch registros e aos fluxos especificados.

Use a política de acesso padrão a seguir para permitir que o Amazon Data Firehose acesse seu bucket do S3. Se você não é o proprietário do bucket do S3, adicione `s3:PutObjectAcl` à lista de ações do Amazon S3, o que concede ao proprietário do bucket acesso total aos objetos entregues pelo Amazon Data Firehose. Essa política também concede ao Amazon Data Firehose acesso CloudWatch para registro de erros e transformação de AWS Lambda dados. A política também tem uma instrução que permite o acesso ao Amazon Kinesis Data Streams. Se você não usar o Kinesis Data Streams como fonte de dados, poderá remover essa instrução. O Amazon Data Firehose não usa o IAM para acessar o Splunk. Para acessar o Splunk, ele usa o token HEC.

JSON

```
{
    "Version": "2012-10-17",
    "Statement":
    [
        {
            "Effect": "Allow",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        },
        {
           "Effect": "Allow",
           "Action": [
               "kms:Decrypt",
               "kms:GenerateDataKey"
           ],
           "Resource": [
               "arn:aws:kms:region:account-id:key/key-id"
           ],
           "Condition": {
               "StringEquals": {
                   "kms:ViaService": "s3.region.amazonaws.com"
               },
               "StringLike": {
                   "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::amzn-s3-
demo-bucket/prefix*"
               }
           }
        },
        {
           "Effect": "Allow",
           "Action": [
```

```
                "kinesis:DescribeStream",
                "kinesis:GetShardIterator",
                "kinesis:GetRecords",
                "kinesis:ListShards"
            ],
            "Resource": "arn:aws:kinesis:region:account-id:stream/stream-name"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:region:account-id:log-group:log-group-name:log-
stream:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction",
                "lambda:GetFunctionConfiguration"
            ],
            "Resource": [
                "arn:aws:lambda:region:account-id:function:function-name:function-
version"
            ]
        }
    ]
}
```

Para obter mais informações sobre como permitir que outros AWS serviços acessem seus AWS recursos, consulte Como criar uma função para delegar permissões a um AWS serviço no Guia do usuário do IAM.

Acesso ao Splunk na VPC

Se a plataforma do Splunk estiver em uma VPC, ela será acessível ao público com um endereço IP público. Além disso, conceda ao Amazon Data Firehose acesso à plataforma Splunk desbloqueando os endereços IP do Amazon Data Firehose. Atualmente, o Amazon Data Firehose usa os blocos CIDR a seguir.

Região	Blocos CIDR
Leste dos EUA (Ohio)	18.216.68.160/27, 18.216.170.64/27, 18.216.170.96/27 \
Leste dos EUA (Norte da Virgínia)	34.238.188.128/26, 34.238.188.192/26, 34.238.195.0/26
Oeste dos EUA (Norte da Califórnia)	13.57.180.0/26
Oeste dos EUA (Oregon)	34.216.24.32/27, 34.216.24.192/27, 34.216.24.224/27
AWS GovCloud (Leste dos EUA)	18.253.138.192/26
AWS GovCloud (Oeste dos EUA)	52.61.204.192/26
Ásia-Pacífico (Hong Kong)	18.162.221.64/26
Ásia-Pacífico (Mumbai)	13.232.67.64/26
Ásia-Pacífico (Seul)	13.209.71.0/26
Ásia-Pacífico (Singapura)	13.229.187.128/26
Ásia-Pacífico (Sydney)	13.211.12.0/26
Ásia-Pacífico (Tailândia)	43.208.112.128/26
Ásia-Pacífico (Tóquio)	13.230.21.0/27, 13.230.21.32/27
Canadá (Central)	35.183.92.64/26
Oeste do Canadá (Calgary)	40.176.98.128/26

Região	Blocos CIDR
Europa (Frankfurt)	18.194.95.192/27, 18.194.95.224/27, 18.195.48.0/27
Europa (Irlanda)	34.241.197.32/27, 34.241.197.64/27, 34.241.197.96/27
Europa (Londres)	18.130.91.0/26
Europa (Paris)	35.180.112.0/26
Europa (Espanha)	18.100.194.0/26
Europe (Stockholm)	13.53.191.0/26
Oriente Médio (Bahrein)	15.185.91.64/26
México (Central)	78.12.207.64/26
América do Sul (São Paulo)	18.228.1.192/26
Europa (Milão)	15.161.135.192/26
África (Cidade do Cabo)	13.244.165.128/26
Ásia-Pacífico (Osaka)	13.208.217.0/26
China (Pequim)	52.81.151.64/26
China (Ningxia)	161.189.23.128/26
Ásia-Pacífico (Jacarta)	108.136.221.128/26
Oriente Médio (Emirados Árabes Unidos)	3.28.159.64/26
Israel (Tel Aviv)	51.16.102.64/26
Europa (Zurique)	16.62.183.64/26

Região	Blocos CIDR
Ásia-Pacífico (Hyderabad)	`18.60.192.192/26`
Ásia-Pacífico (Melbourne)	`16.50.161.192/26`
Ásia-Pacífico (Malásia)	`43.216.44.192/26`

Ingestão de logs de fluxo da VPC no Splunk usando o Amazon Data Firehose

Para saber mais sobre como criar uma assinatura de log de fluxo da VPC, publicar no Firehose e enviar os logs de fluxo da VPC para um destino com suporte, consulte Ingestão de logs de fluxo da VPC no Splunk usando o Amazon Data Firehose.

Acesso ao Snowflake ou ao endpoint de HTTP

Não há um subconjunto de intervalos de endereços IP da AWS específico para o Amazon Data Firehose quando o destino é um endpoint de HTTP ou clusters públicos do Snowflake.

Para adicionar o Firehose a uma lista de permissões para clusters públicos do Snowflake ou aos seus endpoints públicos de HTTP ou HTTPS, adicione todos os intervalos de endereços IP da AWS atuais às suas regras de entrada.

> **Note**
>
> As notificações nem sempre são provenientes de endereços IP na mesma AWS região do tópico associado. Você deve incluir o intervalo AWS de endereços IP para todas as regiões.

Concessão ao Firehose de acesso a um destino do Snowflake

Quando você usa o Snowflake como destino, o Firehose entrega dados para uma conta do Snowflake usando o URL da sua conta do Snowflake. Ele também faz backup dos dados de erro no bucket do Amazon Simple Storage Service que você especifica e, opcionalmente, você pode usar

uma AWS Key Management Service chave que você possui para a criptografia do lado do servidor do Amazon S3. Se o registro de erros estiver ativado, o Firehose enviará erros de entrega de dados para seus fluxos de CloudWatch registros.

É necessário ter um perfil do IAM para criar um fluxo do Firehose. O Firehose assume essa função do IAM e obtém acesso ao bucket, à chave, ao grupo e aos fluxos de CloudWatch registros especificados. Use a política de acesso padrão a seguir para permitir que o Firehose acesse o bucket do S3. Se você não for o proprietário do bucket do S3, adicione `s3:PutObjectAcl` à lista de ações do Amazon Simple Storage Service, o que concede ao proprietário do bucket acesso total aos objetos entregues pelo Firehose. Essa política também concede ao Firehose acesso CloudWatch para registro de erros. A política também tem uma instrução que permite o acesso ao Amazon Kinesis Data Streams. Se você não usar o Kinesis Data Streams como fonte de dados, poderá remover essa instrução. O Firehose não usa o IAM para acessar o Snowflake. Para acessar o Snowflake, ele usa o URL e o ID de voz da sua conta do Snowflake no PrivateLink caso de um cluster privado.

JSON

```
{
"Version": "2012-10-17",
    "Statement":
    [
        {
"Effect": "Allow",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        },
        {
"Effect": "Allow",
           "Action": [
               "kms:Decrypt",
```

```
                "kms:GenerateDataKey"
            ],
            "Resource": [
                "arn:aws:kms:region:account-id:key/key-id"
            ],
            "Condition": {
"StringEquals": {
"kms:ViaService": "s3.region.amazonaws.com"
                },
                "StringLike": {
"kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::amzn-s3-demo-bucket/prefix*"
                }
            }
        },
        {
"Effect": "Allow",
            "Action": [
                "kinesis:DescribeStream",
                "kinesis:GetShardIterator",
                "kinesis:GetRecords",
                "kinesis:ListShards"
            ],
            "Resource": "arn:aws:kinesis:region:account-id:stream/stream-name"
        },
        {
"Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:region:account-id:log-group:log-group-name:log-
stream:*"
            ]
        }
    ]
}
```

Para obter mais informações sobre como permitir que outros AWS serviços acessem seus AWS recursos, consulte Como criar uma função para delegar permissões a um AWS serviço no Guia do usuário do IAM.

Acesso ao Snowflake na VPC

Se o cluster do Snowflake tiver um link privado habilitado, o Firehose usará um dos endpoints da VPC a seguir no momento da criação do link privado para entregar dados ao seu cluster privado sem passar pela Internet pública. Para isso, crie regras de rede do Snowflake para permitir a entrada do seguinte `AwsVpceIds` no cluster em que Região da AWS seu cluster está. Para obter mais informações, consulte Criação de regras de rede no Guia do usuário do Snowflake.

IDs de endpoint da VPC a serem usados com base nas regiões em que seu cluster se encontra

Região da AWS	**`VPCE IDs`**
Leste dos EUA (Ohio)	vpce-0d96cafcd96a50aeb vpce-0cec34343d48f537b
Leste dos EUA (Norte da Virgínia)	vpce-0b4d7e8478e141ba8 vpce-0b75cd681fb507352 vpce-01c03e63820ec00d8 vpce-0c2cfc51dc2882422 vpce-06ca862f019e4e056 vpce-020cda0cfa63f8d1c vpce-0b80504a1a783cd70 vpce-0289b9ff0b5259a96 vpce-0d7add8628bd69a12 vpce-02bfb5966cc59b2af vpce-09e707674af878bf2 vpce-049b52e96cc1a2165 vpce-0bb6c7b7a8a86cdbb vpce-03b22d599f51e80f3

Região da AWS	**VPCE IDs**
	vpce-01d60dc60fc106fe1 vpce-0186d20a4b24ecbef vpce-0533906401a36e416 vpce-05111fb13d396710e vpce-0694613f4fbd6f514 vpce-09b21cb25fe4cc4f4 vpce-06029c3550e4d2399 vpce-00961862a21b033da vpce-01620b9ae33273587 vpce-078cf4ec226880ac9 vpce-0d711bf076ce56381 vpce-066b7e13cbfca6f6e vpce-0674541252d9ccc26 vpce-03540b88dedb4b000 vpce-0b1828e79ad394b95 vpce-0dc0e6f001fb1a60d vpce-0d8f82e71a244098a vpce-00e374d9e3f1af5ce vpce-0c1e3d6631ddb442f

Região da AWS	**VPCE IDs**
Oeste dos EUA (Oregon)	vpce-0f60f72da4cd1e4e7 vpce-0c60d21eb8b1669fd vpce-01c4e3e29afdafbef vpce-0cc6bf2a88da139de vpce-0797e08e169e50662 vpce-033cbe480381b5c0e vpce-00debbdd8f9eb10a5 vpce-08ec2f386c809e889 vpce-0856d14310857b545
Europa (Frankfurt)	vpce-068dbb7d71c9460fb vpce-0a7a7f095942d4ec9
Europa (Irlanda)	vpce-06857e59c005a6276 vpce-04390f4f8778b75f2 vpce-011fd2b1f0aa172fd
Ásia-Pacífico (Tóquio)	vpce-06369e5258144e68a vpce-0f2363cdb8926fbe8
Ásia-Pacífico (Singapura)	vpce-049cd46cce7a12d52 vpce-0e8965a1a4bdb8941
Ásia-Pacífico (Seul)	vpce-0aa444d9001e1faa1 vpce-04a49d4dcfd02b884

Região da AWS	**VPCE IDs**
Ásia-Pacífico (Sydney)	vpce-048a60a182c52be63 vpce-03c19949787fd1859
Ásia-Pacífico (Mumbai)	vpce-0d68cb822f6f0db68 vpce-0517d32692ffcbde2
Europa (Londres)	vpce-0fd1874a0ba3b9374 vpce-08091b1a85e206029
América do Sul (São Paulo)	vpce-065169b8144e4f12e vpce-0493699f0e5762d63
Canadá (Central)	vpce-07e6ed81689d5271f vpce-0f53239730541394c
Europa (Paris)	vpce-09419680077e6488a vpce-0ea81ba2c08140c14
Ásia-Pacífico (Osaka)	vpce-0a9f003e6a7e38c05 vpce-02886510b897b1c5a
Europa (Estocolmo)	vpce-0d96410833219025a vpce-060a32f9a75ba969f
Ásia-Pacífico (Jacarta)	vpce-00add4b9a25e5c649 vpce-004ae2de34338a856

Concessão ao Firehose de acesso a um destino de endpoint de HTTP

É possível usar o Amazon Data Firehose para entregar dados a qualquer destino de endpoint de HTTP. O Amazon Data Firehose também faz backup dos dados no bucket do Amazon S3 especificado, e você tem a opção de usar uma chave do AWS KMS que possua para criptografia no lado do servidor do Amazon S3. Se o registro de erros estiver ativado, o Amazon Data Firehose enviará erros de entrega de dados para seus fluxos de CloudWatch log. Você também pode usar AWS Lambda para transformação de dados.

Você precisa ter um perfil do IAM ao criar um fluxo do Firehose. O Amazon Data Firehose assume essa função do IAM e obtém acesso ao bucket, à chave, ao grupo de CloudWatch registros e aos fluxos especificados.

Use a política de acesso a seguir para permitir que o Amazon Data Firehose acesse o bucket do S3 especificado para backup de dados. Se você não é o proprietário do bucket do S3, adicione `s3:PutObjectAcl` à lista de ações do Amazon S3, o que concede ao proprietário do bucket acesso total aos objetos entregues pelo Amazon Data Firehose. Essa política também concede ao Amazon Data Firehose acesso CloudWatch para registro de erros e transformação de AWS Lambda dados. A política também tem uma instrução que permite o acesso ao Amazon Kinesis Data Streams. Se você não usar o Kinesis Data Streams como fonte de dados, poderá remover essa instrução.

> **Important**
>
> O Amazon Data Firehose não usa o IAM para acessar destinos de endpoints HTTP pertencentes a provedores de serviços terceirizados compatíveis, incluindo Datadog, Dynatrace, LogicMonitor MongoDB, New Relic, Splunk ou Sumo Logic. Para acessar um destino de endpoint de HTTP especificado de propriedade de um provedor de serviços terceirizado com suporte, entre em contato com esse provedor de serviços para obter a chave de API ou a chave de acesso necessária para permitir a entrega de dados do Amazon Data Firehose para esse serviço.

JSON

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

```
        {
            "Effect": "Allow",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        },
        {
           "Effect": "Allow",
           "Action": [
               "kms:Decrypt",
               "kms:GenerateDataKey"
           ],
           "Resource": [
               "arn:aws:kms:region:account-id:key/key-id"
           ],
           "Condition": {
               "StringEquals": {
                   "kms:ViaService": "s3.region.amazonaws.com"
               },
               "StringLike": {
                   "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::amzn-s3-
demo-bucket/prefix*"
               }
           }
        },
        {
           "Effect": "Allow",
           "Action": [
               "kinesis:DescribeStream",
               "kinesis:GetShardIterator",
               "kinesis:GetRecords",
               "kinesis:ListShards"
           ],
           "Resource": "arn:aws:kinesis:region:account-id:stream/stream-name"
        },
```

```
        {
           "Effect": "Allow",
           "Action": [
               "logs:PutLogEvents"
           ],
           "Resource": [
               "arn:aws:logs:region:account-id:log-group:log-group-name:log-
stream:*"
           ]
        },
        {
           "Effect": "Allow",
           "Action": [
               "lambda:InvokeFunction",
               "lambda:GetFunctionConfiguration"
           ],
           "Resource": [
               "arn:aws:lambda:region:account-id:function:function-name:function-
version"
           ]
        }
    ]
}
```

Para obter mais informações sobre como permitir que outros AWS serviços acessem seus AWS recursos, consulte Como criar uma função para delegar permissões a um AWS serviço no Guia do usuário do IAM.

> **Important**
>
> Atualmente, o Amazon Data Firehose NÃO oferece suporte à entrega de dados a endpoints de HTTP em uma VPC.

Entrega entre contas do Amazon MSK

Ao criar um stream do Firehose a partir da sua conta do Firehose (por exemplo, Conta B) e sua origem é um cluster MSK em outra AWS conta (Conta A), você deve ter as seguintes configurações em vigor.

Conta A:

1. No console do Amazon MSK, escolha o cluster provisionado e depois escolha Propriedades.
2. Em Configurações de rede, escolha Editar e ative a Conectividade de várias VPCs.
3. Em Configurações de segurança, escolha Editar política do cluster.

   a. Se o cluster ainda não tiver uma política configurada, marque Incluir entidade principal do serviço Firehose e Habilitar a entrega do S3 entre contas do Firehose. Isso AWS Management Console gerará automaticamente uma política com as permissões apropriadas.

   b. Se o cluster já tiver uma política configurada, adicione as seguintes permissões à política existente:

```
{
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::arn:role/mskaasTestDeliveryRole"
      },
      "Action": [
        "kafka:GetBootstrapBrokers",
        "kafka:DescribeCluster",
        "kafka:DescribeClusterV2",
        "kafka-cluster:Connect"
      ],
      "Resource": "arn:aws:kafka:us-east-1:arn:cluster/DO-NOT-TOUCH-mskaas-
provisioned-privateLink/xxxxxxxxx-2f3a-462a-ba09-xxxxxxxxxx-20"  // ARN of the
 cluster
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::arn:role/mskaasTestDeliveryRole"
      },
      "Action": [
        "kafka-cluster:DescribeTopic",
        "kafka-cluster:DescribeTopicDynamicConfiguration",
        "kafka-cluster:ReadData"
      ],
      "Resource": "arn:aws:kafka:us-east-1:arn:topic/DO-NOT-TOUCH-mskaas-
provisioned-privateLink/xxxxxxxxx-2f3a-462a-ba09-xxxxxxxxxx-20/*"//topic of the
 cluster
    },
    {
```

```
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::233450236687:role/mskaasTestDeliveryRole"
      },
      "Action": "kafka-cluster:DescribeGroup",
      "Resource": "arn:aws:kafka:us-east-1:arn:group/DO-NOT-TOUCH-mskaas-
provisioned-privateLink/xxxxxxxxx-2f3a-462a-ba09-xxxxxxxxxx-20/*" //topic of
 the cluster
    },
 }
```

4. Em Entidade principal da AWS , insira o ID da entidade principal da Conta B.
5. Em Tópico, especifique o tópico do Apache Kafka do qual você deseja que o fluxo do Firehose faça a ingestão dos dados. Depois que o fluxo do Firehose for criado, você não poderá mais atualizar esse tópico.
6. Selecione Save changes (Salvar alterações)

Conta B:

1. No console do Firehose, escolha Criar fluxo do Firehose usando a Conta B.
2. Em Fonte, escolha Amazon Managed Streaming for Apache.
3. Em Configurações da fonte, para o cluster do Amazon Managed Streaming for Apache Kafka, insira o ARN do cluster do Amazon MSK na Conta A.
4. Em Tópico, especifique o tópico do Apache Kafka do qual você deseja que o fluxo do Firehose faça a ingestão dos dados. Depois que o fluxo do Firehose for criado, você não poderá mais atualizar esse tópico.
5. Em Nome do fluxo de entrega, especifique o nome do seu fluxo do Firehose.

Na Conta B, ao criar seu stream do Firehose, você deve ter uma função do IAM (criada por padrão ao usar o AWS Management Console) que conceda ao stream do Firehose acesso de “leitura” ao cluster Amazon MSK entre contas para o tópico configurado.

Veja a seguir o que é configurado pelo AWS Management Console:

```
{
    "Sid": "",
    "Effect": "Allow",
    "Action": [
```

```
        "kafka:GetBootstrapBrokers",
        "kafka:DescribeCluster",
        "kafka:DescribeClusterV2",
        "kafka-cluster:Connect"
        ],
    "Resource": "arn:aws:kafka:us-east-1:arn:cluster/DO-NOT-TOUCH-mskaas-provisioned-
privateLink/xxxxxxxxx-2f3a-462a-ba09-xxxxxxxxxx-20/*" //topic of the cluster
    },
    {
    "Sid": "",
    "Effect": "Allow",
    "Action": [
        "kafka-cluster:DescribeTopic",
        "kafka-cluster:DescribeTopicDynamicConfiguration",
        "kafka-cluster:ReadData"
    ],
    "Resource": "arn:aws:kafka:us-east-1:arn:topic/DO-NOT-TOUCH-mskaas-provisioned-
privateLink/xxxxxxxxx-2f3a-462a-ba09-xxxxxxxxxx-20/mskaas_test_topic" //topic of the
 cluster
    },
    {
    "Sid": "",
    "Effect": "Allow",
    "Action": [
        "kafka-cluster:DescribeGroup"
    ],
    "Resource": "arn:aws:kafka:us-east-1:arn:group/DO-NOT-TOUCH-mskaas-provisioned-
privateLink/xxxxxxxxx-2f3a-462a-ba09-xxxxxxxxxx-20/*" //topic of the cluster
    },
 }
```

Em seguida, é possível concluir a etapa opcional de configuração da transformação de registros e da conversão de formato de registros. Para obter mais informações, consulte (Opcional) Configuração de transformação de registro e conversão de formato.

Entrega entre contas a um destino do Amazon S3

Você pode usar o AWS CLI ou o Amazon Data Firehose APIs para criar um stream do Firehose em uma AWS conta com um destino Amazon S3 em uma conta diferente. O procedimento a seguir mostra um exemplo de configuração de fluxo do Firehose pertencente à conta para entregar os dados a um bucket do Amazon S3 pertencente à conta B.

1. Crie um perfil do IAM na conta A usando as etapas descritas em Concessão ao Firehose de acesso a um destino do Amazon S3.

   > **Note**
   >
   > O bucket do Amazon S3 especificado na política de acesso padrão pertence à conta B neste caso. Lembre-se de adicionar `s3:PutObjectAcl` à lista de ações do Amazon S3 na política de acesso padrão, o que concederá à conta B acesso total aos objetos entregues pelo Amazon Data Firehose. Essa permissão é necessária para a entrega entre contas. O Amazon Data Firehose define o cabeçalho x-amz-acl "" na solicitação como ""bucket-owner-full-control.

2. Para permitir o acesso no perfil do IAM criado anteriormente, crie uma política de bucket do S3 na conta B. O código a seguir é um exemplo de política de bucket. Para obter mais informações, consulte Usar políticas de buckets e do usuário.

   JSON

```
{

    "Version": "2012-10-17",
    "Id": "PolicyID",
    "Statement": [
        {
            "Sid": "StmtID",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::accountA-id:role/iam-role-name"
            },
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
```

```
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

3. Crie um fluxo do Firehose na conta A usando o perfil do IAM criado na etapa 1.

Entrega entre contas para um destino OpenSearch de serviço

Você pode usar o AWS CLI ou o Amazon Data Firehose APIs para criar um stream do Firehose em uma AWS conta com um destino de OpenSearch serviço em outra conta. O procedimento a seguir mostra um exemplo de como você pode criar um stream do Firehose na conta A e configurá-lo para entregar dados a um destino de OpenSearch serviço pertencente à conta B.

1. Criar um perfil do IAM na conta A usando as etapas descritas em the section called “Conceder ao Firehose acesso a um destino de serviço público OpenSearch ”.
2. Para permitir o acesso da função do IAM que você criou na etapa anterior, crie uma política de OpenSearch serviço na conta B. O seguinte JSON é um exemplo.

   JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::Account-A-ID:role/firehose_delivery_role "
      },
      "Action": "es:ESHttpGet",
      "Resource": [
        "arn:aws:es:us-east-1:Account-B-ID:domain/cross-account-cluster/_all/
_settings",
        "arn:aws:es:us-east-1:Account-B-ID:domain/cross-account-cluster/
_cluster/stats",
        "arn:aws:es:us-east-1:Account-B-ID:domain/cross-account-cluster/
roletest*/_mapping/roletest",
```

```
        "arn:aws:es:us-east-1:Account-B-ID:domain/cross-account-cluster/
_nodes",
        "arn:aws:es:us-east-1:Account-B-ID:domain/cross-account-cluster/
_nodes/stats",
        "arn:aws:es:us-east-1:Account-B-ID:domain/cross-account-cluster/
_nodes/*/stats",
        "arn:aws:es:us-east-1:Account-B-ID:domain/cross-account-cluster/
_stats",
        "arn:aws:es:us-east-1:Account-B-ID:domain/cross-account-cluster/
roletest*/_stats",
        "arn:aws:es:us-east-1:Account-B-ID:domain/cross-account-cluster/"
      ]
    }
  ]
}
```

3. Crie um fluxo do Firehose na conta A usando o perfil do IAM criado na etapa 1. Ao criar o stream do Firehose, use o AWS CLI ou o Amazon Data Firehose APIs e especifique o `ClusterEndpoint` campo em vez de Service. `DomainARN` OpenSearch

> **Note**
>
> Para criar um stream do Firehose em uma AWS conta com um destino de OpenSearch serviço em uma conta diferente, você deve usar o AWS CLI ou o Amazon Data Firehose. APIs Você não pode usar o AWS Management Console para criar esse tipo de configuração entre contas.

Uso de tags para controlar o acesso

É possível usar o elemento opcional `Condition` (ou bloco de `Condition`) em uma política do IAM para ajustar o acesso às operações do Amazon Data Firehose com base nas chaves e valores das tags. As subseções a seguir descrevem como fazer isso para as diferentes operações do Amazon Data Firehose. Para saber mais sobre o uso do elemento `Condition` e as operações que podem ser usadas com ele, consulte Elementos de política JSON do IAM: condição.

CreateDeliveryStream

Para a operação `CreateDeliveryStream`, use a chave de condição `aws:RequestTag`. No exemplo a seguir, `MyKey` e `MyValue` representam a chave e o valor correspondente de uma tag. Para obter mais informações, consulte Noções básicas sobre tags.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "firehose:CreateDeliveryStream",
            "firehose:TagDeliveryStream"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/MyKey": "MyValue"
            }
        }
    }]
}
```

TagDeliveryStream

Para a operação `TagDeliveryStream`, use a chave de condição `aws:TagKeys`. No exemplo a seguir, `MyKey` é um exemplo de chave de tag.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "firehose:TagDeliveryStream",
            "Resource": "*",
            "Condition": {
```

```
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "MyKey"
                }
            }
        }
    ]
}
```

UntagDeliveryStream

Para a operação `UntagDeliveryStream`, use a chave de condição `aws:TagKeys`. No exemplo a seguir, `MyKey` é um exemplo de chave de tag.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "firehose:UntagDeliveryStream",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "MyKey"
                }
            }
        }
    ]
}
```

ListDeliveryStreams

Não é possível usar controle de acesso com base em tags com `ListDeliveryStreams`.

Outras operações

Para todas as operações do Firehose, exceto `CreateDeliveryStream`, `TagDeliveryStream`, `UntagDeliveryStream` e `ListDeliveryStreams`, use a chave de condição `aws:RequestTag`. No exemplo a seguir, `MyKey` e `MyValue` representam a chave e o valor correspondente de uma tag.

`ListDeliveryStreams`, use a chave de condição `firehose:ResourceTag` para controlar o acesso com base nas tags desse fluxo do Firehose.

No exemplo a seguir, `MyKey` e `MyValue` representam a chave e o valor correspondente de uma tag. A política se aplicaria somente aos fluxos do Data Firehose com uma tag de nome `MyKey` com um valor de `MyValue`. Para obter mais informações sobre como controlar o acesso com base em tags de recursos, consulte Como controlar o acesso a AWS recursos usando tags no Guia do usuário do IAM.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
            "Effect": "Deny",
            "Action": "firehose:DescribeDeliveryStream",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                      "firehose:ResourceTag/MyKey": "MyValue"
                  }
            }
        }
    ]
}
```

Autentique-se com o AWS Secrets Manager Amazon Data Firehose

O Amazon Data Firehose se integra AWS Secrets Manager para fornecer acesso seguro aos seus segredos e automatizar a rotação de credenciais. Essa integração permite que o Firehose recupere um segredo do Secrets Manager no runtime para se conectar aos destinos de streaming mencionados anteriormente e entregar seus fluxos de dados. Com isso, seus segredos não são visíveis em texto simples durante o fluxo de trabalho de criação do stream, AWS Management Console nem nos parâmetros da API. Ele fornece uma prática segura para gerenciar seus segredos e dispensa você de atividades complexas de gerenciamento de credenciais, como a configuração de funções do Lambda personalizadas para gerenciar as alternâncias de senhas.

Para obter mais informações, consulte o Guia do usuário do AWS Secrets Manager.

Tópicos

Noções básicas sobre segredos

Um segredo pode ser uma senha, um conjunto de credenciais, como nome de usuário e senha, um OAuth token ou outras informações secretas que você armazena de forma criptografada no Secrets Manager.

Para cada destino, você deve especificar o par de valores-chave do segredo no formato JSON correto, conforme mostrado na seção a seguir. O Amazon Data Firehose não conseguirá se conectar ao seu destino se o seu segredo não tiver o formato JSON correto de acordo com o destino.

Formato secreto para bancos de dados como MySQL e PostgreSQL

```
{
    "username":  "<username>",
    "password":  "<password>"
}
```

Formato de segredo para o cluster provisionado do Amazon Redshift e o grupo de trabalho Amazon Redshift sem servidor

```
{
    "username":  "<username>",
    "password":  "<password>"
}
```

Formato de segredo para o Splunk

```
{
    "hec_token":  "<hec token>"
```

```
}
```

Formato de segredo para o Snowflake

```
{
    "user":  "<user>",
    "private_key":  "<private_key>", // without the begin and end private key, remove
 all spaces and newlines
    "key_passphrase":  "<passphrase>" // optional
}
```

Formato secreto para endpoint HTTP, Coralogix, Datadog, Dynatrace, Elastic, Honeycomb, Logz.io, MongoDB Cloud e New Relic LogicMonitor

```
{
    "api_key":  "<apikey>"
}
```

Criar um segredo

Para criar um segredo, siga as etapas em Criar um AWS Secrets Manager segredo no Guia do AWS Secrets Manager usuário.

Uso do segredo

Recomendamos que você use AWS Secrets Manager para armazenar suas credenciais ou chaves para se conectar a destinos de streaming, como Amazon Redshift, endpoint HTTP, Snowflake, Splunk, Coralogix, Datadog, Dynatrace, Elastic, Honeycomb, Logz.io, MongoDB Cloud e New Relic. LogicMonitor

É possível configurar a autenticação com o Secrets Manager para esses destinos por meio do Console de Gerenciamento da AWS no momento da criação do fluxo do Firehose. Para obter mais informações, consulte Definição de configurações do destino. Como alternativa, você também pode usar as operações CreateDeliveryStreame da UpdateDestinationAPI para configurar a autenticação com o Secrets Manager.

O Firehose armazena os segredos em cache com uma criptografia e os usa em todas as conexões com os destinos. Ele atualiza o cache a cada 10 minutos para garantir que as credenciais mais recentes sejam usadas.

É possível optar por desativar a capacidade de recuperar segredos do Secrets Manager a qualquer momento durante o ciclo de vida do fluxo. Se você não quiser usar o Secrets Manager para recuperar segredos, você pode usar a chave de API username/password ou em vez disso.

> **Note**
>
> Embora não haja custo adicional para esse atributo no Firehose, você será cobrado pelo acesso e pela manutenção do Secrets Manager. Para obter mais informações, consulte a página de definição de preços do AWS Secrets Manager.

Concessão de acesso ao Firehose para recuperar o segredo

Para que o Firehose recupere um segredo AWS Secrets Manager, você deve fornecer ao Firehose as permissões necessárias para acessar o segredo e a chave que criptografa seu segredo.

Ao usar AWS Secrets Manager para armazenar e recuperar segredos, há algumas opções de configuração diferentes, dependendo de onde o segredo está armazenado e de como é criptografado.

- Se o segredo estiver armazenado na mesma AWS conta da sua função do IAM e estiver criptografado com a chave AWS gerenciada padrão (`aws/secretsmanager`), a função do IAM que a Firehose presume só precisará de `secretsmanager:GetSecretValue` permissão sobre o segredo.

```
// secret role policy
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "Secret ARN"
        }
    ]
}
```

  Para obter mais informações sobre políticas do IAM, consulte Exemplos de políticas de permissão para o AWS Secrets Manager.

- Se o segredo estiver armazenado na mesma conta do perfil, mas criptografado com uma chave gerenciada pelo cliente (CMK), o perfil precisará de ambas as permissões `secretsmanager:GetSecretValue` e `kms:Decrypt`. A política da CMK também precisa permitir que o perfil do IAM seja execute `kms:Decrypt`.

  JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "Secret ARN"
        },
        {
            "Effect": "Allow",
            "Action": "kms:Decrypt",
            "Resource": "KMSKeyARN"
        }
    ]
}
```

- Se o segredo estiver armazenado em uma AWS conta diferente da sua função e for criptografado com a chave AWS gerenciada padrão, essa configuração não será possível, pois o Secrets Manager não permite acesso entre contas quando o segredo é criptografado com a chave AWS gerenciada.
- Se o segredo for armazenado em uma conta diferente e criptografado com uma CMK, o perfil do IAM precisará da permissão `secretsmanager:GetSecretValue` sobre o segredo e da permissão `kms:Decrypt` na CMK. A política de recursos do segredo e a política de CMK na outra conta também precisam permitir que o perfil do IAM tenha as permissões necessárias. Para obter mais informações, consulte Acesso entre contas.

Alternância do segredo

A alternância é quando você atualiza periodicamente um segredo. Você pode configurar AWS Secrets Manager para alternar automaticamente o segredo para você em uma programação especificada por você. Dessa forma, é possível substituir segredos de longo prazo por segredos de curto prazo. Isso ajuda a reduzir o risco de comprometimento. Para obter mais informações, consulte Rotacionar AWS Secrets Manager segredos no Guia do AWS Secrets Manager usuário.

Gerenciamento de perfis do IAM por meio do console do Amazon Data Firehose

O Amazon Data Firehose é um serviço totalmente gerenciado para entrega de streaming de dados em tempo real a destinos. Também é possível configurar o Firehose para transformar e converter o formato dos seus dados antes de entregá-los. Para usar esses atributos, primeiro você deve fornecer perfis do IAM para conceder permissões ao Firehose ao criar ou editar um fluxo do Firehose. O Firehose usa esse perfil do IAM para todas as permissões necessárias ao fluxo do Firehose.

Por exemplo, considere um cenário em que você cria um stream do Firehose que entrega dados para o Amazon S3, e esse stream do Firehose tem registros de origem do Transform com o recurso ativado. AWS Lambda Nesse caso, você deve fornecer perfis do IAM para conceder permissões ao Firehose para acessar o bucket do S3 e invocar a função do Lambda, conforme mostrado a seguir.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "lambdaProcessing",
        "Effect": "Allow",
        "Action": ["lambda:InvokeFunction", "lambda:GetFunctionConfiguration"],
        "Resource": "arn:aws:lambda:us-east-1:<account id>:function:<lambda
 function name>:<lambda function version>"
    }, {
        "Sid": "s3Permissions",
        "Effect": "Allow",
        "Action": ["s3:AbortMultipartUpload", "s3:GetBucketLocation",
 "s3:GetObject", "s3:ListBucket", "s3:ListBucketMultipartUploads",
 "s3:PutObject"],
        "Resource": ["arn:aws:s3:::<bucket name>", "arn:aws:s3:::<bucket name>/
*"]
    }]
}
```

O console do Firehose permite que você escolha como deseja fornecer essas funções. É possível escolher uma das opções a seguir.

- Escolha um perfil do IAM existente

- Para criar um novo perfil do IAM no console

Escolha um perfil do IAM existente

É possível escolher dentre os perfis do IAM existentes. Com essa opção, certifique-se de que o perfil do IAM escolhido tenha uma política de confiança adequada e as permissões necessárias para sua fonte e destino. Para obter mais informações, consulte Controle de acesso com o Amazon Data Firehose.

Para criar um novo perfil do IAM no console

Como alternativa, é possível usar o console do Firehose para criar um novo perfil em seu nome.

Quando o Firehose cria um perfil do IAM em seu nome, o perfil inclui automaticamente todas as políticas de permissão e confiança que concedem as permissões necessárias com base na configuração do fluxo do Firehose.

Por exemplo, se você não habilitou o atributo Transformar registros da fonte com o AWS Lambda, o console gerará a declaração a seguir na política de permissão.

```
{
  "Sid": "lambdaProcessing",
  "Effect": "Allow",
   "Action": [
     "lambda:InvokeFunction",
     "lambda:GetFunctionConfiguration"
   ],
   "Resource": "arn:aws:lambda:us-east-1:<account id>:function:
%FIREHOSE_POLICY_TEMPLATE_PLACEHOLDER%"
}
```

> **Note**
>
> É seguro ignorar as declarações de política que contenham `%FIREHOSE_POLICY_TEMPLATE_PLACEHOLDER%`, pois elas não concedem permissões sobre nenhum recurso.

O console cria e edita fluxos de trabalho de fluxo do Firehose também cria uma política de confiança e a anexa ao perfil do IAM. Essa política de confiança permite que o Firehose assuma o perfil do IAM. Veja a seguir um exemplo de política de confiança.

JSON

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

⚠ Important

- Você deve evitar usar o mesmo perfil do IAM gerenciado pelo console para vários fluxos do Firehose. Caso contrário, o perfil do IAM pode se tornar excessivamente permissivo ou resultar em erros.
- Para usar declarações de política diferentes em uma política de permissão de um perfil do IAM gerenciado pelo console, é possível criar seu próprio perfil do IAM e copiar as declarações de política para uma política de permissão anexada ao novo perfil. Para anexar o perfil ao fluxo do Firehose, selecione a opção Escolher perfil do IAM existente no Acesso ao serviço.
- O console gerencia qualquer perfil do IAM que contenha a string service-role em seu ARN. Ao escolher a opção de perfil do IAM existente, certifique-se de selecionar um perfil do IAM sem a string service-role em seu ARN, para que o console não faça nenhuma alteração nela.

Etapas para a criação de um perfil do IAM a partir do console

1. Abra o console Firehose em. https://console.aws.amazon.com/firehose/

2. Escolha Criar fluxo do Firehose.
3. Escolha uma fonte e um destino. Para obter mais informações, consulte Tutorial: Criação de um fluxo do Firehose a partir do console.
4. Escolha as configurações do destino. Para obter mais informações, consulte Definição de configurações do destino.
5. Em Configurações avançadas, para Acesso ao serviço, escolha Criar ou atualizar perfil do IAM.

> **Note**
>
> Essa é uma opção padrão. Para usar um perfil existente, selecione a opção Escolher perfil do IAM existente. O console do Firehose não fará nenhuma alteração em seu próprio perfil.

6. Escolha Criar fluxo do Firehose.

Edição de perfil do IAM a partir do console

Quando você edita um fluxo do Firehose, o Firehose atualiza a política de permissão correspondente de acordo com as alterações de configuração e permissão.

Por exemplo, quando você edita o fluxo do Firehose e ativa o atributo Transformar registros da fonte com o AWS Lambda usando a versão mais recente da função do Lambda como `exampleLambdaFunction`, você obtém a declaração de política a seguir na política de permissão.

```
{
  "Sid": "lambdaProcessing",
  "Effect": "Allow",
  "Action": [
    "lambda:InvokeFunction",
    "lambda:GetFunctionConfiguration"
  ],
  "Resource": "arn:aws:lambda:us-east-1:<account id>:function:exampleLambdaFunction:
$LATEST"
}
```

> **Important**
>
> Um perfil do IAM gerenciado pelo console foi projetado para ser autônomo. Não recomendamos modificar a política de permissão ou a política de confiança fora do console.

Etapas para a edição de um perfil do IAM a partir do console

1. Abra o console Firehose em. https://console.aws.amazon.com/firehose/
2. Escolha Fluxos do Firehose e escolha o nome de um fluxo do Firehose que você deseja atualizar.
3. Na guia Configuração, na seção Acesso ao servidor, escolha Editar.
4. Atualize a opção de perfil do IAM.

   > **Note**
   >
   > Por padrão, o console sempre atualiza um perfil do IAM com o padrão service-role em seu ARN. Ao escolher a opção de perfil do IAM existente, certifique-se de selecionar um perfil do IAM sem a string service-role em seu ARN, para que o console não faça nenhuma alteração nela.

5. Escolha Salvar alterações.

Noções básicas de conformidade para o Amazon Data Firehose

Auditores terceirizados avaliam a segurança e a conformidade do Amazon Data Firehose como parte de AWS vários programas de conformidade. Isso inclui SOC, PCI, FedRAMP, HIPAA e outros.

Para obter uma lista de AWS serviços no escopo de programas de conformidade específicos, consulte AWS Serviços no escopo por programa de conformidade. Para obter informações gerais, consulte Programas de conformidade da AWS.

Você pode baixar relatórios de auditoria de terceiros usando AWS Artifact. Para obter mais informações, consulte Baixando relatórios no AWS Artifact.

Sua responsabilidade em termos de conformidade ao usar o Data Firehose é determinada pelo grau de confidencialidade dos dados, pelos objetivos de conformidade da empresa e pelos regulamentos

e leis aplicáveis. Se o uso do Data Firehose estiver sujeito à conformidade com padrões como HIPAA, PCI ou FedRAMP, fornece recursos para ajudar a: AWS

- Guias de início rápido sobre segurança e conformidade — Esses guias de implantação discutem considerações arquitetônicas e fornecem etapas para a implantação de ambientes básicos focados em segurança e conformidade em. 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 conformidade — Essa coleção de pastas de trabalho e guias pode ser aplicada ao seu setor e local.
- AWS Config— Esse AWS serviço avalia se 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 Data Firehose

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 aplicações e bancos de dados que executam o failover automaticamente entre as zonas de disponibilidade 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.

Além da infraestrutura AWS global, o Data Firehose oferece vários recursos para ajudar a suportar suas necessidades de resiliência e backup de dados.

Recuperação de desastres

O Amazon Data Firehose é executado em um modo de tecnologia sem servidor e trata das degradações do host, da disponibilidade das zonas de disponibilidade e de outros problemas

relacionados à infraestrutura, fazendo uma migração automática. Quando isso acontece, o Amazon Data Firehose garante que o fluxo do Firehose seja migrado sem perder nenhum dado.

Noções básicas de segurança de infraestrutura no Amazon Data Firehose

Como um serviço gerenciado, o Amazon Data Firehose é protegido pela segurança de rede AWS global. Para obter informações sobre serviços AWS de segurança e como AWS proteger a infraestrutura, consulte AWS Cloud Security. Para projetar seu AWS ambiente usando as melhores práticas de segurança de infraestrutura, consulte Proteção de infraestrutura no Security Pillar AWS Well-Architected Framework.

Você usa chamadas de API AWS publicadas para acessar o Firehose pela rede. Os clientes devem oferecer compatibilidade com:

- Transport Layer Security (TLS). Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Conjuntos de criptografia com perfect forward secrecy (PFS) como DHE (Ephemeral Diffie-Hellman) ou ECDHE (Ephemeral Elliptic Curve Diffie-Hellman). A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta 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 é possível usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

> **Note**
>
> Para solicitações de HTTPS enviadas, o Amazon Data Firehose usa uma biblioteca de HTTP que seleciona automaticamente a versão mais alta do protocolo TLS com suporte no destino.

Usando o Amazon Data Firehose com AWS PrivateLink

Você pode usar uma interface VPC endpoint (AWS PrivateLink) para acessar o Amazon Data Firehose de sua VPC sem precisar de um Internet Gateway ou NAT Gateway. Os endpoints VPC de interface não exigem um gateway de internet, dispositivo NAT, conexão VPN ou conexão. AWS Direct Connect Os endpoints de VPC de interface são alimentados por AWS PrivateLink uma AWS

tecnologia que permite a comunicação privada entre AWS serviços usando uma interface de rede elástica com privacidade em IPs sua Amazon VPC. Para obter mais informações, consulte o Amazon Virtual Private Cloud.

Usando a interface VPC endpoints ()AWS PrivateLink para Firehose

Para iniciar, crie um endpoint da VPC de interface para que o tráfego do Amazon Data Firehose vindo dos recursos da Amazon VPC comece a fluir através do endpoint da VPC de interface. Quando você cria um endpoint, pode anexar a ele uma política de endpoint que controla o acesso ao Amazon Data Firehose. Para saber mais sobre o uso de políticas para controlar o acesso de um endpoint da VPC ao Amazon Data Firehose, consulte Controle de acesso aos serviços com endpoints da VPC.

O exemplo a seguir mostra como você pode configurar uma AWS Lambda função em uma VPC e criar um VPC endpoint para permitir que a função se comunique com segurança com o serviço Amazon Data Firehose. Neste exemplo, você usa uma política que permite que a função do Lambda liste os fluxos do Firehose na região atual, mas não os descreva.

Criar um VPC endpoint

1. Faça login no AWS Management Console e abra o console da Amazon VPC em. https://console.aws.amazon.com/vpc/
2. No painel da VPC, selecione Endpoints.
3. Escolha Criar endpoint.
4. Na lista de nomes de serviço, escolha `com.amazonaws.`*`your_region`*`.kinesis-firehose`.
5. Escolha a VPC e uma ou mais sub-redes nas quais criar o endpoint.
6. Escolha um ou mais grupos de segurança para associar ao endpoint.
7. Para Policy (Política), selecione Custom (Personalizar) e cole a seguinte política:

```
{
    "Statement": [
        {
            "Sid": "Allow-only-specific-PrivateAPIs",
            "Principal": "*",
            "Action": [
                "firehose:ListDeliveryStreams"
            ],
            "Effect": "Allow",
            "Resource": [
                "*"
```

```
            ]
        },
        {
            "Sid": "Allow-only-specific-PrivateAPIs",
            "Principal": "*",
            "Action": [
                "firehose:DescribeDeliveryStream"
            ],
            "Effect": "Deny",
            "Resource": [
                "*"
            ]
        }
    ]
}
```

8. Escolha Criar endpoint.

Criar um perfil do IAM para ser usado com a função do Lambda

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Perfis e Criar perfil.
3. Em Selecionar o tipo de entidade confiável, deixe a seleção padrão Serviço da AWS .
4. Em Escolha o serviço que usará esse perfil, escolha Lambda.
5. Escolha Next: Permissions (Próximo: Permissões).
6. Na lista de políticas, procure e adicione as duas políticas chamadas `AWS LambdaVPCAccessExecutionRole` e `AmazonDataFirehoseReadOnlyAccess`.

> **Important**
>
> Este é um exemplo. É possível precisar de políticas mais rigorosas para o ambiente de produção.

7. Escolha Próximo: tags. Para a finalidade deste exercício, não é necessário adicionar tags. Escolha Próximo: revisar.
8. Insira um nome para o perfil e escolha Criar perfil.

Criar uma função do Lambda dentro da VPC

1. Abra o AWS Lambda console em https://console.aws.amazon.com/lambda/.
2. Escolha Create function (Criar função).
3. Escolha Author from scratch (Criar do zero).
4. Insira um nome para a função e defina Runtime como `Python 3.9` ou posterior.
5. Em Permissions (Permissões), expanda Choose or create an execution role (Escolher ou criar uma função de execução).
6. Na lista Execution role (Função de execução), selecione Use an existing role (Usar uma função existente).
7. Na lista Existing role (Função existente), selecione a função criada acima.
8. Escolha a opção Criar função.
9. Em Function code (Código da função), cole o código a seguir.

```
   import json
   import boto3
   import os
   from botocore.exceptions import ClientError

   def lambda_handler(event, context):
       REGION = os.environ['AWS_REGION']
       client = boto3.client(
           'firehose',
           REGION

       )
       print("Calling list_delivery_streams with ListDeliveryStreams allowed
policy.")
       delivery_stream_request = client.list_delivery_streams()
       print("Successfully returned list_delivery_streams request %s." % (
           delivery_stream_request
       ))
       describe_access_denied = False
       try:
           print("Calling describe_delivery_stream with DescribeDeliveryStream
denied policy.")
           delivery_stream_info =
client.describe_delivery_stream(DeliveryStreamName='test-describe-denied')
       except ClientError as e:
```

```
            error_code = e.response['Error']['Code']
            print ("Caught %s." % (error_code))
            if error_code == 'AccessDeniedException':
                describe_access_denied = True

        if not describe_access_denied:
            raise
        else:
            print("Access denied test succeeded.")
```

10. Em Basic settings (Configurações básicas), defina o tempo limite como 1 minuto.
11. Em Network (Rede), selecione a VPC onde você criou o endpoint acima e selecione as sub-redes e o grupo de segurança que foram associados ao endpoint quando ele foi criado.
12. Próximo ao alto da página, selecione Salvar.
13. Escolha Testar.
14. Insira o nome de um evento e escolha Criar.
15. Escolha Test (Testar) novamente. Isso faz com que a função seja executada. Depois que o resultado da execução for exibido, expanda Details (Detalhes) e compare a saída do log com o código da função. Resultados com êxito mostram uma lista de fluxos do Firehose na região, bem como a saída a seguir:

    ```
    Calling describe_delivery_stream.

    AccessDeniedException

    Access denied test succeeded.
    ```

Suportado Regiões da AWS

No momento, há suporte para os endpoints da VPC nas regiões a seguir.

- Leste dos EUA (Ohio)
- Leste dos EUA (Norte da Virgínia)
- Oeste dos EUA (Norte da Califórnia)
- Oeste dos EUA (Oregon)
- Ásia-Pacífico (Mumbai)
- Ásia-Pacífico (Seul)

- Ásia-Pacífico (Singapura)
- Ásia-Pacífico (Sydney)
- Ásia-Pacífico (Tailândia)
- Ásia-Pacífico (Tóquio)
- Ásia-Pacífico (Hong Kong)
- Canadá (Central)
- Oeste do Canadá (Calgary)
- China (Pequim)
- China (Ningxia)
- Europa (Frankfurt)
- Europa (Irlanda)
- Europa (Londres)
- Europa (Paris)
- México (Central)
- América do Sul (São Paulo)
- AWS GovCloud (Leste dos EUA)
- AWS GovCloud (Oeste dos EUA)
- Europa (Espanha)
- Oriente Médio (Emirados Árabes Unidos)
- Ásia-Pacífico (Jacarta)
- Ásia-Pacífico (Osaka)
- Israel (Tel Aviv)
- Ásia-Pacífico (Malásia)

Implementação de práticas recomendadas de segurança para o Amazon Data Firehose

O Amazon Data Firehose fornece uma série de atributos de segurança a serem considerados quando você desenvolver e implementar suas próprias diretivas de segurança. As práticas recomendadas a seguir são diretrizes gerais e não representam uma solução completa de

segurança. Como essas práticas recomendadas podem não ser adequadas ou suficientes no seu ambiente, trate-as como considerações úteis em vez de requisitos.

Implemente o acesso de privilégio mínimo

Ao conceder permissões, você decide quem recebe quais permissões para quais recursos do Amazon Data Firehose. Habilite ações específicas que quer permitir nesses recursos. Portanto, é necessário conceder somente as permissões necessárias para executar uma tarefa. A implementação do privilégio de acesso mínimo é fundamental para reduzir o risco de segurança e o impacto que pode resultar de erros ou usuários mal-intencionados.

Usar funções do IAM

Aplicações de produtores e clientes precisam de credenciais válidas para acessar os fluxos do Firehose, e seu fluxo do Firehose precisa de credenciais válidas para acessar os destinos. Você não deve armazenar AWS credenciais diretamente em um aplicativo cliente ou em um bucket do Amazon S3. Essas são credenciais de longo prazo que não são automaticamente alternadas e podem ter um impacto comercial significativo se forem comprometidas.

Em vez disso, você deve usar um perfil do IAM para gerenciar credenciais temporárias que suas aplicações de clientes e produtores usarão para acessar fluxos do Firehose. Ao usar uma função, não é necessário usar credenciais de longo prazo (como um nome de usuário e uma senha ou chaves de acesso) para acessar outros recursos.

Para obter mais informações, consulte os seguintes tópicos no Manual do usuário do IAM:

- Perfis do IAM
- Cenários comuns para perfis: usuários, aplicações e serviços

Implementação da criptografia do lado do servidor em recursos dependentes

É possível criptografar dados em repouso e dados em trânsito no Amazon Data Firehose. Para obter mais informações, consulte Proteção de dados no Amazon Data Firehose.

Use CloudTrail para monitorar chamadas de API

O Amazon Data Firehose é integrado com AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, função ou AWS serviço no Amazon Data Firehose.

Usando as informações coletadas por CloudTrail, você pode determinar a solicitação que foi feita ao Amazon Data Firehose, o endereço IP a partir do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais.

Para obter mais informações, consulte the section called “Registro em log de chamadas de API do Firehose”.

Monitoramento do Amazon Data Firehose

É possível monitorar o Amazon Data Firehose usando os atributos a seguir:

Tópicos

Implementação de práticas recomendadas com CloudWatch alarmes

Adicione CloudWatch alarmes para quando as métricas a seguir excederem o limite de armazenamento em buffer (máximo de 15 minutos).

- `DeliveryToS3.DataFreshness`
- `DeliveryToIceberg.DataFreshness`
- `DeliveryToSplunk.DataFreshness`
- `DeliveryToAmazonOpenSearchService.DataFreshness`
- `DeliveryToAmazonOpenSearchServerless.DataFreshness`
- `DeliveryToHttpEndpoint.DataFreshness`

Além disso, crie alarmes com base nas expressões matemáticas de métricas a seguir.

- `IncomingBytes (Sum per 5 Minutes) / 300` se aproxima de uma porcentagem de `BytesPerSecondLimit`.
- `IncomingRecords (Sum per 5 Minutes) / 300` se aproxima de uma porcentagem de `RecordsPerSecondLimit`.

- `IncomingPutRequests (Sum per 5 Minutes) / 300` se aproxima de uma porcentagem de `PutRequestsPerSecondLimit`.

Outra métrica para a qual recomendamos um alarme é `ThrottledRecords`.

Para obter informações sobre solução de problemas quando os alarmes vá para o `ALARM` estado, consulte Solucionar erros.

Monitoramento do Amazon Data Firehose com métricas CloudWatch

> **Important**
>
> Certifique-se de ativar os alarmes em todas as CloudWatch métricas que pertencem ao destino para identificar os erros em tempo hábil.

O Amazon Data Firehose é integrado com CloudWatch as métricas da Amazon para que você possa coletar, visualizar e analisar CloudWatch métricas para os fluxos do Firehose. Por exemplo, é possível monitorar as métricas `IncomingBytes` e `IncomingRecords` para rastrear os dados dos produtores de dados ingeridos no Amazon Data Firehose.

O Amazon Data Firehose coleta e publica CloudWatch métricas a cada minuto. Porém, se ocorrerem surtos de dados recebidos apenas por alguns segundos, eles podem não ser totalmente capturados ou visíveis nas métricas de um minuto. Isso ocorre porque CloudWatch as métricas são agregadas do Amazon Data Firehose em intervalos de um minuto.

As métricas coletadas para os fluxos do Firehose são gratuitas. Para obter informações sobre as métricas do Kinesis Agent, consulte Monitoramento da integridade do Kinesis Agent.

Tópicos

- CloudWatch métricas de particionamento dinâmico
- CloudWatch métricas de entrega de dados
- Métricas de ingestão de dados
- Métricas no nível da API CloudWatch
- CloudWatch Métricas de transformação de dados

- CloudWatch Métricas de descompressão de logs
- CloudWatch Métricas de conversão de formato
- Métricas de criptografia no lado do servidor (SSE) CloudWatch
- Dimensões do Amazon Data Firehose
- Métricas de uso do Amazon Data Firehose

CloudWatch métricas de particionamento dinâmico

Se o particionamento dinâmico estiver habilitado, o namespace AWS/Firehose incluirá as métricas a seguir.

Métrica	Descrição
`ActivePartitionsLimit`	O número máximo de partições ativas que um fluxo do Firehose processa antes de enviar dados para o bucket de erros. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`PartitionCount`	O número de partições que estão sendo processadas, em outras palavras, a contagem de partições ativas. Esse número varia entre 1 e o limite de contagem de partições que é 500 (padrão). Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`PartitionCountExceeded`	Essa métrica indica se você está excedendo o limite de contagem de partições. Ela emite 1 ou 0 dependendo do limite ser violado ou não.
`JQProcessing.Duration`	Retorna a quantidade de tempo necessária para executar a expressão JQ na função JQ do Lambda.

Métrica	Descrição
	Unidade: milissegundos
`PerPartitionThroughput`	Indica o throughput que está sendo processado por partição. Essa métrica permite monitorar o throughput por partição. Unidades: StandardUnit. BytesSecond
`DeliveryToS3.ObjectCount`	Indica o número de objetos que estão sendo entregues ao bucket do S3. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

CloudWatch métricas de entrega de dados

O namespace `AWS/Firehose` inclui as métricas de nível do serviço a seguir. Se você observar pequenas quedas na média para `BackupToS3.Success`, `DeliveryToS3.Success`, `DeliveryToSplunk.Success`, `DeliveryToAmazonOpenSearchService.Success` ou`DeliveryToRedshift.Success`, isso não indica que há perda de dados. O Amazon Data Firehose faz novas tentativas quando há erros de entrega e não avança até que os registros sejam entregues com êxito ao destino configurado ou ao bucket do S3 de backup.

Tópicos

- Entrega ao OpenSearch serviço
- Entrega sem OpenSearch servidor
- Entrega ao Amazon Redshift
- Entrega ao Amazon S3
- Entrega ao Snowflake
- Entrega ao Splunk
- Entrega para endpoints de HTTP

Entrega ao OpenSearch serviço

Métrica	Descrição
`DeliveryToAmazonOpenSearchService.Bytes`	O número de bytes indexados para o OpenSearch Service no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
`DeliveryToAmazonOpenSearchService.DataFreshness`	A idade (da entrada no Amazon Data Firehose até agora) do registro mais antigo no Amazon Data Firehose. Quaisquer registros mais antigos do que isso foram enviados para o OpenSearch Serviço. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: segundos
`DeliveryToAmazonOpenSearchService.Records`	O número de registros indexados para o OpenSearch Service no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`DeliveryToAmazonOpenSearchService.Success`	A soma dos registros indexados com êxito.
`DeliveryToS3.Bytes`	O número de bytes entregues ao Amazon S3 no período especificado. O Amazon Data Firehose só emite essa métrica quando você habilita o backup de todos os documentos. Estatísticas válidas: mínima, máxima, média, soma, amostras

Métrica	Descrição
	Unidades: contagem
DeliveryToS3.DataFreshness	A idade (da entrada no Amazon Data Firehose até agora) do registro mais antigo no Amazon Data Firehose. Qualquer registro mais antigo que esse foi enviado para o bucket do S3. O Amazon Data Firehose só emite essa métrica quando você habilita o backup de todos os documentos. Unidades: segundos
DeliveryToS3.Records	O número de registros entregues ao Amazon S3 no período especificado. O Amazon Data Firehose só emite essa métrica quando você habilita o backup de todos os documentos. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
DeliveryToS3.Success	A soma de comandos put do Amazon S3 com êxito. O Amazon Data Firehose sempre emite essa métrica, independentemente de o backup estar habilitado apenas para documentos com falha ou para todos os documentos.
DeliveryToAmazonOpenSearchService.AuthFailure	Authentication/authorization error. Verify the OS/ESpolíticas de cluster e permissões de perfil. 0 indica que não há nenhum problema. 1 indica falha de autenticação.
DeliveryToAmazonOpenSearchService.DeliveryRejected	Erro de entrega rejeitada. Verifique a política de cluster e as permissões de perfil de OS/ES. 0 indica que não há nenhum problema. 1 indica uma falha de entrega.

Entrega sem OpenSearch servidor

Métrica	Descrição
`DeliveryToAmazonOpenSearchServerless.Bytes`	O número de bytes indexados para o OpenSearch Sem Servidor no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
`DeliveryToAmazonOpenSearchServerless.DataFreshness`	A idade (da entrada no Amazon Data Firehose até agora) do registro mais antigo no Amazon Data Firehose. Quaisquer registros mais antigos que esse foram enviados ao Sem OpenSearch Servidor. Unidades: segundos
`DeliveryToAmazonOpenSearchServerless.Records`	O número de registros indexados para o OpenSearch Sem Servidor no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`DeliveryToAmazonOpenSearchServerless.Success`	A soma dos registros indexados com êxito.
`DeliveryToS3.Bytes`	O número de bytes entregues ao Amazon S3 no período especificado. O Amazon Data Firehose só emite essa métrica quando você habilita o backup de todos os documentos. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

Métrica	Descrição
DeliveryToS3.DataFreshness	A idade (da entrada no Amazon Data Firehose até agora) do registro mais antigo no Amazon Data Firehose. Qualquer registro mais antigo que esse foi enviado para o bucket do S3. O Amazon Data Firehose só emite essa métrica quando você habilita o backup de todos os documentos. Unidades: segundos
DeliveryToS3.Records	O número de registros entregues ao Amazon S3 no período especificado. O Amazon Data Firehose só emite essa métrica quando você habilita o backup de todos os documentos. Unidades: contagem
DeliveryToS3.Success	A soma de comandos put do Amazon S3 com êxito. O Amazon Data Firehose sempre emite essa métrica, independentemente de o backup estar habilitado apenas para documentos com falha ou para todos os documentos.
DeliveryToAmazonOpenSearchServerless.AuthFailure	Authentication/authorization error. Verify the OS/ESpolíticas de cluster e permissões de perfil. 0 indica que não há nenhum problema. 1 indica falha de autenticação.
DeliveryToAmazonOpenSearchServerless.DeliveryRejected	Erro de entrega rejeitada. Verifique a política de cluster e as permissões de perfil de OS/ES. 0 indica que não há nenhum problema. 1 indica uma falha de entrega.

Entrega ao Amazon Redshift

Métrica	Descrição
DeliveryToRedshift.Bytes	O número de bytes copiados para o Amazon Redshift no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
DeliveryToRedshift.Records	O número de registros copiados para o Amazon Redshift no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
DeliveryToRedshift.Success	A soma de comandos COPY do Amazon Redshift com êxito.
DeliveryToS3.Bytes	O número de bytes entregues ao Amazon S3 no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
DeliveryToS3.DataFreshness	A idade (da entrada no Amazon Data Firehose até agora) do registro mais antigo no Amazon Data Firehose. Qualquer registro mais antigo que esse foi enviado para o bucket do S3. Unidades: segundos
DeliveryToS3.Records	O número de registros entregues ao Amazon S3 no período especificado.

Métrica	Descrição
	Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`DeliveryToS3.Success`	A soma de comandos put do Amazon S3 com êxito.
`BackupToS3.Bytes`	O número de bytes entregues ao Amazon S3 para backup no período especificado. O Amazon Data Firehose emite essa métrica quando o backup no Amazon S3 está ativado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`BackupToS3.DataFreshness`	A idade (da entrada no Amazon Data Firehose até agora) do registro mais antigo no Amazon Data Firehose. Qualquer registro mais antigo que isso foi entregue ao bucket do Amazon S3 para backup. O Amazon Data Firehose emite essa métrica quando o backup no Amazon S3 está ativado. Estatísticas válidas: mínima, máxima, média, amostras Unidades: segundos
`BackupToS3.Records`	O número de registros entregues ao Amazon S3 para backup no período especificado. O Amazon Data Firehose emite essa métrica quando o backup no Amazon S3 está ativado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

Métrica	Descrição
`BackupToS3.Success`	A soma dos comandos put do Amazon S3 com êxito para backup. O Amazon Data Firehose emite essa métrica quando o backup no Amazon S3 está ativado.

Entrega ao Amazon S3

As métricas na tabela a seguir são relativas à entrega ao Amazon S3 quando ele é o destino principal do fluxo do Firehose.

Métrica	Descrição
`DeliveryToS3.Bytes`	O número de bytes entregues ao Amazon S3 no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
`DeliveryToS3.DataFreshness`	A idade (da entrada no Amazon Data Firehose até agora) do registro mais antigo no Amazon Data Firehose. Qualquer registro mais antigo que esse foi enviado para o bucket do S3. Estatísticas válidas: mínima, máxima, média, amostras Unidades: segundos
`DeliveryToS3.Records`	O número de registros entregues ao Amazon S3 no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`DeliveryToS3.Success`	A soma de comandos put do Amazon S3 com êxito.

Métrica	Descrição
`BackupToS3.Bytes`	O número de bytes entregues ao Amazon S3 para backup no período especificado. O Amazon Data Firehose emite essa métrica quando o backup está habilitado (o que só é possível quando a transformação de dados também está habilitada). Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`BackupToS3.DataFreshness`	A idade (da entrada no Amazon Data Firehose até agora) do registro mais antigo no Amazon Data Firehose. Qualquer registro mais antigo que isso foi entregue ao bucket do Amazon S3 para backup. O Amazon Data Firehose emite essa métrica quando o backup está habilitado (o que só é possível quando a transformação de dados também está habilitada). Estatísticas válidas: mínima, máxima, média, amostras Unidades: segundos
`BackupToS3.Records`	O número de registros entregues ao Amazon S3 para backup no período especificado. O Amazon Data Firehose emite essa métrica quando o backup está habilitado (o que só é possível quando a transformação de dados também está habilitada). Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

Métrica	Descrição
`BackupToS3.Success`	A soma dos comandos put do Amazon S3 com êxito para backup. O Amazon Data Firehose emite essa métrica quando o backup está habilitado (o que só é possível quando a transformação de dados também está habilitad a).

Entrega ao Snowflake

Métrica	Descrição
`DeliveryToSnowflake.Bytes`	O número de bytes entregues ao Snowflake no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
`DeliveryToSnowflak e.DataFreshness`	A idade (da chegada ao Farehose até agora) do registro mais antigo no Firehose. Quaisquer registros mais antigos que esse foram enviados ao Snowflake. Observe que pode levar alguns segundos para confirmar os dados no Snowflake após a chamada de inserção do Firehose obtiver êxito. Para saber o tempo necessário para confirmar os dados no Snowflake, consulte a métrica `DeliveryToSnowflake.DataCommitLatency` . Estatísticas válidas: mínima, máxima, média, amostras Unidades: segundos
`DeliveryToSnowflak e.DataCommitLatency`	O tempo necessário para que os dados sejam confirmad os no Snowflake depois de o Firehose ter inserido os registros com êxito. Estatísticas válidas: mínima, máxima, média, amostras

Métrica	Descrição
	Unidades: segundos
DeliveryToSnowflake.Records	O número de registros entregues ao Snowflake no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
DeliveryToSnowflake.Success	A soma das chamadas de inserção com êxito feitas para o Snowflake.
DeliveryToS3.Bytes	O número de bytes entregues ao Amazon S3 no período especificado. Essa métrica só estará disponível quando a entrega para o Snowflake falhar e o Firehose tentar fazer backup dos dados com falha no S3. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
DeliveryToS3.Records	O número de registros entregues ao Amazon S3 no período especificado. Essa métrica só estará disponível quando a entrega para o Snowflake falhar e o Firehose tentar fazer backup dos dados com falha no S3. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
DeliveryToS3.Success	A soma de comandos put do Amazon S3 com êxito. Essa métrica só estará disponível quando a entrega para o Snowflake falhar e o Firehose tentar fazer backup dos dados com falha no S3.

Métrica	Descrição
BackupToS3.DataFreshness	A idade (da entrada no Firehose até agora) do registro mais antigo no Firehose. Quaisquer registros mais antigos que isso foram entregues ao bucket do Amazon S3 para backup. Essa métrica está disponível quando o fluxo do Firehose é configurado para fazer backup de todos os dados. Estatísticas válidas: mínima, máxima, média, amostras Unidades: segundos
BackupToS3.Records	O número de registros entregues ao Amazon S3 para backup no período especificado. Essa métrica está disponível quando o fluxo do Firehose é configurado para fazer backup de todos os dados. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: Contagem
BackupToS3.Bytes	O número de bytes entregues ao Amazon S3 para backup no período especificado. Essa métrica está disponível quando o fluxo do Firehose é configurado para fazer backup de todos os dados. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: Contagem
BackupToS3.Success	A soma de comandos put do Amazon S3 com êxito para backup. O Firehose emite essa métrica quando o fluxo do Firehose é configurado para fazer backup de todos os dados.

Entrega ao Splunk

Métrica	Descrição
`DeliveryToSplunk.Bytes`	O número de bytes enviados para o Splunk ao longo do período de tempo especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
`DeliveryToSplunk.DataAckLatency`	O tempo aproximado que leva para receber uma mensagem de confirmação do Splunk depois que o Amazon Data Firehose envia os dados. A tendência de aumento ou diminuição dessa métrica é mais útil que o valor aproximado absoluto. As tendências de aumento podem indicar taxas de confirmação e indexação mais lentas dos indexadores da Splunk. Estatísticas válidas: mínima, máxima, média, amostras Unidades: segundos
`DeliveryToSplunk.DataFreshness`	A idade (da entrada no Amazon Data Firehose até agora) do registro mais antigo no Amazon Data Firehose. Qualquer registro mais antigo que esse foi enviado para o Splunk. Estatísticas válidas: mínima, máxima, média, amostras Unidades: segundos
`DeliveryToSplunk.Records`	O número de registros enviados para o Splunk ao longo do período de tempo especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

Métrica	Descrição
`DeliveryToSplunk.Success`	A soma dos registros indexados com êxito.
`DeliveryToS3.Success`	A soma de comandos put do Amazon S3 com êxito. Essa métrica é emitida quando o backup no Amazon S3 está habilitado.
`BackupToS3.Bytes`	O número de bytes entregues ao Amazon S3 para backup no período especificado. O Amazon Data Firehose emite essa métrica quando o fluxo do Firehose é configurado para fazer backup de todos os documento s. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`BackupToS3.DataFreshness`	A idade (da entrada no Amazon Data Firehose até agora) do registro mais antigo no Amazon Data Firehose. Qualquer registro mais antigo que isso foi entregue ao bucket do Amazon S3 para backup. O Amazon Data Firehose emite essa métrica quando o fluxo do Firehose é configurado para fazer backup de todos os documento s. Estatísticas válidas: mínima, máxima, média, amostras Unidades: segundos

Métrica	Descrição
`BackupToS3.Records`	O número de registros entregues ao Amazon S3 para backup no período especificado. O Amazon Data Firehose emite essa métrica quando o fluxo do Firehose é configurado para fazer backup de todos os documento s. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`BackupToS3.Success`	A soma dos comandos put do Amazon S3 com êxito para backup. O Amazon Data Firehose emite essa métrica quando o fluxo do Firehose é configurado para fazer backup de todos os documentos.

Entrega para endpoints de HTTP

Métrica	Descrição
`DeliveryToHttpEndp oint.Bytes`	O número de bytes entregues com êxito ao endpoint de HTTP. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
`DeliveryToHttpEndp oint.Records`	O número de registros entregues com êxito ao endpoint de HTTP. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidade: contagens

Métrica	Descrição
`DeliveryToHttpEndpoint.DataFreshness`	A idade do registro mais antigo no Amazon Data Firehose. Estatísticas válidas: mínima, máxima, média, amostras Unidades: segundos
`DeliveryToHttpEndpoint.Success`	A soma de todas as solicitações de entrega de dados com êxito para o endpoint de HTTP Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`DeliveryToHttpEndpoint.ProcessedBytes`	O número de tentativas de bytes processados, incluindo as novas tentativas.
`DeliveryToHttpEndpoint.ProcessedRecords`	O número de tentativas de registro, incluindo as novas tentativas.

Métricas de ingestão de dados

Tópicos

- Ingestão de dados por meio do Kinesis Data Streams
- Ingestão de dados por meio de Direct PUT
- Ingestão de dados do MSK

Ingestão de dados por meio do Kinesis Data Streams

Métrica	Descrição
`DataReadFromKinesisStream.Bytes`	Quando a fonte de dados é um fluxo de dados do Kinesis, essa métrica indica o número de bytes desse

Métrica	Descrição
	fluxo de dados que foram lidos. Esse número inclui releituras devido a failovers. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
`DataReadFromKinesisStream.Records`	Quando a fonte de dados é um fluxo de dados do Kinesis, essa métrica indica o número de registros desse fluxo de dados que foram lidos. Esse número inclui releituras devido a failovers. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`ThrottledDescribeStream`	O número total de vezes que a operação `DescribeStream` será limitada quando a fonte de dados for um streaming de dados do Kinesis. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`ThrottledGetRecords`	O número total de vezes que a operação `GetRecords` será limitada quando a fonte de dados for um streaming de dados do Kinesis. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

Métrica	Descrição
`ThrottledGetShardIterator`	O número total de vezes que a operação `GetShardIterator` será limitada quando a fonte de dados for um streaming de dados do Kinesis. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`KinesisMillisBehindLatest`	Quando a fonte de dados for um streaming de dados do Kinesis, esta métrica indica o número de milissegundos de diferença que o último registro de leitura está em relação ao registro mais recente nesse streaming. Estatísticas válidas: mínima, máxima, média, amostras Unidade: milissegundo

Ingestão de dados por meio de Direct PUT

Métrica	Descrição
`BackupToS3.Bytes`	O número de bytes entregues ao Amazon S3 para backup no período especificado. O Amazon Data Firehose emite essa métrica quando a transformação de dados está habilitada para os destinos do Amazon S3 ou do Amazon Redshift. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
`BackupToS3.DataFreshness`	A idade (da entrada no Amazon Data Firehose até agora) do registro mais antigo no Amazon Data Firehose. Qualquer registro mais antigo que isso foi entregue ao

Métrica	Descrição
	bucket do Amazon S3 para backup. O Amazon Data Firehose emite essa métrica quando a transformação de dados está habilitada para os destinos do Amazon S3 ou do Amazon Redshift. Estatísticas válidas: mínima, máxima, média, amostras Unidades: segundos
`BackupToS3.Records`	O número de registros entregues ao Amazon S3 para backup no período especificado. O Amazon Data Firehose emite essa métrica quando a transformação de dados está habilitada para os destinos do Amazon S3 ou do Amazon Redshift. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`BackupToS3.Success`	A soma dos comandos put do Amazon S3 com êxito para backup. O Amazon Data Firehose emite essa métrica quando a transformação de dados está habilitada para os destinos do Amazon S3 ou do Amazon Redshift.
`BytesPerSecondLimit`	O número máximo atual de bytes por segundo que um fluxo do Firehose pode ingerir antes da limitação. Para solicitar um aumento desse limite, acesse o AWS Support Center e escolha Criar caso e, depois, escolha Aumento de limite de serviço.
`DeliveryToAmazonOpenSearchService.Bytes`	O número de bytes indexados para o OpenSearch Service no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes

Métrica	Descrição
DeliveryToAmazonOpenSearchService.DataFreshness	A idade (da entrada no Amazon Data Firehose até agora) do registro mais antigo no Amazon Data Firehose. Quaisquer registros mais antigos do que isso foram enviados para o OpenSearch Serviço. Estatísticas válidas: mínima, máxima, média, amostras Unidades: segundos
DeliveryToAmazonOpenSearchService.Records	O número de registros indexados para o OpenSearch Service no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
DeliveryToAmazonOpenSearchService.Success	A soma dos registros indexados com êxito.
DeliveryToRedshift.Bytes	O número de bytes copiados para o Amazon Redshift no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
DeliveryToRedshift.Records	O número de registros copiados para o Amazon Redshift no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
DeliveryToRedshift.Success	A soma de comandos COPY do Amazon Redshift com êxito.

Métrica	Descrição
DeliveryToS3.Bytes	O número de bytes entregues ao Amazon S3 no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
DeliveryToS3.DataFreshness	A idade (da entrada no Amazon Data Firehose até agora) do registro mais antigo no Amazon Data Firehose. Qualquer registro mais antigo que esse foi enviado para o bucket do S3. Estatísticas válidas: mínima, máxima, média, amostras Unidades: segundos
DeliveryToS3.Records	O número de registros entregues ao Amazon S3 no período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
DeliveryToS3.Success	A soma de comandos put do Amazon S3 com êxito.
DeliveryToSplunk.Bytes	O número de bytes enviados para o Splunk ao longo do período de tempo especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes

Métrica	Descrição
`DeliveryToSplunk.DataAckLatency`	O tempo aproximado que leva para receber uma mensagem de confirmação do Splunk depois que o Amazon Data Firehose envia os dados. A tendência de aumento ou diminuição dessa métrica é mais útil que o valor aproximado absoluto. As tendências de aumento podem indicar taxas de confirmação e indexação mais lentas dos indexadores da Splunk. Estatísticas válidas: mínima, máxima, média, amostras Unidades: segundos
`DeliveryToSplunk.DataFreshness`	A idade (da entrada no Amazon Data Firehose até agora) do registro mais antigo no Amazon Data Firehose. Qualquer registro mais antigo que esse foi enviado para o Splunk. Estatísticas válidas: mínima, máxima, média, amostras Unidades: segundos
`DeliveryToSplunk.Records`	O número de registros enviados para o Splunk ao longo do período de tempo especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`DeliveryToSplunk.Success`	A soma dos registros indexados com êxito.

Métrica	Descrição
`IncomingBytes`	O número de bytes ingeridos com êxito no fluxo do Firehose durante o período especificado. O controle de utilização pode ser aplicado à ingestão de dados quando ela excede um dos limites do fluxo do Firehose. Os dados com controle de utilização não serão contabilizados em `IncomingBytes` .. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
`IncomingPutRequests`	O número de PutRecordBatch solicitações PutRecord e solicitações com êxito durante o período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`IncomingRecords`	O número de registros ingeridos com êxito no fluxo do Firehose durante o período especificado. O controle de utilização pode ser aplicado à ingestão de dados quando ela excede um dos limites do fluxo do Firehose. Os dados com controle de utilização não serão contabilizados em `IncomingRecords` .. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

Métrica	Descrição
`RecordsPerSecondLimit`	O número máximo atual de registros por segundo que um fluxo do Firehose pode ingerir antes da limitação. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`ThrottledRecords`	O número de registros que foram limitados porque a ingestão de dados excedeu um dos limites de fluxo do Firehose. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

Ingestão de dados do MSK

Métrica	Descrição
`DataReadFromSource.Records`	O número de registros lidos do Kafka Topic de origem. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`DataReadFromSource.Bytes`	O número de bytes lidos do Kafka Topic de origem. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes

Métrica	Descrição
SourceThrottled.Delay	A quantidade de tempo que o cluster do Kafka de origem demora para retornar os registros do Kafka Topic de origem. Estatísticas válidas: mínima, máxima, média, amostras Unidade: milissegundos
BytesPerSecondLimit	Limite atual do throughput com o qual o Firehose lerá em cada partição do Kafka Topic de origem. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes/segundo
KafkaOffsetLag	A diferença entre o maior deslocamento de registro que o Firehose leu no Kafka Topic de origem e o maior deslocamento de registro disponível do Kafka Topic de origem. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
FailedValidation.Records	O número de registros que não foram aprovados na validação de registros. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

Métrica	Descrição
`FailedValidation.Bytes`	O número de bytes que não foram aprovados na validação de registros. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
`DataReadFromSource.Backpressured`	Indica que um fluxo do Firehose está atrasado na leitura de registros da partição da fonte porque BytesPerSecondLimit cada partição foi excedida ou porque o fluxo normal de entrega está lento ou parou Unidade: booleano

Métricas no nível da API CloudWatch

O namespace `AWS/Firehose` inclui as métricas de nível da API a seguir.

Métrica	Descrição
`DescribeDeliveryStream.Latency`	O tempo gasto por operação `DescribeDeliveryStream` , medido ao longo do período de tempo especificado. Estatísticas válidas: mínima, máxima, média, amostras Unidade: milissegundos
`DescribeDeliveryStream.Requests`	O número total de solicitações `DescribeDeliveryStream` . Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

Métrica	Descrição
`ListDeliveryStreams.Latency`	O tempo gasto por operação `ListDeliveryStream` , medido ao longo do período de tempo especificado. Estatísticas válidas: mínima, máxima, média, amostras Unidade: milissegundos
`ListDeliveryStreams.Requests`	O número total de solicitações `ListFirehose` . Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`PutRecord.Bytes`	O número de bytes colocados no fluxo do Firehose usando `PutRecord` durante período especificado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
`PutRecord.Latency`	O tempo gasto por operação `PutRecord` , medido ao longo do período de tempo especificado. Estatísticas válidas: mínima, máxima, média, amostras Unidade: milissegundos
`PutRecord.Requests`	O número total de solicitações `PutRecord` , igual ao número total de registros das operações `PutRecord` . Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

Métrica	Descrição
PutRecordBatch.Bytes	O número de bytes colocados no fluxo do Firehose usando PutRecordBatch durante período especific ado. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
PutRecordBatch.Latency	O tempo gasto por operação PutRecordBatch , medido ao longo do período de tempo especificado. Estatísticas válidas: mínima, máxima, média, amostras Unidade: milissegundos
PutRecordBatch.Records	O número total de registros das operações PutRecord Batch . Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
PutRecordBatch.Requests	O número total de solicitações PutRecordBatch . Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

Métrica	Descrição
PutRequestsPerSecondLimit	O número máximo de solicitações put por segundo que um fluxo do Firehose pode processar antes da limitação. Esse número inclui PutRecordBatch solicitações PutRecord e solicitações. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
ThrottledDescribeStream	O número total de vezes que a operação DescribeStream será limitada quando a fonte de dados for um streaming de dados do Kinesis. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
ThrottledGetRecords	O número total de vezes que a operação GetRecords será limitada quando a fonte de dados for um streaming de dados do Kinesis. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
ThrottledGetShardIterator	O número total de vezes que a operação GetShardIterator será limitada quando a fonte de dados for um streaming de dados do Kinesis. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

Métrica	Descrição
UpdateDeliveryStream.Latency	O tempo gasto por operação UpdateDeliveryStream , medido ao longo do período de tempo especificado. Estatísticas válidas: mínima, máxima, média, amostras Unidade: milissegundos
UpdateDeliveryStream.Requests	O número total de solicitações UpdateDeliveryStream . Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

CloudWatch Métricas de transformação de dados

Se a transformação de dados com o Lambda estiver habilitada, o AWS/Firehose namespace incluirá as métricas a seguir.

Métrica	Descrição
ExecuteProcessing.Duration	O tempo que leva cada invocação da função do Lambda realizada pelo Firehose. Unidade: milissegundos
ExecuteProcessing.Success	A soma das invocações com êxito da função do Lambda sobre a soma do total de invocações da função do Lambda.
SucceedProcessing.Records	O número de registros processados com êxito no período especificado. Unidades: contagem
SucceedProcessing.Bytes	O número de bytes processados com êxito no período especificado.

Métrica	Descrição
	Unidades: bytes

CloudWatch Métricas de descompressão de logs

Se a descompactação estiver habilitada para entrega de CloudWatch logs, o `AWS/Firehose` namespace incluirá as métricas a seguir.

Métrica	Descrição
`OutputDecompressedBytes.Success`	Dados descompactados em bytes com êxito Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
`OutputDecompressedBytes.Failed`	Dados descompactados em bytes com falha Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: bytes
`OutputDecompressedRecords.Success`	Número de registros descompactados com êxito Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
`OutputDecompressedRecords.Failed`	Número de registros descompactados com falha Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

CloudWatch Métricas de conversão de formato

Se a conversão de formato estiver desativada, o namespace `AWS/Firehose` incluirá as seguintes métricas.

Métrica	Descrição
`SucceedConversion.Records`	O número de registros convertidos com êxito. Unidades: contagem
`SucceedConversion.Bytes`	O tamanho dos registros convertidos com êxito. Unidades: bytes
`FailedConversion.Records`	O número de registros que não puderam ser convertidos. Unidades: contagem
`FailedConversion.Bytes`	O tamanho dos registros que não puderam ser convertidos. Unidades: bytes

Métricas de criptografia no lado do servidor (SSE) CloudWatch

O namespace do `AWS/Firehose` inclui as seguintes métricas relacionadas à SSE.

Métrica	Descrição
`KMSKeyAccessDenied`	O número de vezes que o serviço encontra uma `KMSAccessDeniedException` para o fluxo do Firehose. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

Métrica	Descrição
KMSKeyDisabled	O número de vezes que o serviço encontra uma KMSDisabledException para o fluxo do Firehose. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
KMSKeyInvalidState	O número de vezes que o serviço encontra uma KMSInvalidStateException para o fluxo do Firehose. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem
KMSKeyNotFound	O número de vezes que o serviço encontra uma KMSNotFoundException para o fluxo do Firehose. Estatísticas válidas: mínima, máxima, média, soma, amostras Unidades: contagem

Dimensões do Amazon Data Firehose

Para filtrar as métricas por fluxo do Firehose, use a dimensão DeliveryStreamName.

Métricas de uso do Amazon Data Firehose

Você pode usar métricas CloudWatch de uso do para fornecer visibilidade sobre o uso de recursos de sua conta. Use essas métricas para visualizar o uso do serviço atual nos CloudWatch gráficos e painéis do.

As métricas de uso da cota do serviço estão no namespace AWS/Usage e são coletadas a cada três minutos.

No momento, o único nome da métrica nesse namespace que é CloudWatch publicado é. `ResourceCount` Essa métrica é publicada com as dimensões `Service`, `Class`, `Type` e `Resource`.

Métrica	Descrição
`ResourceCount`	O número dos recursos especificados em execução em sua conta. Os recursos são definidos pelas dimensões associadas à métrica. A estatística mais útil para essa métrica é MAXIMUM, que representa o número máximo de recursos usados durante o período de 3 minutos.

As dimensões a seguir são usadas para refinar as métricas de uso publicadas pelo Amazon Data Firehose.

Dimensão	Descrição
`Service`	O nome do AWS serviço da que contém o recurso. Para as métricas de uso do Amazon Data Firehose, o valor dessa dimensão é `Firehose`.
`Class`	A classe do recurso sob acompanhamento. As métricas de uso da API do Amazon Data Firehose usam essa dimensão com um valor de `None`.
`Type`	O tipo de recurso que está sendo acompanhado. Atualmente, quando a dimensão Service é `Firehose`, o único valor válido para Type é `Resource`.
`Resource`	O nome do AWS recurso. Atualmente, quando a dimensão Service é `Firehose`, o único valor válido para Resource é `DeliveryStreams` .

CloudWatch Métricas de acesso para o Amazon Data Firehose

É possível monitorar as métricas do Amazon Data Firehose usando o CloudWatch console, a linha de comando ou CloudWatch a API. Os procedimentos a seguir mostram como acessar as métricas usando os seguintes métodos:

Como acessar as métricas usando o CloudWatch console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. Na barra de navegação, escolha uma região.
3. No painel de navegação, selecione Métricas.
4. Escolha o namespace Firehose.
5. Escolha Métricas de fluxo do Firehose ou Métricas do Firehose.
6. Selecione uma métrica a ser adicionada ao gráfico.

Como acessar as métricas usando a AWS CLI

Use as métricas e get-metric-statisticsos comandos da lista.

```
aws cloudwatch list-metrics --namespace "AWS/Firehose"
```

```
aws cloudwatch get-metric-statistics --namespace "AWS/Firehose" \
--metric-name DescribeDeliveryStream.Latency --statistics Average --period 3600 \
--start-time 2017-06-01T00:00:00Z --end-time 2017-06-30T00:00:00Z
```

Monitoramento do Amazon Data Firehose usando logs CloudWatch

O Amazon Data Firehose é integrado ao Amazon CloudWatch Logs para que você possa visualizar os logs de erros específicos quando a invocação de transformação ou entrega de dados do Lambda falhar. É possível habilitar o registro em log dos erros do Amazon Data Firehose ao criar o fluxo do Firehose.

Se você habilitar o registro em log dos erros no console do Amazon Data Firehose, um grupo de logs e os fluxos de logs correspondentes serão criados para o fluxo do Firehose em seu nome. O formato do nome do grupo de logs é `/aws/kinesisfirehose/`*`delivery-stream-name`*, em que

delivery-stream-name é o nome do fluxo do Firehose correspondente. DestinationDelivery é um fluxo de logs criado e usado para registrar em log quaisquer erros relacionados à entrega ao destino principal. Outro fluxo de logs denominado BackupDelivery só é criado se o backup do S3 estiver habilitado para o destino. O fluxo de logs de BackupDelivery é usado para registrar em log quaisquer erros relacionados à entrega ao backup do S3.

Por exemplo, se você criar o fluxo do Firehose "MyStream" com o Amazon Redshift como destino e habilitar o registro em log dos erros do Amazon Data Firehose, os itens a seguir serão criados em seu nome: um grupo de logs aws/kinesisfirehose/MyStream denominado e dois fluxos de logs denominados e. DestinationDelivery BackupDelivery Neste exemplo, DestinationDelivery será usado para registrar em log quaisquer erros relacionados à entrega ao destino do Amazon Redshift e também ao destino intermediário do S3. BackupDelivery, caso o backup do S3 esteja habilitado, será usado para registrar em log quaisquer erros relacionados à entrega ao bucket de backup do S3.

É possível habilitar o registro em log dos erros do Amazon Data Firehose AWS CloudFormation usando a AWS CLI, a API ou o com a CloudWatchLoggingOptions configuração. Para fazer isso, crie um grupo de logs e um fluxo de log com antecedência. É recomendável reservar esse grupo de logs e esse fluxo de logs exclusivamente para o registro em log dos erros do Amazon Data Firehose. Além disso, garanta que a política do IAM; associada tenha a permissão "logs:putLogEvents". Para obter mais informações, consulte Controle de acesso com o Amazon Data Firehose.

Observe que o Amazon Data Firehose não garante que todos os logs de erros de entrega sejam enviados para CloudWatch o Logs. Quando a taxa de falhas de entrega é alta, o Amazon Data Firehose faz uma amostragem dos logs de erros de entrega antes de enviá-los para CloudWatch o Logs.

Existe um custo nominal para os logs de erros enviados para o CloudWatch Logs. Para obter mais informações, consulte Amazon CloudWatch Pricing.

Conteúdo

Erros de entrega de dados

A lista a seguir inclui os códigos e as mensagens de erro de entrega de dados para cada destino do Amazon Data Firehose. Cada mensagem de erro também descreve a ação apropriada a ser executada para resolver o problema.

Erros

Erros de entrega de dados do Amazon S3

O Amazon Data Firehose pode enviar os erros relacionados ao Amazon S3 a seguir para o Logs. CloudWatch

Código de erro	Mensagem de erros e informações
`S3.KMS.NotFoundException`	“A AWS KMS chave fornecida não foi encontrada. Se você acredita estar usando uma AWS KMS chave do válida com o perfil correto, verifique se há algum problema na conta à qual a AWS KMS chave do está associada.”
`S3.KMS.RequestLimitExceeded`	"O limite de solicitações do KMS por segundo foi excedido durante a tentativa de criptografar objetos do S3. Aumente o limite de solicitação por segundo." Para obter mais informações, consulte Limites no Guia do desenvolvedor do AWS Key Management Service .

Código de erro	Mensagem de erros e informações
S3.AccessDenied	"Acesso negado. Certifique-se de que a política de confiança do perfil do IAM fornecido permita que o Amazon Data Firehose assuma o perfil e que a política de acesso padrão permite o acesso ao bucket do S3."
S3.AccountProblem	“Há um problema na AWS conta da que impede a conclusão bem-sucedida da operação. Entre em contato com o AWS Support."
S3.AllAccessDisabled	"O acesso à conta fornecida foi desabilitado. Entre em contato com AWS o Support.”
S3.InvalidPayer	"O acesso à conta fornecida foi desabilitado. Entre em contato com AWS o Support.”
S3.NotSignedUp	"A conta não está cadastrada no Amazon S3. Cadastre a conta ou use outra conta."
S3.NoSuchBucket	"O bucket especificado não existe. Crie o bucket ou use um bucket existente."
S3.MethodNotAllowed	"O método especificado não é permitido neste recurso. Modifique a política do bucket para que sejam concedidas as permissões corretas de operação do Amazon S3."
InternalError	"Ocorreu um erro durante a entrega dos dados. Será feita uma nova tentativa de entrega; se o erro persistir, ele será reportado à AWS para resolução.”
S3.KMS.KeyDisabled	"A chave do KMS fornecida está desabilitada. Habilite a chave ou use uma chave diferente."
S3.KMS.InvalidStateException	"A chave do KMS fornecida está em um estado inválido. Use uma chave diferente."
KMS.InvalidStateException	"A chave do KMS fornecida está em um estado inválido. Use uma chave diferente."

Código de erro	Mensagem de erros e informações
`KMS.DisabledException`	"A chave do KMS fornecida está desabilitada. Corrija a chave ou use uma chave diferente."
`S3.SlowDown`	"A taxa de solicitação de put para o bucket especificado era muito alta. Aumente o tamanho do buffer do fluxo do Firehose ou reduza as solicitações de put de outras aplicações."
`S3.SubscriptionRequired`	"O acesso foi negado ao chamar o S3. Certifique-se de que o perfil do IAM e a chave do KMS (se fornecida) passadas tenham a assinatura do Amazon S3."
`S3.InvalidToken`	"O formato do token fornecido é malformado ou é inválido por algum outro motivo. Verifique as credenciais fornecidas."
`S3.KMS.KeyNotConfigured`	"Chave do KMS não configurada. Configure o KMSMaster KeyID ou desabilite a criptografia para o bucket do S3.”
`S3.KMS.AsymmetricCMKNotSupported`	“O Amazon S3 só é compatível com simetria. CMKs Não é possível usar uma CMK assimétrica para criptografar dados no Amazon S3. Como obter o tipo da CMK, use a DescribeKey operação do KMS.”
`S3.IllegalLocationConstraintException`	"Atualmente, o Firehose usa o endpoint global do s3 para entrega de dados ao bucket do s3 configurado. A região do bucket do s3 configurado não é compatível com o endpoint global do s3. Crie um fluxo do Firehose na mesma região do bucket do s3 ou use o bucket do s3 na região com suporte para o endpoint global."
`S3.InvalidPrefixConfigurationException`	"O prefixo do s3 personalizado usado para a avaliação do timestamp é inválido. Verifique se o prefixo s3 contém expressões válidas para a data e hora atuais do ano."
`DataFormatConversion.MalformedData`	"Caractere ilegal encontrado entre tokens."

Erros de entrega de dados de tabelas do Apache Iceberg

Para erros de entrega de dados de tabelas do Apache Iceberg, consulte Entrega de dados às tabelas do Apache Iceberg.

Erros de entrega de dados do Amazon Redshift

O Amazon Data Firehose pode enviar os erros relacionados ao Amazon Redshift a seguir para o Logs. CloudWatch

Código de erro	Mensagem de erros e informações
`Redshift.TableNotFound`	"A tabela em que os dados foram carregados não foi encontrada. Verifique se a tabela especificada existe." Não foi possível encontrar no Amazon Redshift a tabela de destinos na qual os dados do S3 deveriam ser copiados. Observe que o Amazon Data Firehose não criará a tabela do Amazon Redshift se ela não existir.
`Redshift.SyntaxError`	"O comando COPY contém um erro de sintaxe. Repita o comando."
`Redshift.AuthenticationFailed`	"Falha na autenticação do nome de usuário e da senha fornecidos. Forneça um nome de usuário e uma senha válidos."
`Redshift.AccessDenied`	"Acesso negado. Certifique-se de que a política de confiança do perfil do IAM fornecido permita que o Amazon Data Firehose assuma o perfil."
`Redshift.S3BucketAccessDenied`	"O comando COPY não pôde acessar o bucket do S3. Verifique se a política de acesso padrão do perfil do IAM fornecido permite o acesso ao bucket do S3."
`Redshift.DataLoadFailed`	"Falha no carregamento de dados na tabela. Verifique se há detalhes na tabela de sistema STL_LOAD_ERRORS."
`Redshift.ColumnNotFound`	"Uma coluna do comando COPY não consta na tabela. Especifique um nome de coluna válido."

Código de erro	Mensagem de erros e informações
`Redshift.DatabaseNotFound`	"Não foi possível encontrar o banco de dados especificado na configuração do destino ou na URL de JDBC do Amazon Redshift. Especifique um nome de banco de dados válido."
`Redshift.IncorrectCopyOptions`	"Foram fornecidas opções de COPY redundantes ou conflitantes. Algumas opções não são compatíveis em determinadas combinações. Verifique a referência do comando COPY para obter mais informações." Para obter mais informações sobre as visualizações do Amazon Redshift, consulte Comando COPY do Amazon Redshift no Guia do desenvolvedor de bando de dados do Amazon.
`Redshift.MissingColumn`	"Há uma coluna no esquema de tabelas definida como NOT NULL sem o valor DEFAULT e não incluída na lista de colunas. Exclua essa coluna, certifique-se de que os dados carregados sempre forneçam um valor para essa coluna ou adicione um valor padrão ao esquema do Amazon Redshift para essa tabela."
`Redshift.ConnectionFailed`	"Falha na conexão com o cluster do Amazon Redshift especificado. Certifique-se de que as configurações de segurança permitam conexões do Amazon Data Firehose, que o cluster ou o banco de dados especificado na configuração do destino ou no URL de JDBC do Amazon Redshift estejam corretos e que o cluster esteja disponível."
`Redshift.ColumnMismatch`	"O número de jsonpaths no comando COPY e o número de colunas na tabela de destinos devem corresponder. Repita o comando."
`Redshift.IncorrectOrMissingRegion`	"O Amazon Redshift tentou usar o endpoint de região incorreto para acessar o bucket do S3. Especifique um valor de região correto nas opções do comando COPY ou certifique-se de que o bucket do S3 esteja na mesma região do banco de dados do Amazon Redshift."
`Redshift.IncorrectJsonPathsFile`	"O arquivo jsonpaths fornecido não está em um formato JSON compatível. Repita o comando."

Código de erro	Mensagem de erros e informações
`Redshift.MissingS3File`	"Um ou mais arquivos do S3 exigidos pelo Amazon Redshift foram removidos do bucket do S3. Verifique as politicas do bucket do S3 para remover qualquer exclusão automática dos arquivos do S3."
`Redshift.InsufficientPrivilege`	"O usuário não tem permissões para carregar dados na tabela. Verifique se as permissões de usuário do Amazon Redshift incluam o privilégio de INSERT."
`Redshift.ReadOnlyCluster`	"Não é possível executar a consulta porque o sistema está no modo de redimensionamento. Tente executar a consulta novamente mais tarde."
`Redshift.DiskFull`	"Não foi possível carregar os dados porque o disco está cheio. Aumente a capacidade do cluster do Amazon Redshift ou exclua os dados não utilizados para liberar espaço em disco."
`InternalError`	"Ocorreu um erro durante a entrega dos dados. Será feita uma nova tentativa de entrega; se o erro persistir, ele será reportado à AWS para resolução.”
`Redshift.ArgumentNotSupported`	"O comando COPY contém opções sem suporte."
`Redshift.AnalyzeTableAccessDenied`	"Acesso negado. A cópia do S3 para o Redshift está falhando porque a análise da tabela só pode ser feita pelo proprietário da tabela ou do banco de dados."
`Redshift.SchemaNotFound`	“O esquema especificado na configuração do destino DataTableName do Amazon Redshift não foi encontrado. Especifique um nome de esquema válido."

Código de erro	Mensagem de erros e informações
`Redshift.ColumnSpecifiedMoreThanOnce`	"A mesma coluna está especificada mais de uma vez na lista de colunas. Certifique-se de que as colunas duplicadas sejam removidas."
`Redshift.ColumnNotNullWithoutDefault`	"Uma coluna não nula sem DEFAULT não está incluída na lista de colunas. Certifique-se de que essas colunas estejam incluídas na lista de colunas."
`Redshift.IncorrectBucketRegion`	"O Redshift tentou usar um bucket em uma região diferente da região do cluster. Especifique um bucket na mesma região da região do cluster."
`Redshift.S3SlowDown`	"Alta taxa de solicitação ao S3. Reduza a taxa para evitar que o controle de utilização seja aplicado."
`Redshift.InvalidCopyOptionForJson`	"Use um caminho automático do S3 ou válido para copyOption do json."
`Redshift.InvalidCopyOptionJSONPathFormat`	“Falha de COPY com erro\" JSONPath Formato inválido. O índice da matriz está fora do intervalo\". Corrija a JSONPath expressão.”
`Redshift.InvalidCopyOptionRBACAclNotAllowed`	"Falha de COPY com erro\" Não é possível usar a estrutura de acl RBAC enquanto a propagação de permissões não está habilitada. \"
`Redshift.DiskSpaceQuotaExceeded`	"Transação abortada porque a cota de espaço em disco foi excedida. Libere espaço em disco ou solicite uma cota maior para os esquemas."

Código de erro	Mensagem de erros e informações
`Redshift.ConnectionsLimitExceeded`	"Limite de conexão excedido para o usuário."
`Redshift.SslNotSupported`	"A conexão com o cluster especificado do Amazon Redshift falhou porque o servidor não é compatível com SSL. Verifique as configurações do cluster."
`Redshift.HoseNotFound`	"O hose foi excluído. Verifique o status do hose."
`Redshift.Delimiter`	"O delimitador copyOptions no copyCommand é inválido. Certifique-se de que ele seja um caractere único."
`Redshift.QueryCancelled`	"O usuário cancelou a operação COPY."
`Redshift.CompressionMismatch`	"O hose está configurado com UNCOMPRESSED, mas CopyOption inclui um formato de compactação."
`Redshift.EncryptionCredentials`	"A opção ENCRYPTED requer credenciais no formato: 'aws_iam_role=...;master_symmetric_key=...' or 'aws_access_key_id=...;aws_secret_access_key=...[;token=...];master_symmetric_key=...'"
`Redshift.InvalidCopyOptions`	"Opções de configuração de COPY inválidas."
`Redshift.InvalidMessageFormat`	"O comando Copy contém um caractere inválido."

Código de erro	Mensagem de erros e informações
`Redshift.TransactionIdLimitReached`	"Limite de ID de transação atingido."
`Redshift.DestinationRemoved`	"Verifique se o destino do redshift existe e está configurado corretamente na configuração do Firehose."
`Redshift.OutOfMemory`	"O cluster do Redshift está ficando sem memória. Certifique-se de que o cluster tenha capacidade suficiente."
`Redshift.CannotForkProcess`	"O cluster do Redshift está ficando sem memória. Certifique-se de que o cluster tenha capacidade suficiente."
`Redshift.SslFailure`	"A conexão SSL foi fechada durante o handshake."
`Redshift.Resize`	"O Amazon Redshift está redimensionando o cluster. O Firehose não poderá fornecer dados enquanto o cluster estiver sendo redimensionado."
`Redshift.ImproperQualifiedName`	"O nome qualificado é inadequado (muitos nomes pontilhados)."
`Redshift.InvalidJsonPathFormat`	“ JSONPath Formato inválido.”
`Redshift.TooManyConnectionsException`	"Muitas conexões com o Redshift."

Código de erro	Mensagem de erros e informações
`Redshift.PSQLException`	“PSQlExceção observada no Redshift.”
`Redshift.DuplicateSecondsSpecification`	"Especificação de segundos duplicados no formato de data/hora."
`Redshift.RelationCouldNotBeOpened`	"Foi encontrado um erro do Redshift, não foi possível abrir a relação. Verifique os logs do Redshift para o banco de dados especificado."
`Redshift.TooManyClients`	"Exceção do Redshift devido a muitos clientes encontrados. Revisite o máximo de conexões com o banco de dados se houver vários produtores gravando nele simultaneamente."

Erros de entrega de dados do Snowflake

O Firehose pode enviar os erros relacionados ao Snowflake no Logs. CloudWatch

Código de erro	Mensagem de erros e informações
`Snowflake.InvalidUrl`	"O Firehose não conseguiu se conectar ao Snowflake. Certifique-se de que o URL da conta esteja especificado corretamente na configuração de destino do Snowflake."
`Snowflake.InvalidUser`	"O Firehose não conseguiu se conectar ao Snowflake. Certifique-se de que o usuário esteja especificado corretamente na configuração de destino do Snowflake."
`Snowflake.InvalidRole`	"O perfil especificado do Snowflake não existe ou não está autorizado. Certifique-se de que o perfil seja concedido ao usuário especificado"
`Snowflake.InvalidTable`	"A tabela fornecida não existe ou não está autorizada"

Código de erro	Mensagem de erros e informações
`Snowflake.InvalidSchema`	"O esquema fornecido não existe ou não está autorizado"
`Snowflake.InvalidDatabase`	"O banco de dados fornecido não existe ou não está autorizado"
`Snowflake.InvalidPrivateKeyOrPassphrase`	"A chave privada ou frase secreta especificada não é válida. Observe que a chave privada fornecida deve ser uma chave privada PEM RSA válida."
`Snowflake.MissingColumns`	"A solicitação de inserção foi rejeitada devido à falta de colunas na carga útil de entrada. Certifique-se de que os valores sejam especificados para todas as colunas não anuláveis"
`Snowflake.ExtraColumns`	"A solicitação de inserção foi rejeitada devido a colunas extras. As colunas não presentes na tabela não devem ser especificadas"
`Snowflake.InvalidInput`	"A entrega falhou devido ao formato de entrada inválido. Certifique-se de que a carga útil de entrada fornecida esteja no formato JSON aceitável"
`Snowflake.IncorrectValue`	"A entrega falhou devido ao tipo de dados incorreto na carga útil de entrada. Certifique-se de que os valores JSON especificados na carga útil de entrada estejam de acordo com o tipo de dados declarado na definição da tabela do Snowflake."

Erros de entrega de dados do Splunk

O Amazon Data Firehose pode enviar os erros relacionados ao Spunk no Logs. CloudWatch

Código de erro	Mensagem de erros e informações
`Splunk.ProxyWithou`	“Se você tiver um proxy (ELB ou outro) entre o Amazon Data Firehose e o nó do HEC, deverá habilitar as sessões persistentes para aceitar o HEC.” ACKs

Código de erro	Mensagem de erros e informações
`tStickySessions`	
`Splunk.DisabledToken`	"O token do HEC está desativado. Ativar o token para permitir a entrega dados para o Splunk."
`Splunk.InvalidToken`	"O token do HEC é inválido. Atualize o Amazon Data Firehose com um token do HEC válido."
`Splunk.InvalidDataFormat`	"Os dados não estão formatados corretamente. Para ver como formatar os dados corretamente para endpoints de HEC de eventos ou brutos, consulte Dados de evento do Splunk."
`Splunk.InvalidIndex`	"O token do HEC ou a entrada está configurada com um índice inválido. Verifique a configuração do índice e tente novamente."
`Splunk.ServerError`	"A entrega de dados ao Splunk falhou devido a um erro de servidor do nó do HEC. O Amazon Data Firehose fará uma nova tentativa de enviar os dados se o período para novas tentativas no Amazon Data Firehose for maior que 0. Se todas as novas tentativas falharem, o Amazon Data Firehose fará o backup dos dados no Amazon S3."
`Splunk.DisabledAck`	"Confirmação do indexador está desativada para o token do HEC. Ative a confirmação do indexador e tente novamente. Para obter mais informações, consulte Ativar confirmação do indexador."
`Splunk.AckTimeout`	"Não recebeu uma confirmação do HEC antes do tempo limite de confirmação do HEC expirar. Embora o tempo limite para confirmação tenha expirado, é possível que os dados tenham sido anexados com êxito no Splunk. O Amazon Data Firehose faz o backup no Amazon S3 dos dados cujo tempo limite para confirmação expirou."
`Splunk.MaxRetriesFailed`	"Falha para entregar dados para o Splunk ou para receber confirmação. Verifique o status do HEC e tente novamente."

Código de erro	Mensagem de erros e informações
`Splunk.ConnectionTimeout`	"A conexão com o Splunk expirou. Isso pode ser um erro temporário e a será feita uma nova tentativa de solicitação. O Amazon Data Firehose fará o backup dos dados no Amazon S3 se todas as novas tentativas falharem."
`Splunk.InvalidEndpoint`	"Não foi possível se conectar ao endpoint do HEC. Certifique-se de que o URL do endpoint do HEC seja válida e acessível no Amazon Data Firehose."
`Splunk.ConnectionClosed`	"Não foi possível enviar dados para a Splunk devido a uma falha de conexão. Isso pode ser um erro temporário. Aumentar o período para novas tentativas na configuração do Amazon Data Firehose pode evitar essas falhas temporárias."
`Splunk.SSLUnverified`	"Não foi possível se conectar ao endpoint do HEC. O host não corresponde ao certificado fornecido pelo peer. Certifique-se de que o certificado e o host são válidos."
`Splunk.SSLHandshake`	"Não foi possível se conectar ao endpoint do HEC. Certifique-se de que o certificado e o host são válidos."
`Splunk.URLNotFound`	"A URL solicitada não foi encontrada no servidor do Splunk. Verifique o cluster do Splunk e certifique-se de que ele esteja configurado corretamente."
`Splunk.ServerError.ContentTooLarge`	"A entrega de dados para a Splunk falhou devido a um erro no servidor com uma mensagem statusCode: 413: a solicitação que seu cliente enviou era muito grande. Consulte a documentação do splunk para configurar max_content_length."
`Splunk.IndexerBusy`	"A entrega de dados ao Splunk falhou devido a um erro de servidor do nó do HEC. Certifique-se de que o endpoint do HEC ou do Elastic Load Balancer esteja acessível e íntegro."

Código de erro	Mensagem de erros e informações
`Splunk.ConnectionRecycled`	"A conexão do Firehose com o Splunk foi reciclada. A entrega será repetida."
`Splunk.AcknowledgementsDisabled`	"Não foi possível obter confirmações no POST. Certifique-se de que as confirmações estejam habilitadas no endpoint do HEC."
`Splunk.InvalidHecResponseCharacter`	"Caracteres inválidos encontrados na resposta do HEC, certifique-se de verificar o serviço e a configuração do HEC."

ElasticSearch Erros de entrega de dados

O Amazon Data Firehose pode enviar os ElasticSearch erros a seguir para CloudWatch o Logs.

Código de erro	Mensagem de erros e informações
`ES.AccessDenied`	"Acesso negado. Certifique-se de que o perfil do IAM associado fornecido ao firehose não seja excluído."
`ES.ResourceNotFound`	“O domínio especificado do AWS Elasticsearch não existe.”

Erros de entrega de dados do endpoint de HTTPS

O Amazon Data Firehose pode enviar os erros a seguir relacionados ao endpoint HTTP para o Logs. CloudWatch Se nenhum desses erros corresponder ao problema que você está tendo, o erro padrão é o seguinte: "Ocorreu um erro interno ao tentar entregar os dados. Será feita uma nova tentativa de entrega; se o erro persistir, ele será reportado à AWS para resolução.”

Código de erro	Mensagem de erros e informações
`HttpEndpoint.RequestTimeout`	O tempo limite para entrega expirou antes que uma resposta fosse recebida e ela será repetida. Se esse erro persistir, entre em contato com a equipe de atendimento do AWS Firehose.
`HttpEndpoint.ResponseTooLarge`	"A resposta recebida do endpoint é muito grande. Entre em contato com o proprietário do endpoint para resolver esse problema."
`HttpEndpoint.InvalidResponseFromDestination`	"A resposta recebida do endpoint especificado é inválida. Entre em contato com o proprietário do endpoint para resolver o problema."
`HttpEndpoint.DestinationException`	"A resposta a seguir foi recebida do destino do endpoint."
`HttpEndpoint.ConnectionFailed`	"Não foi possível se conectar ao endpoint. Entre em contato com o proprietário do endpoint para resolver esse problema."
`HttpEndpoint.ConnectionReset`	"Não é possível manter a conexão com o endpoint. Entre em contato com o proprietário do endpoint para resolver esse problema."
`HttpEndpoint.ConnectionReset`	"Problemas em manter a conexão com o endpoint. Entre em contato com o proprietário do endpoint."
`HttpEndpoint.ResponseReasonPhraseExceededLimit`	"A frase do motivo da resposta recebida do endpoint excede o limite configurado de 64 caracteres."

Código de erro	Mensagem de erros e informações
`HttpEndpoint.InvalidResponseFromDestination`	"A resposta recebida do endpoint é inválida. Consulte Solução de problemas de endpoints de HTTP na documentação do Firehose para obter mais informações. Motivo: "
`HttpEndpoint.DestinationException`	"A entrega para o endpoint não teve êxito. Consulte Solução de problemas de endpoints de HTTP na documentação do Firehose para obter mais informações. Resposta recebida com código de status "
`HttpEndpoint.InvalidStatusCode`	"Recebeu um código de status de resposta inválido."
`HttpEndpoint.SSLHandshakeFailure`	"Não foi possível concluir um handshake do SSL com o endpoint. Entre em contato com o proprietário do endpoint para resolver esse problema."
`HttpEndpoint.SSLHandshakeFailure`	"Não foi possível concluir um handshake do SSL com o endpoint. Entre em contato com o proprietário do endpoint para resolver esse problema."
`HttpEndpoint.SSLFailure`	"Não foi possível concluir o handshake do TLS com o endpoint. Entre em contato com o proprietário do endpoint para resolver esse problema."
`HttpEndpoint.SSLHandshakeCertificatePathFailure`	"Não foi possível concluir um handshake do SSL com o endpoint devido ao caminho de certificação inválido. Entre em contato com o proprietário do endpoint para resolver esse problema."

Código de erro	Mensagem de erros e informações
`HttpEndpoint.SSLHandshakeCertificatePathValidationFailure`	"Não foi possível concluir um handshake do SSL com o endpoint devido à falha na validação do caminho de certificação. Entre em contato com o proprietário do endpoint para resolver esse problema."
`HttpEndpoint.MakeRequestFailure.IllegalUriException`	“A HttpEndpoint solicitação falhou devido a uma entrada inválida no URI. Certifique-se de que todos os caracteres no URI de entrada sejam válidos."
`HttpEndpoint.MakeRequestFailure.IllegalCharacterInHeaderValue`	“A HttpEndpoint solicitação falhou devido a um erro de resposta ilegal. Caractere ilegal '\n' no valor do cabeçalho."
`HttpEndpoint.IllegalResponseFailure`	“A HttpEndpoint solicitação falhou devido a um erro de resposta ilegal. A mensagem HTTP não deve conter mais de um cabeçalho Content-Type."
`HttpEndpoint.IllegalMessageStart`	“A HttpEndpoint solicitação falhou devido a um erro de resposta ilegal. Início de mensagem HTTP ilegal. Consulte Solução de problemas de endpoints de HTTP na documentação do Firehose para obter mais informações."

Erros de entrega de dados do Amazon OpenSearch Service

Para o destino do OpenSearch Serviço, o Amazon Data Firehose envia erros para CloudWatch os logs à medida que eles são retornados pelo OpenSearch Serviço.

Além dos erros que podem ser OpenSearch retornados dos clusters, você pode encontrar estes dois erros:

- Authentication/authorization error occurs during attempt to deliver data to destination OpenSearch Service cluster. This can happen due to any permission issues and/orintermitentemente quando a configuração do domínio de destino do Amazon Data Firehose OpenSearch Service é modificada. Verifique a política do cluster e as permissões do perfil.
- Não foi possível entregar os dados ao cluster do OpenSearch Service de destino devido a authentication/authorization failures. This can happen due to any permission issues and/or intermitentemente quando a configuração do domínio de destino do Amazon Data Firehose é OpenSearch modificada. Verifique a política do cluster e as permissões do perfil.

Código de erro	Mensagem de erros e informações
`OS.AccessDenied`	"Acesso negado. Certifique-se de que a política de confiança do perfil do IAM fornecido permita que o Firehose assuma o perfil e que a política de acesso padrão permite o acesso À API do Amazon OpenSearch Service.”
`OS.AccessDenied`	"Acesso negado. Certifique-se de que a política de confiança do perfil do IAM fornecido permita que o Firehose assuma o perfil e que a política de acesso padrão permite o acesso À API do Amazon OpenSearch Service.”
`OS.AccessDenied`	"Acesso negado. Certifique-se de que o perfil do IAM associado fornecido ao firehose não seja excluído."
`OS.AccessDenied`	"Acesso negado. Certifique-se de que o perfil do IAM associado fornecido ao firehose não seja excluído."
`OS.ResourceNotFound`	“O domínio especificado OpenSearch do Amazon Service não existe.”

Código de erro	Mensagem de erros e informações
`OS.ResourceNotFound`	“O domínio especificado OpenSearch do Amazon Service não existe.”
`OS.AccessDenied`	"Acesso negado. Certifique-se de que a política de confiança do perfil do IAM fornecido permita que o Firehose assuma o perfil e que a política de acesso padrão permite o acesso À API do Amazon OpenSearch Service.”
`OS.RequestTimeout`	“A solicitação ao cluster do Amazon OpenSearch Service ou à coleção do OpenSearch Sem Servidor atingiu o tempo limite. Certifique-se de que o cluster ou a coleção tenha capacidade suficiente para a workload atual."
`OS.ClusterError`	“O cluster do Amazon OpenSearch Service retornou um erro não especificado.”
`OS.RequestTimeout`	“A solicitação ao cluster do Amazon OpenSearch Service atingiu o tempo limite. Certifique-se de que o cluster tenha capacidade suficiente para a workload atual."
`OS.ConnectionFailed`	“Problemas ao conectar com o cluster do Amazon OpenSearch Service ou com a coleção OpenSearch Sem Servidor. Certifique-se de que o cluster ou a coleção esteja íntegro e acessível."
`OS.ConnectionReset`	“Não é possível manter a conexão com o cluster do Amazon OpenSearch Service ou com a coleção do OpenSearch Serverless. Entre em contato com o proprietário do cluster ou da coleção para resolver esse problema."
`OS.ConnectionReset`	“Problemas em manter a conexão com o cluster do Amazon OpenSearch Service ou com a coleção do OpenSearch Serverless. Certifique-se de que o cluster ou a coleção esteja íntegro e tenha capacidade suficiente para a workload atual."

Código de erro	Mensagem de erros e informações
`OS.ConnectionReset`	“Problemas em manter a conexão com o cluster do Amazon OpenSearch Service ou com a coleção do OpenSearch Serverless. Certifique-se de que o cluster ou a coleção esteja íntegro e tenha capacidade suficiente para a workload atual."
`OS.AccessDenied`	"Acesso negado. Certifique-se de que a política de acesso no cluster do Amazon OpenSearch Service conceda acesso ao perfil do IAM configurado.”
`OS.ValidationException`	“O OpenSearch cluster retornou uma ESService exceção. Um dos motivos é que o cluster foi atualizado para o OS 2.x ou superior, mas o hose ainda tem o TypeName parâmetro configurado. Atualize a configuração da mangueira definindo TypeName a como uma string vazia ou altere o endpoint para o cluster, que é compatível com parâmetro Tipo.”
`OS.ValidationException`	"O membro deve atender ao padrão de expressão regular: [a-z][a-z0-9\ \-]+
`OS.JsonParseException`	“O cluster Amazon OpenSearch Service retornou um JsonParse Exception. Certifique-se de que os dados inseridos sejam válidos."
`OS.AmazonOpenSearchServiceParseException`	“O cluster Amazon OpenSearch Service retornou um AmazonOpenSearchServiceParseException. Certifique-se de que os dados inseridos sejam válidos."
`OS.ExplicitIndexInBulkNotAllowed`	“Certifique-se de querest.action.multi.allow_explicit_index esteja definido como verdadeiro no cluster do Amazon Service.” OpenSearch
`OS.ClusterError`	“O cluster do Amazon OpenSearch Service ou a coleção OpenSearch Sem Servidor retornou um erro não especificado.”

Código de erro	Mensagem de erros e informações
`OS.ClusterBlockException`	“O cluster retornou um ClusterBlockException. Ele pode estar sobrecarregado."
`OS.InvalidARN`	“O ARN do Amazon OpenSearch Service fornecido é inválido. Verifique sua DeliveryStream configuração.”
`OS.MalformedData`	"Um ou mais registros estão malformado. Certifique-se de que cada registro seja um único objeto JSON válido e não contenha novas linhas."
`OS.InternalError`	"Ocorreu um erro durante a tentativa de entrega dos dados. Será feita uma nova tentativa de entrega; se o erro persistir, ele será reportado à AWS para resolução.”
`OS.AliasWithMultipleIndicesNotAllowed`	"O alias tem mais de um índice associado a ele. Certifique-se de que o alias tenha apenas um índice associado a ele."
`OS.UnsupportedVersion`	“No momento, o Amazon OpenSearch Service 6.0 não é compatível com o Amazon Data Firehose. Entre em contato com o AWS Support para obter mais informações.”
`OS.CharConversionException`	"Um ou mais registros continham um caractere inválido."
`OS.InvalidDomainNameLength`	"O tamanho do nome de domínio não está dentro dos limites válidos do sistema operacional."
`OS.VPCDomainNotSupported`	“No momento, não VPCs há suporte para domínios do Amazon OpenSearch Service no momento.”

Código de erro	Mensagem de erros e informações
`OS.ConnectionError`	“O servidor http fechou a conexão inesperadamente. Verifique a integridade do cluster do Amazon OpenSearch Service ou da coleção do OpenSearch Serverless.”
`OS.LargeFieldData`	“O cluster do Amazon OpenSearch Service abortou a solicitação, pois ela continha dados de campo maiores do que o permitido.”
`OS.BadGateway`	“O cluster do Amazon OpenSearch Service ou a coleção OpenSearch Sem Servidor abortou a solicitação com uma resposta: 502 Gateway insatisfatório.”
`OS.ServiceException`	“Erro recebido do cluster do Amazon OpenSearch Service ou da coleção OpenSearch Sem Servidor. Se o cluster ou a coleção estiver por trás de uma VPC, garanta que a configuração da rede permita a conectividade."
`OS.GatewayTimeout`	“O Firehose encontrou erros de tempo limite ao se conectar ao cluster do Amazon OpenSearch Service ou à coleção do OpenSearch Serverless.”
`OS.MalformedData`	“O Amazon Data Firehose não oferece suporte a comandos de API Bulk do Amazon OpenSearch Service dentro do registro do Firehose.”
`OS.ResponseEntryCountMismatch`	"A resposta da API Bulk continha mais entradas do que o número de registros enviados. Certifique-se de que cada registro contenha somente um objeto JSON e de que não haja novas linhas."

Erros de invocação do Lambda

O Amazon Data Firehose pode enviar os erros de invocação do Lambda ao Logs. CloudWatch

Código de erro	Mensagem de erros e informações
`Lambda.AssumeRoleAccessDenied`	"Acesso negado. Certifique-se de que a política de confiança do perfil do IAM fornecido permita que o Amazon Data Firehose assuma o perfil."

Código de erro	Mensagem de erros e informações
`Lambda.InvokeAccessDenied`	"Acesso negado. Certifique-se de que a política de acesso permita o acesso à função do Lambda."
`Lambda.JsonProcessingException`	"Ocorreu um erro ao analisar os registros retornados da função do Lambda. Certifique-se de que os registros retornados sigam o modelo de status exigido pelo Amazon Data Firehose." Para obter mais informações, consulte Parâmetros necessários para transformação de dados.
`Lambda.InvokeLimitExceeded`	"O limite de execução simultânea do Lambda foi excedido. Aumente o limite de execução simultânea." Para obter mais informações, consulte Limites do AWS Lambda no Guia do desenvolvedor do AWS Lambda .
`Lambda.DuplicatedRecordId`	"Foram retornados vários registros com o mesmo ID. Certifique-se de que a função do Lambda retorne um registro exclusivo IDs para cada registro.” Para obter mais informações, consulte Parâmetros necessários para transformação de dados.
`Lambda.MissingRecordId`	“Um ou mais registros não IDs foram devolvidos. Certifique-se de que a função do Lambda retorne todos os registros IDs recebidos.” Para obter mais informações, consulte Parâmetros necessários para transformação de dados.
`Lambda.ResourceNotFound`	"A função do Lambda especificada não existe. Use uma função existente ."
`Lambda.InvalidSubnetIDException`	"O ID de sub-rede especificado na configuração da VPC da função do Lambda é inválido. Verifique se o ID de sub-rede é válido."

Código de erro	Mensagem de erros e informações
`Lambda.InvalidSecurityGroupIDException`	"O ID de grupo de segurança especificado na configuração da VPC da função do Lambda é inválido. Verifique se o ID de security group é válido."
`Lambda.SubnetIPAddressLimitReachedException`	“O não AWS Lambda conseguiu configurar o acesso à VPC para a função do Lambda porque há uma ou mais sub-redes configuradas sem endereços IP disponíveis. Aumente o limite de endereços IP." Para obter mais informações consulte Limites da Amazon VPC: VPP e sub-redes no Manual do usuário da Amazon VPC.
`Lambda.ENILimitReachedException`	“O não AWS Lambda conseguiu criar uma interface de rede elástica (ENI) na VPC especificada como parte da configuração da função do Lambda porque o limite de interfaces de rede foi atingido. Aumente o limite de interfaces de rede." Para obter mais informações, consulte Limites da VPC: interfaces de rede no Guia do usuário da Amazon VPC.
`Lambda.FunctionTimedOut`	A invocação da função do Lambda atingiu o tempo limite. Aumente a configuração de tempo limite na função do Lambda. Para obter mais informações, consulte Configurar tempo limite das funções.
`Lambda.FunctionError`	Isso pode acontecer devido a um dos seguintes erros: • Estrutura da saída inválida. Verifique a função e certifique-se de que a saída esteja no formato necessário. Além disso, certifique-se de que os registros processados contenham um status de resultado válido de `Dropped`, `Ok` ou `ProcessingFailed` . • A função do Lambda foi invocada com êxito, mas retornou um resultado de erro. • O Lambda não pôde descriptografar as variáveis de ambiente porque o acesso ao KMS foi negado. Verifique as configurações de chave do KMS da função, bem como a política de chave. Para obter mais informações, consulte Solução de erros de acesso de chave.

Código de erro	Mensagem de erros e informações
`Lambda.FunctionRequestTimedOut`	O Amazon Data Firehose encontrou o erro: a solicitação não foi concluída antes do erro de configuração do tempo limite da solicitação ao invocar o Lambda. Revisite o código do Lambda para verificar se o código do Lambda deveria ser executado além do tempo limite configurado. Nesse caso, considere ajustar as configurações do Lambda, incluindo memória e tempo limite. Para obter mais informações, consulte Configurar as opções da função do Lambda.
`Lambda.TargetServerFailedToRespond`	O Amazon Data Firehose encontrou um erro. Erro: o servidor de destino não respondeu ao chamar o serviço AWS Lambda.
`Lambda.InvalidZipFileException`	O Amazon Data Firehose foi encontrado InvalidZipFileException ao invocar a função do Lambda. Verifique as configurações da função do Lambda e o arquivo zip do código do Lambda.
`Lambda.InternalServerError`	“O Amazon Data Firehose foi encontrado InternalServerError ao chamar o serviço Lambda AWS . O Amazon Data Firehose tentará enviar dados novamente um número fixo de vezes. Você pode especificar ou substituir as opções de nova tentativa usando o `CreateDeliveryStream` ou. `UpdateDestination` APIs Se o erro persistir, entre em contato com a equipe de suporte do AWS Lambda.
`Lambda.ServiceUnavailable`	O Amazon Data Firehose foi encontrado ServiceUnavailableException ao chamar o serviço Lambda AWS . O Amazon Data Firehose tentará enviar dados novamente um número fixo de vezes. Você pode especificar ou substituir as opções de nova tentativa usando o `CreateDeliveryStream` ou. `UpdateDestination` APIs Se o erro persistir, entre em contato com o suporte do AWS Lambda.
`Lambda.InvalidSecurityToken`	Não é possível invocar a função do Lambda devido ao token de segurança inválido. A invocação do Lambda entre partições não é compatível.

Código de erro	Mensagem de erros e informações
`Lambda.InvocationFailure`	Isso pode acontecer devido a um dos seguintes erros: • O Amazon Data Firehose encontrou erros ao chamar o AWS Lambda. A operação será tentada novamente; se o erro persistir, ele será reportado à AWS para resolução." • O Amazon Data Firehose encontrou um do KMSInvalid StateException Lambda. O Lambda não conseguiu descriptografar as variáveis de ambiente porque a chave do KMS usada está em um estado inválido para descriptografia. Verifique a chave do KMS da função do Lambda. • O Amazon Data Firehose encontrou um do AWS LambdaException Lambda. O Lambda não conseguiu inicializar a imagem do contêiner fornecida. Verifique a imagem. • O Amazon Data Firehose encontrou erros de tempo limite ao chamar o Lambda. AWS O tempo limite máximo da função é de 5 minutos. Para obter mais informações, consulte Duração da execução da transformação de dados.
`Lambda.JsonMappingException`	"Ocorreu um erro ao analisar os registros retornados pela função do Lambda. Certifique-se de que o campo de dados esteja codificado na base 64.

Erros de invocação do Kinesis

O Amazon Data Firehose pode enviar os erros de invocação do Kinesis a seguir para o Logs. CloudWatch

Código de erro	Mensagem de erros e informações
`Kinesis.AccessDenied`	"O acesso foi negado ao chamar o Kinesis. Certifique-se de que a política de acesso no perfil do IAM usado permita o acesso ao Kinesis APIs apropriado.”

Código de erro	Mensagem de erros e informações
`Kinesis.ResourceNotFound`	"O Firehose falhou ao ler o fluxo. Se o Firehose estiver conectado ao Kinesis Stream, o fluxo pode não existir ou o fragmento pode ter sido mesclado ou dividido. Se o Firehose for do DirectPut tipo, ele pode não existir mais.”
`Kinesis.SubscriptionRequired`	"O acesso foi negado ao chamar o Kinesis. Certifique-se de que o perfil do IAM passado para o acesso ao fluxo do Kinesis tenha uma assinatura do AWS Kinesis.”
`Kinesis.Throttling`	"Erro de controle de utilização encontrado ao chamar o Kinesis. Isso pode ser devido a outras aplicações estarem chamando o APIs mesmo fluxo do Firehose ou porque você criou muitos fluxos do Firehose com o mesmo fluxo do Kinesis como fonte.”
`Kinesis.Throttling`	"Erro de controle de utilização encontrado ao chamar o Kinesis. Isso pode ser devido a outras aplicações estarem chamando o APIs mesmo fluxo do Firehose ou porque você criou muitos fluxos do Firehose com o mesmo fluxo do Kinesis como fonte.”
`Kinesis.AccessDenied`	"O acesso foi negado ao chamar o Kinesis. Certifique-se de que a política de acesso no perfil do IAM usado permita o acesso ao Kinesis APIs apropriado.”
`Kinesis.AccessDenied`	"O acesso foi negado ao tentar chamar as operações de API no Kinesis Stream subjacente. Certifique-se de que o perfil do IAM seja propagado e válido."
`Kinesis.KMS.AccessDeniedException`	"O Firehose não tem acesso à chave do KMS usada para criptografar/descriptografar o Kinesis Stream. Por favor, conceda ao perfil de entrega do Firehose acesso à chave."
`Kinesis.KMS.KeyDisabled`	"O Firehose não consegue ler o Kinesis Stream de origem porque a chave do KMS usada para criptografá-lo/descriptografá-lo está desabilitada. Habilite a chave para que as leituras possam continuar."

Código de erro	Mensagem de erros e informações
`Kinesis.KMS.InvalidStateException`	"O Firehose não consegue ler o Kinesis Stream de origem porque a chave do KMS usada para criptografá-lo está em um estado inválido."
`Kinesis.KMS.NotFoundException`	"O Firehose não consegue ler o Kinesis Stream de origem porque a chave do KMS usada para criptografá-lo não foi encontrada."

Erros de invocação do Kinesis DirectPut

O Amazon Data Firehose pode enviar os erros de DirectPut invocação do Kinesis a seguir para o Logs. CloudWatch

Código de erro	Mensagem de erros e informações
`Firehose.KMS.AccessDeniedException`	"O Firehose não tem acesso à chave do KMS. Por favor, verifique a política de chave."
`Firehose.KMS.InvalidStateException`	"O Firehose não consegue descriptografar os dados porque a chave do KMS usada para criptografá-los está em um estado inválido."
`Firehose.KMS.NotFoundException`	"O Firehose não consegue descriptografar os dados porque a chave do KMS usada para criptografá-los não foi encontrada."
`Firehose.KMS.KeyDisabled`	"O Firehose não consegue descriptografar os dados porque a chave do KMS usada para criptografar os dados está desabilitada. Habilite a chave para que a entrega de dados possa prosseguir."

AWS Glue Erros de invocação comuns

O Amazon Data Firehose pode enviar os erros a seguir de AWS Glue invocação do ao Logs. CloudWatch

Código de erro	Mensagem de erros e informações
`DataFormatConversion.InvalidSchema`	"O esquema é inválido."
`DataFormatConversion.EntityNotFound`	“O especificado table/database could not be found. Please ensure that the table/database existe e que os valores fornecidos na configuração do esquema estão corretos, especialmente no que diz respeito ao uso de maiúsculas e minúsculas.”
`DataFormatConversion.InvalidInput`	"Não foi possível encontrar um esquema correspondente do glue. Certifique-se de que o banco de dados especificado com o ID do catálogo fornecido exista."
`DataFormatConversion.InvalidInput`	"Não foi possível encontrar um esquema correspondente do glue. Certifique-se de que o ARN passado esteja no formato correto."
`DataFormatConversion.InvalidInput`	"Não foi possível encontrar um esquema correspondente do glue. Certifique-se de que o catalogId fornecido seja válido."
`DataFormatConversion.InvalidVersionId`	"Não foi possível encontrar um esquema correspondente do glue. Certifique-se de que a versão especificada da tabela exista."

Código de erro	Mensagem de erros e informações
`DataFormatConversion.NonExistentColumns`	"Não foi possível encontrar um esquema correspondente do glue. Certifique-se de que a tabela esteja configurada com um descritor de armazenamento não nulo contendo as colunas de destino."
`DataFormatConversion.AccessDenied`	"Acesso negado ao assumir o perfil. Certifique-se de que o perfil especificado na configuração de conversão de formato de dados tenha concedido permissão ao serviço Firehose para assumi-lo."
`DataFormatConversion.ThrottledByGlue`	"Erro de controle de utilização encontrado ao chamar o Glue. Aumente o limite da taxa de solicitações ou reduza a taxa atual de chamadas ao glue por outras aplicações."
`DataFormatConversion.AccessDenied`	"O acesso foi negado ao chamar o Glue. Certifique-se de que o perfil especificado na configuração de conversão de formato de dados tenha as permissões necessárias."
`DataFormatConversion.InvalidGlueRole`	"Perfil inválido. Certifique-se de que o perfil especificado na configuração de conversão de formato de dados exista."
`DataFormatConversion.InvalidGlueRole`	"O token de segurança incluído na solicitação é inválido. Certifique-se de que o perfil do IAM associado fornecido ao firehose não seja excluído."
`DataFormatConversion.GlueNotAvailableInRegion`	“O AWS Glue ainda não está disponível na região que você especificou; especifique uma região diferente.”

Código de erro	Mensagem de erros e informações
`DataFormatConversion.GlueEncryptionException`	"Houve um erro ao recuperar a chave-mestra. Certifique-se de que a chave exista e tenha as permissões de acesso corretas."
`DataFormatConversion.SchemaValidationTimeout`	"O tempo limite foi atingido ao recuperar a tabela do Glue. Se você tiver um grande número de versões de tabela do Glue, adicione a permissão 'glue:GetTableVersion' (recomendada) ou exclua as versões não utilizadas da tabela. Se você não tiver um grande número de tabelas no Glue, entre em contato com o AWS Support.”
`DataFirehose.InternalError`	"O tempo limite foi atingido ao recuperar a tabela do Glue. Se você tiver um grande número de versões de tabela do Glue, adicione a permissão 'glue:GetTableVersion' (recomendada) ou exclua as versões não utilizadas da tabela. Se você não tiver um grande número de tabelas no Glue, entre em contato com o AWS Support.”
`DataFormatConversion.GlueEncryptionException`	"Houve um erro ao recuperar a chave-mestra. Certifique-se de que a chave exista e que o estado esteja correto."

DataFormatConversion Erros de invocação comuns

O Amazon Data Firehose pode enviar os erros a seguir de DataFormatConversion invocação do ao Logs. CloudWatch

Código de erro	Mensagem de erros e informações
`DataFormatConversion.InvalidSchema`	"O esquema é inválido."

Código de erro	Mensagem de erros e informações
`DataFormatConversion.ValidationException`	"Os nomes e tipos das colunas não devem ser strings vazias."
`DataFormatConversion.ParseError`	"Foi encontrado um JSON malformados."
`DataFormatConversion.MalformedData`	"Os dados não correspondem ao esquema."
`DataFormatConversion.MalformedData`	"O comprimento da chave json não deve ser maior que 262.144"
`DataFormatConversion.MalformedData`	"Os dados não podem ser decodificados como UTF-8."
`DataFormatConversion.MalformedData`	"Caractere ilegal encontrado entre tokens."
`DataFormatConversion.InvalidTypeFormat`	"O formato do tipo é inválido. Verifique a sintaxe do tipo."

Código de erro	Mensagem de erros e informações
`DataFormatConversion.InvalidSchema`	"Esquema inválido. Certifique-se de que não haja caracteres especiais nem espaços em branco nos nomes das colunas."
`DataFormatConversion.InvalidRecord`	"O registro não está de acordo com o esquema. Uma ou mais chaves de mapa eram inválidas para map<string,string>."
`DataFormatConversion.MalformedData`	"O JSON recebido continha uma primitiva no nível superior. O nível superior deve ser um objeto ou uma matriz."
`DataFormatConversion.MalformedData`	"O JSON recebido continha uma primitiva no nível superior. O nível superior deve ser um objeto ou uma matriz."
`DataFormatConversion.MalformedData`	"O registro estava vazio ou continha apenas espaços em branco."
`DataFormatConversion.MalformedData`	"Foram encontrados caracteres inválidos."
`DataFormatConversion.MalformedData`	"Foi encontrado um formato de timestamp inválido ou incompatível. Consulte o Guia do desenvolvedor do Firehose para ver os formatos de carimbo de data/hora com suporte."

Código de erro	Mensagem de erros e informações
`DataFormatConversion.MalformedData`	"Um tipo escalar foi encontrado nos dados, mas um tipo complexo estava especificado no esquema."
`DataFormatConversion.MalformedData`	"Os dados não correspondem ao esquema."
`DataFormatConversion.MalformedData`	"Um tipo escalar foi encontrado nos dados, mas um tipo complexo estava especificado no esquema."
`DataFormatConversion.ConversionFailureException`	"ConversionFailureException"
`DataFormatConversion.DataFormatConversionCustomerErrorException`	"DataFormatConversionCustomerErrorException"

Código de erro	Mensagem de erros e informações
`DataFormatConversion.DataFormatConversionCustomerErrorException`	"DataFormatConversionCustomerErrorException"
`DataFormatConversion.MalformedData`	"Os dados não correspondem ao esquema."
`DataFormatConversion.InvalidSchema`	"O esquema é inválido."
`DataFormatConversion.MalformedData`	"Os dados não correspondem ao esquema. Formato inválido para uma ou mais datas."
`DataFormatConversion.MalformedData`	"Os dados contêm uma estrutura JSON altamente aninhada que não é compatível."
`DataFormatConversion.EntityNotFound`	“O especificado table/database could not be found. Please ensure that the table/database existe e que os valores fornecidos na configuração do esquema estão corretos, especialmente no que diz respeito ao uso de maiúsculas e minúsculas.”

Código de erro	Mensagem de erros e informações
`DataFormatConversion.InvalidInput`	"Não foi possível encontrar um esquema correspondente do glue. Certifique-se de que o banco de dados especificado com o ID do catálogo fornecido exista."
`DataFormatConversion.InvalidInput`	"Não foi possível encontrar um esquema correspondente do glue. Certifique-se de que o ARN passado esteja no formato correto."
`DataFormatConversion.InvalidInput`	"Não foi possível encontrar um esquema correspondente do glue. Certifique-se de que o catalogId fornecido seja válido."
`DataFormatConversion.InvalidVersionId`	"Não foi possível encontrar um esquema correspondente do glue. Certifique-se de que a versão especificada da tabela exista."
`DataFormatConversion.NonExistentColumns`	"Não foi possível encontrar um esquema correspondente do glue. Certifique-se de que a tabela esteja configurada com um descritor de armazenamento não nulo contendo as colunas de destino."
`DataFormatConversion.AccessDenied`	"Acesso negado ao assumir o perfil. Certifique-se de que o perfil especificado na configuração de conversão de formato de dados tenha concedido permissão ao serviço Firehose para assumi-lo."
`DataFormatConversion.ThrottledByGlue`	"Erro de controle de utilização encontrado ao chamar o Glue. Aumente o limite da taxa de solicitações ou reduza a taxa atual de chamadas ao glue por outras aplicações."

Código de erro	Mensagem de erros e informações
`DataFormatConversion.AccessDenied`	"O acesso foi negado ao chamar o Glue. Certifique-se de que o perfil especificado na configuração de conversão de formato de dados tenha as permissões necessárias."
`DataFormatConversion.InvalidGlueRole`	"Perfil inválido. Certifique-se de que o perfil especificado na configuração de conversão de formato de dados exista."
`DataFormatConversion.GlueNotAvailableInRegion`	“O AWS Glue ainda não está disponível na região que você especificou; especifique uma região diferente.”
`DataFormatConversion.GlueEncryptionException`	"Houve um erro ao recuperar a chave-mestra. Certifique-se de que a chave exista e tenha as permissões de acesso corretas."
`DataFormatConversion.SchemaValidationTimeout`	"O tempo limite foi atingido ao recuperar a tabela do Glue. Se você tiver um grande número de versões de tabela do Glue, adicione a permissão 'glue:GetTableVersion' (recomendada) ou exclua as versões não utilizadas da tabela. Se você não tiver um grande número de tabelas no Glue, entre em contato com o AWS Support.”
`DataFirehose.InternalError`	"O tempo limite foi atingido ao recuperar a tabela do Glue. Se você tiver um grande número de versões de tabela do Glue, adicione a permissão 'glue:GetTableVersion' (recomendada) ou exclua as versões não utilizadas da tabela. Se você não tiver um grande número de tabelas no Glue, entre em contato com o AWS Support.”

Código de erro	Mensagem de erros e informações
`DataFormatConversion.MalformedData`	"Um ou mais campos têm formato incorreto."

CloudWatch Logs de acesso para o Amazon Data Firehose

É possível visualizar os logs de erros relacionados a falhas na entrega de dados do Amazon Data Firehose usando o console do Amazon Data Firehose ou o console do. CloudWatch Os procedimentos a seguir mostram como acessar os logs de erros usando estes dois métodos:

Para acessar os logs de erros usando o console do Amazon Data Firehose

1. Faça login no AWS Management Console e abra o console do Firehose em /firehose https://console.aws.amazon.com
2. Na barra de navegação, escolha uma AWS região da.
3. Escolha um nome de fluxo do Firehose para acessar a página de detalhes do fluxo do Firehose.
4. Escolha Error Log para exibir uma lista de logs de erros relacionados à falha de entrega de dados.

Para acessar os logs de erros usando o CloudWatch console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. Na barra de navegação, escolha uma Região.
3. No painel de navegação, selecione Logs.
4. Escolha um grupo de logs e um fluxo de logs para visualizar uma lista de logs de erros relacionados à falha de entrega de dados.

Monitoramento da integridade do Kinesis Agent

O Kinesis Agent publica CloudWatch métricas personalizadas com um namespace de. AWS KinesisAgent Ele ajuda a avaliar a integridade do agente, enviando dados para o Amazon Data

Firehose conforme especificado e consumindo a quantidade apropriada de CPU e de recursos de memória no produtor de dados.

As métricas, como número de registros e bytes enviados, são úteis para compreender a taxa em que o agente está enviando dados ao fluxo do Firehose. Quando essas métricas ficarem abaixo dos limites esperados em alguns percentuais ou caírem para zero, isso poderá indicar problemas de configuração, erros de rede ou problemas de integridade do agente. As métricas como consumo de CPU e memória no host e contadores de erros do agente indicam uso de recurso por parte do produtor de dados e fornece informações sobre erros potenciais de configuração ou de host. Por fim, o agente também registra exceções de serviço para ajudar a investigar problemas do agente.

As métricas do agente são reportadas na região especificada na configuração de agente `cloudwatch.endpoint`. Para obter mais informações, consulte Especificação das definições de configuração do agente.

As métricas do Cloudwatch publicadas de vários Kinesis Agents são agregadas ou combinadas.

Há um custo nominal para as métricas emitidas pelo Kinesis Agent, que são habilitadas por padrão. Para obter mais informações, consulte Amazon CloudWatch Pricing.

Monitor com CloudWatch

O Kinesis Agent envia as métricas a seguir para. CloudWatch

Métrica	Descrição
`BytesSent`	O número de bytes enviados para o fluxo do Firehose no período especificado. Unidades: bytes
`RecordSendAttempts`	O número de tentativas de registro (primeira vez ou como nova tentativa) em uma chamada para `PutRecordBatch` no período especificado. Unidades: contagem
`RecordSendErrors`	O número de registros que retornaram status de falha em uma chamada para `PutRecordBatch` , incluindo novas tentativas, no período especificado.

Métrica	Descrição
	Unidades: contagem
`ServiceErrors`	O número de chamadas para `PutRecordBatch` que resultaram em erro de serviço (diferente de um erro de controle de utilização) no período especificado. Unidades: contagem

Registro em log de chamadas de API do Amazon Data Firehose com AWS CloudTrail

O Amazon Data Firehose é integrado com o AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, um perfil ou um AWS serviço da no Amazon Data Firehose. CloudTrail captura as chamadas de API para o Amazon Data Firehose como eventos. As chamadas capturadas incluem as chamadas do console do Amazon Data Firehose e as chamadas de código para as operações de API do Amazon Data Firehose. Se você criar uma trilha, poderá habilitar a entrega contínua de CloudTrail eventos para um bucket do Amazon S3, incluindo eventos para o Amazon Data Firehose. Se não configurar uma trilha, você ainda poderá visualizar os eventos mais recentes no CloudTrail console do em Event history (Histórico de eventos). Usando as informações coletadas pelo CloudTrail, é possível determinar a solicitação feita ao Amazon Data Firehose, o endereço IP do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita, além de detalhes adicionais.

Para saber mais sobre o CloudTrail, incluindo como configurá-lo e ativá-lo, consulte o AWS CloudTrail Manual do usuário do.

Informações sobre Firehose em CloudTrail

CloudTrail O está habilitado na sua AWS conta da ao criá-la. Quando uma atividade de evento com suporte ocorre no Amazon Data Firehose, ela é registrada em um CloudTrail evento juntamente com outros eventos de AWS serviços da no Histórico de eventos. É possível visualizar, pesquisar e baixar eventos recentes em sua AWS conta. Para obter mais informações, consulte Visualizar eventos com o histórico de eventos do CloudTrail.

Para ter um registro contínuo de eventos na sua AWS conta da, incluindo os eventos do Amazon Data Firehose, crie uma trilha. Uma trilha permite CloudTrail entregar arquivos de log a um bucket

do Amazon S3. Por padrão, quando você cria uma trilha no console, ela é aplicada a todas as AWS regiões da. A trilha registra logs de eventos de todas as regiões na AWS partição da e entrega os arquivos de log para o bucket do Amazon S3 especificado por você. Além disso, é possível configurar outros AWS serviços da para analisar mais ainda mais e agir com base nos dados de eventos coletados nos CloudTrail logs do. Para obter mais informações, consulte:

- Visão Geral para Criar uma Trilha
- CloudTrail Serviços e integrações compatíveis
- Configurando notificações do Amazon SNS para CloudTrail
- Recebimento de arquivos de CloudTrail log de várias regiões e Recebimento de arquivos de CloudTrail log de várias contas

O Amazon Data Firehose oferece suporte ao registro em log das seguintes ações como eventos em arquivos de CloudTrail log:

- CreateDeliveryStream
- DeleteDeliveryStream
- DescribeDeliveryStream
- ListDeliveryStreams
- ListTagsForDeliveryStream
- TagDeliveryStream
- StartDeliveryStreamEncryption
- StopDeliveryStreamEncryption
- UntagDeliveryStream
- UpdateDestination

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar o seguinte:

- Se a solicitação foi feita com credenciais de usuário-raiz ou do AWS Identity and Access Management (IAM).
- Se a solicitação foi feita com credenciais de segurança temporárias de uma função ou de um usuário federado.
- Se a solicitação foi feita por outro AWS serviço da.

Para obter mais informações, consulte Elemento userIdentity do CloudTrail.

Exemplo: Entradas de arquivo de log do Firehose

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log a um bucket do Amazon S3 especificado. 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, a data e hora da ação, parâmetros de solicitação, e assim por diante. arquivos de log do CloudTrail não são um rastreamento de pilha ordenada das chamadas da API pública. Assim, elas não são exibidas em nenhuma ordem específica.

O exemplo a seguir mostra uma entrada de CloudTrail registro que demonstra as `DeleteDeliveryStream` ações `CreateDeliveryStream` `DescribeDeliveryStream``ListDeliveryStreams`,`UpdateDestination`,, e.


```
{
  "Records":[
        {
            "eventVersion":"1.02",
            "userIdentity":{
                "type":"IAMUser",
                "principalId":"AKIAIOSFODNN7EXAMPLE",
                "arn":"arn:aws:iam::111122223333:user/CloudTrail_Test_User",
                "accountId":"111122223333",
                "accessKeyId":"AKIAI44QH8DHBEXAMPLE",
                "userName":"CloudTrail_Test_User"
            },
            "eventTime":"2016-02-24T18:08:22Z",
            "eventSource":"firehose.amazonaws.com",
            "eventName":"CreateDeliveryStream",
            "awsRegion":"us-east-1",
            "sourceIPAddress":"127.0.0.1",
            "userAgent":"aws-internal/3",
            "requestParameters":{
                "deliveryStreamName":"TestRedshiftStream",
                "redshiftDestinationConfiguration":{
                "s3Configuration":{
                    "compressionFormat":"GZIP",
                    "prefix":"prefix",
                    "bucketARN":"arn:aws:s3:::amzn-s3-demo-bucket",
                    "roleARN":"arn:aws:iam::111122223333:role/Firehose",
                    "bufferingHints":{
```

```
                        "sizeInMBs":3,
                        "intervalInSeconds":900
                    },
                    "encryptionConfiguration":{
                        "kMSEncryptionConfig":{
                            "aWSKMSKeyARN":"arn:aws:kms:us-east-1:key"
                        }
                    }
                },
                "clusterJDBCURL":"jdbc:redshift://example.abc123.us-
west-2.redshift.amazonaws.com:5439/dev",
                "copyCommand":{
                    "copyOptions":"copyOptions",
                    "dataTableName":"dataTable"
                },
                "password":"",
                "username":"",
                "roleARN":"arn:aws:iam::111122223333:role/Firehose"
            }
        },
        "responseElements":{
            "deliveryStreamARN":"arn:aws:firehose:us-
east-1:111122223333:deliverystream/TestRedshiftStream"
        },
        "requestID":"958abf6a-db21-11e5-bb88-91ae9617edf5",
        "eventID":"875d2d68-476c-4ad5-bbc6-d02872cfc884",
        "eventType":"AwsApiCall",
        "recipientAccountId":"111122223333"
    },
    {
        "eventVersion":"1.02",
        "userIdentity":{
            "type":"IAMUser",
            "principalId":"AKIAIOSFODNN7EXAMPLE",
            "arn":"arn:aws:iam::111122223333:user/CloudTrail_Test_User",
            "accountId":"111122223333",
            "accessKeyId":"AKIAI44QH8DHBEXAMPLE",
            "userName":"CloudTrail_Test_User"
        },
        "eventTime":"2016-02-24T18:08:54Z",
        "eventSource":"firehose.amazonaws.com",
        "eventName":"DescribeDeliveryStream",
        "awsRegion":"us-east-1",
        "sourceIPAddress":"127.0.0.1",
```

```
        "userAgent":"aws-internal/3",
        "requestParameters":{
            "deliveryStreamName":"TestRedshiftStream"
        },
        "responseElements":null,
        "requestID":"aa6ea5ed-db21-11e5-bb88-91ae9617edf5",
        "eventID":"d9b285d8-d690-4d5c-b9fe-d1ad5ab03f14",
        "eventType":"AwsApiCall",
        "recipientAccountId":"111122223333"
    },
    {
        "eventVersion":"1.02",
        "userIdentity":{
            "type":"IAMUser",
            "principalId":"AKIAIOSFODNN7EXAMPLE",
            "arn":"arn:aws:iam::111122223333:user/CloudTrail_Test_User",
            "accountId":"111122223333",
            "accessKeyId":"AKIAI44QH8DHBEXAMPLE",
            "userName":"CloudTrail_Test_User"
        },
        "eventTime":"2016-02-24T18:10:00Z",
        "eventSource":"firehose.amazonaws.com",
        "eventName":"ListDeliveryStreams",
        "awsRegion":"us-east-1",
        "sourceIPAddress":"127.0.0.1",
        "userAgent":"aws-internal/3",
        "requestParameters":{
            "limit":10
        },
        "responseElements":null,
        "requestID":"d1bf7f86-db21-11e5-bb88-91ae9617edf5",
        "eventID":"67f63c74-4335-48c0-9004-4ba35ce00128",
        "eventType":"AwsApiCall",
        "recipientAccountId":"111122223333"
    },
    {
        "eventVersion":"1.02",
        "userIdentity":{
            "type":"IAMUser",
            "principalId":"AKIAIOSFODNN7EXAMPLE",
            "arn":"arn:aws:iam::111122223333:user/CloudTrail_Test_User",
            "accountId":"111122223333",
            "accessKeyId":"AKIAI44QH8DHBEXAMPLE",
            "userName":"CloudTrail_Test_User"
```

```
        },
        "eventTime":"2016-02-24T18:10:09Z",
        "eventSource":"firehose.amazonaws.com",
        "eventName":"UpdateDestination",
        "awsRegion":"us-east-1",
        "sourceIPAddress":"127.0.0.1",
        "userAgent":"aws-internal/3",
        "requestParameters":{
            "destinationId":"destinationId-000000000001",
            "deliveryStreamName":"TestRedshiftStream",
            "currentDeliveryStreamVersionId":"1",
            "redshiftDestinationUpdate":{
                "roleARN":"arn:aws:iam::111122223333:role/Firehose",
                "clusterJDBCURL":"jdbc:redshift://example.abc123.us-
west-2.redshift.amazonaws.com:5439/dev",
                "password":"",
                "username":"",
                "copyCommand":{
                    "copyOptions":"copyOptions",
                    "dataTableName":"dataTable"
                },
                "s3Update":{
                    "bucketARN":"arn:aws:s3:::amzn-s3-demo-bucket-update",
                    "roleARN":"arn:aws:iam::111122223333:role/Firehose",
                    "compressionFormat":"GZIP",
                    "bufferingHints":{
                        "sizeInMBs":3,
                        "intervalInSeconds":900
                    },
                    "encryptionConfiguration":{
                        "kMSEncryptionConfig":{
                            "aWSKMSKeyARN":"arn:aws:kms:us-east-1:key"
                        }
                    },
                    "prefix":"arn:aws:s3:::amzn-s3-demo-bucket"
                }
            }
        },
        "responseElements":null,
        "requestID":"d549428d-db21-11e5-bb88-91ae9617edf5",
        "eventID":"1cb21e0b-416a-415d-bbf9-769b152a6585",
        "eventType":"AwsApiCall",
        "recipientAccountId":"111122223333"
    },
```

```
    {
        "eventVersion":"1.02",
        "userIdentity":{
            "type":"IAMUser",
            "principalId":"AKIAIOSFODNN7EXAMPLE",
            "arn":"arn:aws:iam::111122223333:user/CloudTrail_Test_User",
            "accountId":"111122223333",
            "accessKeyId":"AKIAI44QH8DHBEXAMPLE",
            "userName":"CloudTrail_Test_User"
        },
        "eventTime":"2016-02-24T18:10:12Z",
        "eventSource":"firehose.amazonaws.com",
        "eventName":"DeleteDeliveryStream",
        "awsRegion":"us-east-1",
        "sourceIPAddress":"127.0.0.1",
        "userAgent":"aws-internal/3",
        "requestParameters":{
            "deliveryStreamName":"TestRedshiftStream"
        },
        "responseElements":null,
        "requestID":"d85968c1-db21-11e5-bb88-91ae9617edf5",
        "eventID":"dd46bb98-b4e9-42ff-a6af-32d57e636ad1",
        "eventType":"AwsApiCall",
        "recipientAccountId":"111122223333"
    }
  ]
}
```

Exemplos de código para Firehose usando AWS SDKs

Os exemplos de código a seguir mostram como usar o Firehose com um kit de desenvolvimento AWS de software (SDK).

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar perfis de serviço individuais, você pode ver as ações no contexto em seus cenários relacionados.

Cenários são exemplos de código que mostram como realizar tarefas específicas chamando várias funções dentro de um serviço ou combinadas com outros Serviços da AWS.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUso do Firehose com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Exemplos de código

- Exemplos básicos de uso do Firehose AWS SDKs
  - Ações para Firehose usando AWS SDKs
    - Use PutRecord com um AWS SDK ou CLI
    - Use PutRecordBatch com um AWS SDK ou CLI
- Cenários para o uso do Firehose AWS SDKs
  - Uso do Amazon Data Firehose para processar registros individuais e em lote

Exemplos básicos de uso do Firehose AWS SDKs

Os exemplos de código a seguir mostram como usar os conceitos básicos do Amazon Data AWS SDKs Firehose com.

Exemplos

- Ações para Firehose usando AWS SDKs
  - Use PutRecord com um AWS SDK ou CLI
  - Use PutRecordBatch com um AWS SDK ou CLI

Ações para Firehose usando AWS SDKs

Os exemplos de código a seguir demonstram como realizar ações individuais do Firehose com. AWS SDKs Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções para configurar e executar o código.

Esses trechos chamam a API do Firehose e são trechos de código de programas maiores que devem ser executados no contexto. É possível ver as ações em contexto em Cenários para o uso do Firehose AWS SDKs .

Os exemplos a seguir incluem apenas as ações mais utilizadas. Para obter uma lista completa, consulte a Referência de API do Amazon Data Firehose.

Exemplos

- Use PutRecord com um AWS SDK ou CLI
- Use PutRecordBatch com um AWS SDK ou CLI

Use **`PutRecord`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `PutRecord`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Inserção de registros no Firehose

CLI

AWS CLI

Para gravar um registro em um fluxo

O exemplo `put-record` a seguir grava dados em um fluxo. Os dados são codificados no formato Base64.

```
aws firehose put-record \
    --delivery-stream-name my-stream \
    --record '{"Data":"SGVsbG8gd29ybGQ="}'
```

Saída:

```
{
    "RecordId": "RjB5K/nnoGFHqwTsZlNd/
TTqvjE8V5dsyXZTQn2JXrdpMTOwssyEb6nfC8fwf1whhwnItt4mvrn+gsqeK5jB7QjuLg283+Ps4Sz/
j1Xujv31iDhnPdaLw4BOyM9Amv7PcCuB2079RuM0NhoakbyUymlwY8yt20G8X2420wu1jlFafhci4erAt7QhDEvpw
    "Encrypted": false
}
```

Para obter mais informações, consulte Sending Data to an Amazon Kinesis Data Firehose Delivery Stream no Guia do desenvolvedor do Amazon Kinesis Data Firehose.

- Para obter detalhes da API, consulte PutRecordna Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
   /**
    * Puts a record to the specified Amazon Kinesis Data Firehose delivery
stream.
    *
    * @param record The record to be put to the delivery stream. The record must
be a {@link Map} of String keys and Object values.
    * @param deliveryStreamName The name of the Amazon Kinesis Data Firehose
delivery stream to which the record should be put.
    * @throws IllegalArgumentException if the input record or delivery stream
name is null or empty.
    * @throws RuntimeException if there is an error putting the record to the
delivery stream.
    */
   public static void putRecord(Map<String, Object> record, String
deliveryStreamName) {
       if (record == null || deliveryStreamName == null ||
deliveryStreamName.isEmpty()) {
           throw new IllegalArgumentException("Invalid input: record or delivery
stream name cannot be null/empty");
```

```
        }
        try {
            String jsonRecord = new ObjectMapper().writeValueAsString(record);
            Record firehoseRecord = Record.builder()

.data(SdkBytes.fromByteArray(jsonRecord.getBytes(StandardCharsets.UTF_8)))
                .build();

            PutRecordRequest putRecordRequest = PutRecordRequest.builder()
                .deliveryStreamName(deliveryStreamName)
                .record(firehoseRecord)
                .build();

            getFirehoseClient().putRecord(putRecordRequest);
            System.out.println("Record sent: " + jsonRecord);
        } catch (Exception e) {
            throw new RuntimeException("Failed to put record: " + e.getMessage(),
e);
        }
    }
```

- Para obter detalhes da API, consulte PutRecorda Referência AWS SDK for Java 2.x da API.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
class FirehoseClient:
    """
    AWS Firehose client to send records and monitor metrics.

    Attributes:
        config (object): Configuration object with delivery stream name and
 region.
        delivery_stream_name (str): Name of the Firehose delivery stream.
```

```
        region (str): AWS region for Firehose and CloudWatch clients.
        firehose (boto3.client): Boto3 Firehose client.
        cloudwatch (boto3.client): Boto3 CloudWatch client.
    """

    def __init__(self, config):
        """
        Initialize the FirehoseClient.

        Args:
            config (object): Configuration object with delivery stream name and
region.
        """
        self.config = config
        self.delivery_stream_name = config.delivery_stream_name
        self.region = config.region
        self.firehose = boto3.client("firehose", region_name=self.region)
        self.cloudwatch = boto3.client("cloudwatch", region_name=self.region)


    @backoff.on_exception(
        backoff.expo, Exception, max_tries=5, jitter=backoff.full_jitter
    )
    def put_record(self, record: dict):
        """
        Put individual records to Firehose with backoff and retry.

        Args:
            record (dict): The data record to be sent to Firehose.

        This method attempts to send an individual record to the Firehose
delivery stream.
        It retries with exponential backoff in case of exceptions.
        """
        try:
            entry = self._create_record_entry(record)
            response = self.firehose.put_record(
                DeliveryStreamName=self.delivery_stream_name, Record=entry
            )
            self._log_response(response, entry)
        except Exception:
            logger.info(f"Fail record: {record}.")
            raise
```

- Para obter detalhes da API, consulte a PutRecordReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUso do Firehose com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`PutRecordBatch`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `PutRecordBatch`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Inserção de registros no Firehose

CLI

AWS CLI

Como gravar vários registros em um fluxo

O exemplo de `put-record-batch` a seguir grava três registro em um fluxo. Os dados são codificados no formato Base64.

```
aws firehose put-record-batch \
    --delivery-stream-name my-stream \
    --records file://records.json
```

Conteúdo de `myfile.json`:

```
[
    {"Data": "Rmlyc3QgdGhpbmc="},
    {"Data": "U2Vjb25kIHRoaW5n"},
    {"Data": "VGhpcmQgdGhpbmc="}
]
```

Saída:

```
{
    "FailedPutCount": 0,
    "Encrypted": false,
    "RequestResponses": [
        {
            "RecordId": "9D2OJ6t2EqCTZTXwGzeSv/EVHxRoRCw89xd+o3+sXg8DhYOaWKPSmZy/
CGlRVEys1u1xbeKh6VofEYKkoeiDrcjrxhQp9iF7sUW7pujiMEQ5LzlrzCkGosxQn
+3boDnURDEaD42V7GiixpOyLJkYZcae1i7HzlCEoy9LJhMr8EjDSi4Om/9Vc2uhwwuAtGE0XKpxJ2WD7ZRWtAnYlK
        },
        {
            "RecordId": "jFirejqxCLlK5xjH/UNmlMVcjktEN76I7916X9PaZ
+PVaOSXDfU1WGOqEZhxq2js7xcZ552eoeDxsuTU1MSq9nZTbVfb6cQTIXnm/
GsuF37Uhg67GKmR5z90l6XKJ+/
+pDloFv7Hh9a3oUS6wYm3DcNRLTHHAimANp1PhkQvWpvLRfzbuCUkBphR2QVzhP9OiHLbzGwy8/
DfH8sqWEUYASNJKS8GXP5s"
        },
        {
            "RecordId":
 "oy0amQ40o5Y2YV4vxzufdcMOOw6n3EPr3tpPJGoYVNKH4APPVqNcbUgefo1stEFRg4hTLrf2k6eliHu/9+YJ5R3
DTBt3qBlmTj7Xq8SKVbO1S7YvMTpWkMKA86f8JfmT8BMKoMb4XZS/sOkQLe+qh0sYKXWl"
        }
    ]
}
```

Para obter mais informações, consulte Sending Data to an Amazon Kinesis Data Firehose Delivery Stream no Guia do desenvolvedor do Amazon Kinesis Data Firehose.

- Para obter detalhes da API, consulte PutRecordBatchna Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    /**
```

```
    * Puts a batch of records to an Amazon Kinesis Data Firehose delivery
stream.
    *
    * @param records           a list of maps representing the records to be
sent
    * @param batchSize         the maximum number of records to include in each
batch
    * @param deliveryStreamName the name of the Kinesis Data Firehose delivery
stream
    * @throws IllegalArgumentException if the input parameters are invalid (null
or empty)
    * @throws RuntimeException         if there is an error putting the record
batch
    */
   public static void putRecordBatch(List<Map<String, Object>> records, int
batchSize, String deliveryStreamName) {
       if (records == null || records.isEmpty() || deliveryStreamName == null ||
deliveryStreamName.isEmpty()) {
           throw new IllegalArgumentException("Invalid input: records or
delivery stream name cannot be null/empty");
       }
       ObjectMapper objectMapper = new ObjectMapper();

       try {
           for (int i = 0; i < records.size(); i += batchSize) {
               List<Map<String, Object>> batch = records.subList(i, Math.min(i +
batchSize, records.size()));

               List<Record> batchRecords = batch.stream().map(record -> {
                   try {
                       String jsonRecord =
objectMapper.writeValueAsString(record);
                       return Record.builder()

.data(SdkBytes.fromByteArray(jsonRecord.getBytes(StandardCharsets.UTF_8)))
                           .build();
                   } catch (Exception e) {
                       throw new RuntimeException("Error creating Firehose
record", e);
                   }
               }).collect(Collectors.toList());

               PutRecordBatchRequest request = PutRecordBatchRequest.builder()
                   .deliveryStreamName(deliveryStreamName)
```

```
                    .records(batchRecords)
                    .build();

                PutRecordBatchResponse response =
getFirehoseClient().putRecordBatch(request);

                if (response.failedPutCount() > 0) {
                    response.requestResponses().stream()
                        .filter(r -> r.errorCode() != null)
                        .forEach(r -> System.err.println("Failed record: " +
r.errorMessage()));
                }
                System.out.println("Batch sent with size: " +
batchRecords.size());
            }
        } catch (Exception e) {
            throw new RuntimeException("Failed to put record batch: " +
e.getMessage(), e);
        }
    }
```

- Para obter detalhes da API, consulte PutRecordBatcha Referência AWS SDK for Java 2.x da API.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
class FirehoseClient:
    """
    AWS Firehose client to send records and monitor metrics.

    Attributes:
```

```
        config (object): Configuration object with delivery stream name and
region.
        delivery_stream_name (str): Name of the Firehose delivery stream.
        region (str): AWS region for Firehose and CloudWatch clients.
        firehose (boto3.client): Boto3 Firehose client.
        cloudwatch (boto3.client): Boto3 CloudWatch client.
    """

    def __init__(self, config):
        """
        Initialize the FirehoseClient.

        Args:
            config (object): Configuration object with delivery stream name and
region.
        """
        self.config = config
        self.delivery_stream_name = config.delivery_stream_name
        self.region = config.region
        self.firehose = boto3.client("firehose", region_name=self.region)
        self.cloudwatch = boto3.client("cloudwatch", region_name=self.region)


    @backoff.on_exception(
        backoff.expo, Exception, max_tries=5, jitter=backoff.full_jitter
    )
    def put_record_batch(self, data: list, batch_size: int = 500):
        """
        Put records in batches to Firehose with backoff and retry.

        Args:
            data (list): List of data records to be sent to Firehose.
            batch_size (int): Number of records to send in each batch. Default is
500.

        This method attempts to send records in batches to the Firehose delivery
stream.
        It retries with exponential backoff in case of exceptions.
        """
        for i in range(0, len(data), batch_size):
            batch = data[i : i + batch_size]
            record_dicts = [{"Data": json.dumps(record)} for record in batch]
            try:
                response = self.firehose.put_record_batch(
```

```
                    DeliveryStreamName=self.delivery_stream_name,
 Records=record_dicts
                )
                self._log_batch_response(response, len(batch))
            except Exception as e:
                logger.info(f"Failed to send batch of {len(batch)} records.
 Error: {e}")
```

- Para obter detalhes da API, consulte a PutRecordBatchReferência da API AWS SDK for Python (Boto3).

Rust

SDK para Rust

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
async fn put_record_batch(
    client: &Client,
    stream: &str,
    data: Vec<Record>,
) -> Result<PutRecordBatchOutput, SdkError<PutRecordBatchError>> {
    client
        .put_record_batch()
        .delivery_stream_name(stream)
        .set_records(Some(data))
        .send()
        .await
}
```

- Para obter detalhes da API, consulte a PutRecordBatchreferência da API AWS SDK for Rust.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUso do Firehose com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Cenários para o uso do Firehose AWS SDKs

Os exemplos de código a seguir mostram como implementar cenários comuns no Firehose com. AWS SDKs Esses cenários mostram como realizar tarefas específicas chamando várias funções dentro do Firehose ou combinadas com outros Serviços da AWS. Cada cenário inclui um link para o código-fonte completo, onde podem ser encontradas instruções sobre como configurar e executar o código.

Os cenários têm como alvo um nível intermediário de experiência para ajudar você a compreender ações de serviço em contexto.

Exemplos

- Uso do Amazon Data Firehose para processar registros individuais e em lote

Uso do Amazon Data Firehose para processar registros individuais e em lote

Os exemplos de código a seguir mostram como usar o Firehose para processar registros individuais e em lote.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Este exemplo coloca registros individuais e em lote no Firehose.

```
/**
 * Amazon Firehose Scenario example using Java V2 SDK.
 *
```

```
 * Demonstrates individual and batch record processing,
 * and monitoring Firehose delivery stream metrics.
 */
public class FirehoseScenario {

    private static FirehoseClient firehoseClient;
    private static CloudWatchClient cloudWatchClient;

    public static void main(String[] args) {
        final String usage = """
                Usage:
                    <deliveryStreamName>
                Where:
                    deliveryStreamName - The Firehose delivery stream name.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            return;
        }

        String deliveryStreamName = args[0];

        try {
            // Read and parse sample data.
            String jsonContent = readJsonFile("sample_records.json");
            ObjectMapper objectMapper = new ObjectMapper();
            List<Map<String, Object>> sampleData =
 objectMapper.readValue(jsonContent, new TypeReference<>() {});

            // Process individual records.
            System.out.println("Processing individual records...");
            sampleData.subList(0, 100).forEach(record -> {
                try {
                    putRecord(record, deliveryStreamName);
                } catch (Exception e) {
                    System.err.println("Error processing record: " +
 e.getMessage());
                }
            });

            // Monitor metrics.
            monitorMetrics(deliveryStreamName);
```

```
            // Process batch records.
            System.out.println("Processing batch records...");
            putRecordBatch(sampleData.subList(100, sampleData.size()), 500,
deliveryStreamName);
            monitorMetrics(deliveryStreamName);

        } catch (Exception e) {
            System.err.println("Scenario failed: " + e.getMessage());
        } finally {
            closeClients();
        }
    }

    private static FirehoseClient getFirehoseClient() {
        if (firehoseClient == null) {
            firehoseClient = FirehoseClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return firehoseClient;
    }

    private static CloudWatchClient getCloudWatchClient() {
        if (cloudWatchClient == null) {
            cloudWatchClient = CloudWatchClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return cloudWatchClient;
    }

    /**
     * Puts a record to the specified Amazon Kinesis Data Firehose delivery
stream.
     *
     * @param record The record to be put to the delivery stream. The record must
be a {@link Map} of String keys and Object values.
     * @param deliveryStreamName The name of the Amazon Kinesis Data Firehose
delivery stream to which the record should be put.
     * @throws IllegalArgumentException if the input record or delivery stream
name is null or empty.
     * @throws RuntimeException if there is an error putting the record to the
delivery stream.
     */
```

```
   public static void putRecord(Map<String, Object> record, String
deliveryStreamName) {
       if (record == null || deliveryStreamName == null ||
deliveryStreamName.isEmpty()) {
           throw new IllegalArgumentException("Invalid input: record or delivery
stream name cannot be null/empty");
       }
       try {
           String jsonRecord = new ObjectMapper().writeValueAsString(record);
           Record firehoseRecord = Record.builder()

.data(SdkBytes.fromByteArray(jsonRecord.getBytes(StandardCharsets.UTF_8)))
               .build();

           PutRecordRequest putRecordRequest = PutRecordRequest.builder()
               .deliveryStreamName(deliveryStreamName)
               .record(firehoseRecord)
               .build();

           getFirehoseClient().putRecord(putRecordRequest);
           System.out.println("Record sent: " + jsonRecord);
       } catch (Exception e) {
           throw new RuntimeException("Failed to put record: " + e.getMessage(),
e);
       }
   }


   /**
    * Puts a batch of records to an Amazon Kinesis Data Firehose delivery
stream.
    *
    * @param records           a list of maps representing the records to be
sent
    * @param batchSize         the maximum number of records to include in each
batch
    * @param deliveryStreamName the name of the Kinesis Data Firehose delivery
stream
    * @throws IllegalArgumentException if the input parameters are invalid (null
or empty)
    * @throws RuntimeException         if there is an error putting the record
batch
    */
```

```
   public static void putRecordBatch(List<Map<String, Object>> records, int
batchSize, String deliveryStreamName) {
       if (records == null || records.isEmpty() || deliveryStreamName == null ||
deliveryStreamName.isEmpty()) {
           throw new IllegalArgumentException("Invalid input: records or
delivery stream name cannot be null/empty");
       }
       ObjectMapper objectMapper = new ObjectMapper();

       try {
           for (int i = 0; i < records.size(); i += batchSize) {
               List<Map<String, Object>> batch = records.subList(i, Math.min(i +
batchSize, records.size()));

               List<Record> batchRecords = batch.stream().map(record -> {
                   try {
                       String jsonRecord =
objectMapper.writeValueAsString(record);
                       return Record.builder()

.data(SdkBytes.fromByteArray(jsonRecord.getBytes(StandardCharsets.UTF_8)))
                           .build();
                   } catch (Exception e) {
                       throw new RuntimeException("Error creating Firehose
record", e);
                   }
               }).collect(Collectors.toList());

               PutRecordBatchRequest request = PutRecordBatchRequest.builder()
                   .deliveryStreamName(deliveryStreamName)
                   .records(batchRecords)
                   .build();

               PutRecordBatchResponse response =
getFirehoseClient().putRecordBatch(request);

               if (response.failedPutCount() > 0) {
                   response.requestResponses().stream()
                       .filter(r -> r.errorCode() != null)
                       .forEach(r -> System.err.println("Failed record: " +
r.errorMessage()));
               }
               System.out.println("Batch sent with size: " +
batchRecords.size());
```

```
            }
        } catch (Exception e) {
            throw new RuntimeException("Failed to put record batch: " +
e.getMessage(), e);
        }
    }

    public static void monitorMetrics(String deliveryStreamName) {
        Instant endTime = Instant.now();
        Instant startTime = endTime.minusSeconds(600);

        List<String> metrics = List.of("IncomingBytes", "IncomingRecords",
"FailedPutCount");
        metrics.forEach(metric -> monitorMetric(metric, startTime, endTime,
deliveryStreamName));
    }

    private static void monitorMetric(String metricName, Instant startTime,
Instant endTime, String deliveryStreamName) {
        try {
            GetMetricStatisticsRequest request =
GetMetricStatisticsRequest.builder()
                .namespace("AWS/Firehose")
                .metricName(metricName)

.dimensions(Dimension.builder().name("DeliveryStreamName").value(deliveryStreamName).bui
                .startTime(startTime)
                .endTime(endTime)
                .period(60)
                .statistics(Statistic.SUM)
                .build();

            GetMetricStatisticsResponse response =
getCloudWatchClient().getMetricStatistics(request);
            double totalSum =
response.datapoints().stream().mapToDouble(Datapoint::sum).sum();
            System.out.println(metricName + ": " + totalSum);
        } catch (Exception e) {
            System.err.println("Failed to monitor metric " + metricName + ": " +
e.getMessage());
        }
    }

    public static String readJsonFile(String fileName) throws IOException {
```

```
        try (InputStream inputStream =
 FirehoseScenario.class.getResourceAsStream("/" + fileName);
             Scanner scanner = new Scanner(inputStream, StandardCharsets.UTF_8))
 {
            return scanner.useDelimiter("\\\\A").next();
        } catch (Exception e) {
            throw new RuntimeException("Error reading file: " + fileName, e);
        }
    }

    private static void closeClients() {
        try {
            if (firehoseClient != null) firehoseClient.close();
            if (cloudWatchClient != null) cloudWatchClient.close();
        } catch (Exception e) {
            System.err.println("Error closing clients: " + e.getMessage());
        }
    }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK for Java 2.x .
  - PutRecord
  - PutRecordBatch

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Este script insere registros individuais e em lote no Firehose.

```
import json
import logging
import random
```

```
from datetime import datetime, timedelta

import backoff
import boto3

from config import get_config


def load_sample_data(path: str) -> dict:
    """
    Load sample data from a JSON file.

    Args:
        path (str): The file path to the JSON file containing sample data.

    Returns:
        dict: The loaded sample data as a dictionary.
    """
    with open(path, "r") as f:
        return json.load(f)


# Configure logging
logging.basicConfig(level=logging.INFO)
logger = logging.getLogger(__name__)


class FirehoseClient:
    """
    AWS Firehose client to send records and monitor metrics.

    Attributes:
        config (object): Configuration object with delivery stream name and
 region.
        delivery_stream_name (str): Name of the Firehose delivery stream.
        region (str): AWS region for Firehose and CloudWatch clients.
        firehose (boto3.client): Boto3 Firehose client.
        cloudwatch (boto3.client): Boto3 CloudWatch client.
    """

    def __init__(self, config):
        """
        Initialize the FirehoseClient.
```

```
        Args:
            config (object): Configuration object with delivery stream name and
region.
        """
        self.config = config
        self.delivery_stream_name = config.delivery_stream_name
        self.region = config.region
        self.firehose = boto3.client("firehose", region_name=self.region)
        self.cloudwatch = boto3.client("cloudwatch", region_name=self.region)


    @backoff.on_exception(
        backoff.expo, Exception, max_tries=5, jitter=backoff.full_jitter
    )
    def put_record(self, record: dict):
        """
        Put individual records to Firehose with backoff and retry.

        Args:
            record (dict): The data record to be sent to Firehose.

        This method attempts to send an individual record to the Firehose
delivery stream.
        It retries with exponential backoff in case of exceptions.
        """
        try:
            entry = self._create_record_entry(record)
            response = self.firehose.put_record(
                DeliveryStreamName=self.delivery_stream_name, Record=entry
            )
            self._log_response(response, entry)
        except Exception:
            logger.info(f"Fail record: {record}.")
            raise


    @backoff.on_exception(
        backoff.expo, Exception, max_tries=5, jitter=backoff.full_jitter
    )
    def put_record_batch(self, data: list, batch_size: int = 500):
        """
        Put records in batches to Firehose with backoff and retry.

        Args:
```

```
            data (list): List of data records to be sent to Firehose.
            batch_size (int): Number of records to send in each batch. Default is
500.

        This method attempts to send records in batches to the Firehose delivery
stream.
        It retries with exponential backoff in case of exceptions.
        """
        for i in range(0, len(data), batch_size):
            batch = data[i : i + batch_size]
            record_dicts = [{"Data": json.dumps(record)} for record in batch]
            try:
                response = self.firehose.put_record_batch(
                    DeliveryStreamName=self.delivery_stream_name,
Records=record_dicts
                )
                self._log_batch_response(response, len(batch))
            except Exception as e:
                logger.info(f"Failed to send batch of {len(batch)} records.
Error: {e}")


   def get_metric_statistics(
        self,
        metric_name: str,
        start_time: datetime,
        end_time: datetime,
        period: int,
        statistics: list = ["Sum"],
   ) -> list:
        """
        Retrieve metric statistics from CloudWatch.

        Args:
            metric_name (str): The name of the metric.
            start_time (datetime): The start time for the metric statistics.
            end_time (datetime): The end time for the metric statistics.
            period (int): The granularity, in seconds, of the returned data
points.
            statistics (list): A list of statistics to retrieve. Default is
['Sum'].

        Returns:
            list: List of datapoints containing the metric statistics.
```

```
        """
        response = self.cloudwatch.get_metric_statistics(
            Namespace="AWS/Firehose",
            MetricName=metric_name,
            Dimensions=[
                {"Name": "DeliveryStreamName", "Value":
self.delivery_stream_name},
            ],
            StartTime=start_time,
            EndTime=end_time,
            Period=period,
            Statistics=statistics,
        )
        return response["Datapoints"]

    def monitor_metrics(self):
        """
        Monitor Firehose metrics for the last 5 minutes.

        This method retrieves and logs the 'IncomingBytes', 'IncomingRecords',
and 'FailedPutCount' metrics
        from CloudWatch for the last 5 minutes.
        """
        end_time = datetime.utcnow()
        start_time = end_time - timedelta(minutes=10)
        period = int((end_time - start_time).total_seconds())

        metrics = {
            "IncomingBytes": self.get_metric_statistics(
                "IncomingBytes", start_time, end_time, period
            ),
            "IncomingRecords": self.get_metric_statistics(
                "IncomingRecords", start_time, end_time, period
            ),
            "FailedPutCount": self.get_metric_statistics(
                "FailedPutCount", start_time, end_time, period
            ),
        }

        for metric, datapoints in metrics.items():
            if datapoints:
                total_sum = sum(datapoint["Sum"] for datapoint in datapoints)
                if metric == "IncomingBytes":
                    logger.info(
```

```
                        f"{metric}: {round(total_sum)} ({total_sum / (1024 *
1024):.2f} MB)"
                    )
                else:
                    logger.info(f"{metric}: {round(total_sum)}")
            else:
                logger.info(f"No data found for {metric} over the last 5
minutes")


    def _create_record_entry(self, record: dict) -> dict:
        """
        Create a record entry for Firehose.

        Args:
            record (dict): The data record to be sent.

        Returns:
            dict: The record entry formatted for Firehose.

        Raises:
            Exception: If a simulated network error occurs.
        """
        if random.random() < 0.2:
            raise Exception("Simulated network error")
        elif random.random() < 0.1:
            return {"Data": '{"malformed": "data"'}
        else:
            return {"Data": json.dumps(record)}

    def _log_response(self, response: dict, entry: dict):
        """
        Log the response from Firehose.

        Args:
            response (dict): The response from the Firehose put_record API call.
            entry (dict): The record entry that was sent.
        """
        if response["ResponseMetadata"]["HTTPStatusCode"] == 200:
            logger.info(f"Sent record: {entry}")
        else:
            logger.info(f"Fail record: {entry}")

    def _log_batch_response(self, response: dict, batch_size: int):
```

```
        """
        Log the batch response from Firehose.

        Args:
            response (dict): The response from the Firehose put_record_batch API
 call.
            batch_size (int): The number of records in the batch.
        """
        if response.get("FailedPutCount", 0) > 0:
            logger.info(
                f'Failed to send {response["FailedPutCount"]} records in batch of
 {batch_size}'
            )
        else:
            logger.info(f"Successfully sent batch of {batch_size} records")


if __name__ == "__main__":
    config = get_config()
    data = load_sample_data(config.sample_data_file)
    client = FirehoseClient(config)

    # Process the first 100 sample network records
    for record in data[:100]:
        try:
            client.put_record(record)
        except Exception as e:
            logger.info(f"Put record failed after retries and backoff: {e}")
    client.monitor_metrics()

    # Process remaining records using the batch method
    try:
        client.put_record_batch(data[100:])
    except Exception as e:
        logger.info(f"Put record batch failed after retries and backoff: {e}")
    client.monitor_metrics()
```

Este arquivo contém a configuração do script acima.

```
class Config:
    def __init__(self):
        self.delivery_stream_name = "ENTER YOUR DELIVERY STREAM NAME HERE"
```

```
        self.region = "us-east-1"
        self.sample_data_file = (
            "../../../../../scenarios/features/firehose/resources/
sample_records.json"
        )


def get_config():
    return Config()
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência de API do AWS SDK para Python (Boto3).
  - PutRecord
  - PutRecordBatch

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUso do Firehose com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Solução de problemas de erros no Amazon Data Firehose

Se o Firehose encontrar erros durante a entrega ou o processamento de dados, ele tentará novamente até que a duração da repetição configurada expire. Se o período de repetição terminar antes que os dados sejam entregues com êxito, o Firehose fará o backup dos dados para o bucket de backup configurado do S3. Se o destino for o Amazon S3 e houver falha na entrega, ou se a entrega para o bucket do S3 falhar, o Firehose continuará tentando até o período de retenção terminar.

Para obter informações sobre como rastrear erros de entrega usando CloudWatch, consultethe section called “Monitore com CloudWatch registros”.

Direct PUT

Para fluxos do Firehose de `DirectPut`, o Firehose retém os registros por 24 horas. Para um fluxo do Firehose cuja fonte de dados seja um fluxo de dados do Kinesis, é possível alterar o período de retenção conforme descrito em Alteração do período de retenção de dados. Nesse caso, o Firehose tenta novamente as operações a seguir indefinidamente: `DescribeStream`, `GetRecords` e `GetShardIterator`.

Se o fluxo do Firehose usar `DirectPut`, verifique as métricas `IncomingBytes` e `IncomingRecords` para ver se há tráfego de entrada. Se você estiver usando o `PutRecord` ou o `PutRecordBatch`, certifique-se de detectar as exceções e tentar novamente. Recomendamos uma política de repetição com recuo exponencial com tremulação e diversas tentativas. Além disso, se você usar a `PutRecordBatch` API, certifique-se de que seu código verifique o valor de FailedPutCountna resposta mesmo quando a chamada da API for bem-sucedida.

Kinesis Data Stream

Se o fluxo do Firehose usar um fluxo de dados do Kinesis como fonte, verifique as métricas `IncomingBytes` e `IncomingRecords` para o fluxo de dados da fonte. Além disso, certifique-se de que as métricas `DataReadFromKinesisStream.Bytes` e `DataReadFromKinesisStream.Records` estejam sendo emitidas para o fluxo do Firehose.

Problemas comuns

Veja a seguir dicas de solução de problemas para ajudar a resolver problemas comuns ao trabalhar com um fluxo do Firehose.

Fluxo do Firehose indisponível

O stream do Firehose não está disponível como destino para CloudWatch registros, CloudWatch eventos ou ações de AWS IoT, pois alguns AWS serviços só podem enviar mensagens e eventos para um stream do Firehose que esteja no mesmo stream. Região da AWS Verifique se o seu fluxo do Firehose está localizado na mesma região que os outros serviços.

Sem dados no destino

Se não há problemas de ingestão de dados e as métricas emitidas para o fluxo do Firehose parecem boas, mas você não está vendo os dados no destino, verifique a lógica do leitor. Certifique-se de que o leitor esteja analisando corretamente todos os dados.

Métrica de atualidade de dados aumentando ou não emitida

A métrica de atualidade de dados informa o quanto os dados estão atualizados no seu fluxo do Firehose. Ela é a idade do registro de dados mais antigo no fluxo do Firehose, medido a partir do momento em que o Firehose incluiu os dados até o momento atual. O Firehose fornece métricas que podem ser usadas para monitorar a atualidade dos dados. Para identificar a métrica de atualização de dados de um destino específico, consulte the section called “Monitoramento com CloudWatch métricas”.

Se você habilitar o backup de todos os eventos ou todos os documentos, monitore duas métricas de atualização de dados separadas: uma para o destino principal e outra para o backup.

Se a métrica de atualidade de dados não estiver sendo emitida, isso significa que não há entrega ativa para o fluxo do Firehose. Isso acontece quando a entrega de dados está completamente bloqueada ou quando não há dados de entrada.

Se a métrica de atualização de dados estiver aumentando constantemente, isso significa que a entrega de dados está em atraso. Isso pode acontecer por um dos seguintes motivos.

- O destino não comporta a taxa de entrega. Se o Firehose encontrar erros transitórios devido a grande volume de tráfego, a entrega poderá sofrer atrasos. Isso pode acontecer para destinos diferentes do Amazon S3 (pode acontecer para OpenSearch Service, Amazon Redshift ou Splunk). Certifique-se de que o destino tenha capacidade suficiente para comportar o tráfego de entrada.
- O destino é lento. A entrega de dados pode sofrer atrasos se o Firehose encontrar alta latência. Monitore a métrica da latência do destino.

- A função do Lambda está lenta. Isso pode levar a uma taxa de entrega de dados inferior à taxa de ingestão de dados para o fluxo do Firehose. Se possível, melhore a eficiência da função do Lambda. Por exemplo, se a função executa a E/S de rede, use vários threads ou a E/S assíncrona para aumentar o paralelismo. Além disso, considere aumentar o tamanho da memória da função do Lambda para que a alocação de CPU possa aumentar de acordo. Isso pode levar a invocações do Lambda mais rápidas. Para obter informações sobre como configurar funções Lambda, consulte Configurando AWS funções Lambda.
- Há falhas durante a entrega de dados. Para obter informações sobre como monitorar erros usando o Amazon CloudWatch Logs, consultethe section called “Monitore com CloudWatch registros”.
- Se a fonte de dados do fluxo do Firehose for um fluxo de dados do Kinesis, é possível que esteja ocorrendo um controle de utilização. Verifique as métricas `ThrottledGetRecords`, `ThrottledGetShardIterator` e `ThrottledDescribeStream`. Se houver vários consumidores conectados ao fluxo de dados do Kinesis, considere o seguinte:
  - Se as métricas `ThrottledGetRecords` e `ThrottledGetShardIterator` estiverem altas, recomendamos aumentar o número de estilhaços provisionados para o fluxo de dados.
  - Se `ThrottledDescribeStream` for alto, recomendamos que você adicione a `kinesis:listshards` permissão à função configurada em KinesisStreamSourceConfiguration.
- Dicas de baixa capacidade de buffer para o destino. Isso pode aumentar o número de ciclos necessários para que o Firehose atinja o destino, o que pode gerar atrasos na entrega. Considere aumentar o valor das dicas de buffer. Para obter mais informações, consulte BufferingHints.
- Uma longa duração para repetições pode gerar atrasos na entrega quando os erros são frequentes. Considere reduzir a duração das repetições. Além disso, monitore os erros e tente reduzi-los. Para obter informações sobre como monitorar erros usando o Amazon CloudWatch Logs, consultethe section called “Monitore com CloudWatch registros”.
- Se o destino for Splunk e `DeliveryToSplunk.DataFreshness` estiver alto, mas `DeliveryToSplunk.Success` parecer bom, o cluster do Splunk pode estar ocupado. Libere o cluster do Splunk se possível. Como alternativa, entre em contato com o suporte da AWS e solicite um aumento no número de canais que o Firehose está usando para se comunicar com o cluster do Splunk.

Falha na conversão de formato de registro para Apache Parquet

Isso acontece se você pegar dados do DynamoDB que incluem `Set` o tipo, transmiti-los por meio do Lambda para um stream do Firehose e usar AWS Glue Data Catalog an para converter o formato de registro em Apache Parquet.

Quando o AWS Glue rastreador indexa os tipos de dados do conjunto do DynamoDB (`StringSet`,, e`BinarySet`)`NumberSet`, ele os armazena no catálogo de dados como, e, respectivamente. `SET<STRING>` `SET<BIGINT>` `SET<BINARY>` No entanto, para que o Firehose converta os registros de dados para o formato Apache Parquet, ele requer os tipos de dados do Apache Hive. Como os tipos de conjunto não são tipos de dados válidos do Apache Hive, há falha na conversão. Para que a conversão funcione, atualize o catálogo de dados com os tipos de dados do Apache Hive. É possível fazer isso alterando `set` para `array` no catálogo de dados.

Para alterar um ou mais tipos de dados de **`set`** para **`array`** em um catálogo AWS Glue de dados

1. Faça login no AWS Management Console e abra o AWS Glue console em https://console.aws.amazon.com/glue/.
2. No painel esquerdo, no cabeçalho Data catalog (Catálogo de dados), escolha Tables (Tabelas).
3. Na lista de tabelas, escolha o nome da tabela na qual você precisa modificar um ou mais tipos de dados. Você será redirecionado para a página de detalhes da tabela
4. Escolha o botão Editar esquema no canto superior direito da página de detalhes.
5. Na coluna Data type (Tipo de dados), escolha o primeiro tipo de dados `set`.
6. Na lista suspensa Column type (Tipo de coluna), altere o tipo de `set` para `array`.
7. No ArraySchemacampo, insira`array<string>`,`array<int>`, ou`array<binary>`, dependendo do tipo de dados apropriado para seu cenário.
8. Selecione Atualizar.
9. Repita as etapas anteriores para converter outros tipos `set` em tipos `array`.
10. Escolha Salvar.

Campos ausentes para objeto transformado para Lambda

Quando você usa a transformação de dados do Lambda para alterar dados de JSON para o objeto Parquet, alguns campos podem estar ausentes após a transformação. Isso acontecerá se o seu objeto JSON tiver letras maiúsculas e a distinção entre maiúsculas e minúsculas estiver definida

como `false`, o que pode levar a uma incompatibilidade nas chaves JSON após a transformação dos dados, causando a falta de dados no objeto Parquet resultante no bucket do S3.

Para corrigir isso, certifique-se de que a configuração do Firehose tenha `deserializationOption: case.insensitive` definida como `true`, de forma que as chaves JSON correspondam após a transformação.

Solução de problemas do Amazon S3

Verifique o seguinte se os dados não forem entregues ao bucket do Amazon Simple Storage Service (Amazon S3).

- Verifique as métricas `IncomingBytes` e `IncomingRecords` do Firehose para garantir que os dados sejam enviados para o fluxo do Firehose com êxito. Para obter mais informações, consulte Monitoramento do Amazon Data Firehose com métricas CloudWatch .
- Se a transformação de dados com o Lambda estiver habilitada, verifique a métrica `ExecuteProcessingSuccess` do Firehose para se certificar de que o Firehose tenha tentado invocar a função do Lambda. Para obter mais informações, consulte Monitoramento do Amazon Data Firehose com métricas CloudWatch .
- Verifique a métrica do `DeliveryToS3.Success` Firehose para se certificar de que ele tenha tentado colocar dados no bucket do Amazon S3. Para obter mais informações, consulte Monitoramento do Amazon Data Firehose com métricas CloudWatch .
- Habilite do registro de erros caso ele ainda não esteja habilitado e verifique se os logs de erros acusa falha de entrega. Para obter mais informações, consulte Monitoramento do Amazon Data Firehose usando logs CloudWatch .
- Se você ver uma mensagem de erro no registro dizendo “Firehose found InternalServerError when call Amazon S3 service”. A operação será tentada novamente; se o erro persistir, entre em contato com o S3 para resolução." , isso pode ser devido ao aumento significativo nas taxas de solicitação em uma única partição no S3. É possível otimizar os padrões de design de prefixo do S3 para mitigar o problema. Para obter mais informações, consulte Padrões de Design de Práticas Recomendadas: Otimizando a Performance do Amazon S3. Se isso não resolver o problema, entre em contato com o AWS Support para obter mais assistência.
- Certifique-se de que o bucket do Amazon S3 especificado no fluxo do Firehose ainda exista.
- Se a transformação de dados com o Lambda estiver habilitada, certifique-se de que a função do Lambda especificada no fluxo do Firehose ainda exista.

- Certifique-se de que o perfil do IAM especificado no seu fluxo do Firehose tenha acesso ao bucket do S3 e à função do Lambda (se a transformação de dados estiver habilitada). Além disso, certifique-se de que a função do IAM tenha acesso ao grupo de CloudWatch registros e aos fluxos de registros para verificar os registros de erros. Para obter mais informações, consulte Concessão ao Firehose de acesso a um destino do Amazon S3.
- Se você estiver usando transformação de dados, certifique-se de que a função do Lambda nunca retorne respostas cuja carga útil exceda 6 MB. Para obter mais informações, consulte Amazon Data FirehoseData Transformation.

Solução de problemas do Amazon Redshift

Verifique o seguinte se os dados não forem entregues ao cluster provisionado do Amazon Redshift ou ao grupo de trabalho do Amazon Redshift sem servidor.

Os dados são entregues no bucket do S3 antes de serem carregados no Amazon Redshift. Se os dados não forem entregues ao bucket do S3, consulte Solução de problemas do Amazon S3.

- Verifique a métrica `DeliveryToRedshift.Success` do Firehose para garantir que ele tenha tentado copiar dados do bucket do S3 para o cluster provisionado do Amazon Redshift ou para o grupo de trabalho do Amazon Redshift sem servidor. Para obter mais informações, consulte Monitoramento do Amazon Data Firehose com métricas CloudWatch .
- Habilite do registro de erros caso ele ainda não esteja habilitado e verifique se os logs de erros acusa falha de entrega. Para obter mais informações, consulte Monitoramento do Amazon Data Firehose usando logs CloudWatch .
- Verifique a tabela `STL_CONNECTION_LOG` do Amazon Redshift para ver se o Firehose pode fazer conexões com êxito. Nessa tabela, você conseguirá ver as conexões e os respectivos status com base em um nome de usuário. Para obter mais informações, consulte `STL_CONNECTION_LOG` no Guia do desenvolvedor de banco de dados do Amazon Redshift.
- Se a verificação anterior mostrar que as conexões estão sendo estabelecidas, verifique a tabela `STL_LOAD_ERRORS` do Amazon Redshift para saber o motivo da falha do comando COPY. Para obter mais informações, consulte `STL_LOAD_ERRORS` no Guia do desenvolvedor de banco de dados do Amazon Redshift.
- Certifique-se de que a configuração do Amazon Redshift no fluxo do Firehose seja precisa e válida.
- Certifique-se de que o perfil do IAM especificado no seu fluxo do Firehose tenha acesso ao bucket do S3 do qual o Amazon Redshift copia os dados e também a função do Lambda para

transformação de dados (se a transformação de dados estiver habilitada). Além disso, certifique-se de que a função do IAM tenha acesso ao grupo de CloudWatch registros e aos fluxos de registros para verificar os registros de erros. Para obter mais informações, consulte Conceder ao Firehose acesso a um destino do Amazon Redshift .

- Se o seu cluster provisionado do Amazon Redshift ou grupo de trabalho do Amazon Redshift sem servidor estiver em uma nuvem privada virtual (VPC), certifique-se de que o cluster permita o acesso a partir dos endereços IP do Firehose. Para obter mais informações, consulte Conceder ao Firehose acesso a um destino do Amazon Redshift .
- Certifique-se de que o cluster provisionado do Amazon Redshift ou o grupo de trabalho do Amazon Redshift sem servidor esteja disponível publicamente.
- Se você estiver usando transformação de dados, certifique-se de que a função do Lambda nunca retorne respostas cuja carga útil exceda 6 MB. Para obter mais informações, consulte Amazon Data FirehoseData Transformation.

Solução de problemas do Amazon OpenSearch Service

Verifique o seguinte se os dados não forem entregues ao seu domínio OpenSearch de serviço.

É possível fazer o backup simultâneo dos dados no bucket do Amazon S3. Se os dados não forem entregues ao bucket do S3, consulte Solução de problemas do Amazon S3.

- Verifique as métricas `IncomingBytes` e `IncomingRecords` do Firehose para garantir que os dados sejam enviados para o fluxo do Firehose com êxito. Para obter mais informações, consulte Monitoramento do Amazon Data Firehose com métricas CloudWatch .
- Se a transformação de dados com o Lambda estiver habilitada, verifique a métrica `ExecuteProcessingSuccess` do Firehose para se certificar de que o Firehose tenha tentado invocar a função do Lambda. Para obter mais informações, consulte Monitoramento do Amazon Data Firehose com métricas CloudWatch .
- Verifique a `DeliveryToAmazonOpenSearchService.Success` métrica Firehose para garantir que o Firehose tenha tentado indexar dados no cluster de serviços. OpenSearch Para obter mais informações, consulte Monitoramento do Amazon Data Firehose com métricas CloudWatch .
- Habilite do registro de erros caso ele ainda não esteja habilitado e verifique se os logs de erros acusa falha de entrega. Para obter mais informações, consulte Monitoramento do Amazon Data Firehose usando logs CloudWatch .

- Certifique-se de que a configuração do OpenSearch serviço em seu stream do Firehose seja precisa e válida.
- Se a transformação de dados com o Lambda estiver habilitada, certifique-se de que a função do Lambda especificada no fluxo do Firehose ainda exista. Além disso, certifique-se de que a função do IAM tenha acesso ao grupo de CloudWatch registros e aos fluxos de registros para verificar os registros de erros. Para obter mais informações, consulte Concessão FirehoseAccess a um destino OpenSearch de serviço público.
- Certifique-se de que a função do IAM especificada em seu stream do Firehose possa acessar seu cluster de OpenSearch serviços, bucket de backup do S3 e função Lambda (se a transformação de dados estiver ativada). Além disso, certifique-se de que a função do IAM tenha acesso ao grupo de CloudWatch registros e aos fluxos de registros para verificar os registros de erros. Para obter mais informações, consulte Conceder ao Firehose acesso a um destino de serviço público OpenSearch .
- Se você estiver usando transformação de dados, certifique-se de que a função do Lambda nunca retorne respostas cuja carga útil exceda 6 MB. Para obter mais informações, consulte Amazon Data FirehoseData Transformation.
- O Amazon Data Firehosecurently não suporta a entrega de registros CloudWatch para o OpenSearch destino do Amazon Service porque a Amazon CloudWatch combina vários eventos de log em um registro Firehose e o OpenSearch Amazon Service não pode aceitar vários eventos de log em um registro. Como alternativa, você pode considerar o uso do filtro de assinatura do Amazon OpenSearch Service em CloudWatch registros.

Solução de problemas do Splunk

Verifique o seguinte se os dados não forem entregues ao endpoint do Splunk.

- Se a sua plataforma do Splunk estiver em uma VPC, certifique-se de que o Firehose possa acessá-lo. Para obter mais informações, consulte Acesso ao Splunk na VPC.
- Se você usa um balanceador de AWS carga, certifique-se de que seja um Classic Load Balancer ou um Application Load Balancer. Além disso, habilite sessões persistentes com a expiração de cookies desabilitada para o Classic Load Balancer, e a expiração definida como máxima (7 dias) para o Application Load Balancer. Para obter informações sobre como fazer isso, consulte Persistência de sessão baseada na duração para Classic Load Balancer ou Application Load Balancer.

- Revise os requisitos da plataforma Splunk. O complemento do Splunk para o Firehose exige a versão da plataforma Splunk 6.6.X ou posterior. Para obter mais informações, consulte Complemento do Splunk para o Amazon Kinesis Firehose.
- Se você tiver um proxy (Elastic Load Balancing ou outro) entre o Firehose e o nó HTTP Event Collector (HEC), habilite sessões fixas para suportar confirmações HEC (). ACKs
- Verifique se o token do HEC que você está usando é válido.
- Verifique se o token do HEC está ativado.
- Verifique se os dados que você está enviando ao Splunk estão formatados corretamente. Para obter mais informações, consulte Formatar eventos para o Coletor de eventos HTTP.
- Verifique se o token do HEC e o evento de entrada estão configurados com um índice válido.
- Quando um upload para o Splunk falhar devido a um erro do servidor a partir do nó HEC, a solicitação será automaticamente tentada novamente Se todas as novas tentativas falharem, o backup dos dados será feito no Amazon S3. Verifique se os dados aparecem no Amazon S3, o que é uma indicação dessa falha.
- Verifique se você habilitou a confirmação do indexador no token do HEC.
- Aumente o valor de `HECAcknowledgmentTimeoutInSeconds` na configuração de destino do Splunk do seu fluxo do Firehose.
- Aumente o valor de `DurationInSeconds` em `RetryOptions` na configuração do destino do Splunk do seu fluxo do Firehose.
- Verifique o status do HEC.
- Se você estiver usando transformação de dados, certifique-se de que a função do Lambda nunca retorne respostas cuja carga útil exceda 6 MB. Para obter mais informações, consulte Transformação de dados do Amazon Data Firehose.
- Verifique se o parâmetro do Splunk chamado `ackIdleCleanup` está definido como `true`. Ele é "false" por padrão. Para definir o parâmetro como `true`, faça o seguinte:
  - Para uma implantação de nuvem gerenciada do Splunk, envie um caso usando o portal de suporte do Splunk. Nesse caso, peça para que o suporte do Splunk habilite o coletor de eventos HTTP, defina o `ackIdleCleanup` como `true` em `inputs.conf` e crie ou modifique um load balancer para ser usado com esse complemento.
  - Para uma implantação empresarial de Splunk distribuída, defina o parâmetro `ackIdleCleanup` como verdadeiro no arquivo `inputs.conf`. Para usuários do *nix, este arquivo está localizado em `$SPLUNK_HOME/etc/apps/splunk_httpinput/local/`. Para usuários do Windows, está em `%SPLUNK_HOME%\etc\apps\splunk_httpinput\local\`.

  - Para uma implantação empresarial Splunk de única instância, defina o parâmetro `ackIdleCleanup` como `true` no arquivo `inputs.conf`. Para usuários do *nix, este arquivo está localizado em `$SPLUNK_HOME/etc/apps/splunk_httpinput/local/`. Para usuários do Windows, está em `%SPLUNK_HOME%\etc\apps\splunk_httpinput\local\`.
- Certifique-se de que o perfil do IAM especificado no seu fluxo do Firehose tenha acesso ao bucket de backup do S3 e à função do Lambda para transformação de dados (se a transformação de dados estiver habilitada). Além disso, certifique-se de que a função do IAM tenha acesso ao grupo de CloudWatch registros e aos fluxos de registros para verificar os registros de erros. Para obter mais informações, consulte Conceder FirehoseAccess a um destino do Splunk.
- Para redirecionar os dados que foram entregues ao bucket de erros do S3 (backup do S3) de volta ao Splunk, siga as etapas mencionadas na Documentação do Splunk.
- Consulte Solução de problemas do Splunk para o Amazon Kinesis Firehose.

Solução de problemas do Snowflake

Esta seção descreve as etapas comuns de solução de problemas ao usar o Snowflake como destino

Falha na criação de fluxo do Firehose

Se a criação do stream Firehose falhar em um stream que entrega dados para um cluster Snowflake PrivateLink habilitado, isso indica que o VPCE-ID não pode ser acessado pelo Firehose. Isso pode ocorrer devido a um dos motivos a seguir:

- VPCE-ID incorreto. Confirme se não há erros tipográficos.
- O Firehose não é compatível com o Snowflake sem região na versão prévia. URLs Forneça o URL usando o Localizador de contas do Snowflake. Consulte a Documentação do Snowflake para obter mais detalhes.
- Confirme se o stream do Firehose foi criado na mesma AWS região da região do Snowflake.
- Se o problema persistir, entre em contato com o AWS suporte.

Falhas na entrega

Verifique as possibilidades a seguir se os dados não estiverem sendo entregues à sua tabela do Snowflake. Os dados de falha na entrega do Snowflake serão entregues ao bucket de erros do S3 junto com um código de erro e uma mensagem de erro que correspondem à carga útil. A seguir há

alguns cenários de erro comuns. Para ver a lista completa de códigos de erro, consulte Erros de entrega de dados do Snowflake.

- Código de erro: Snowflake. DefaultRoleMissing: indica que a função snowflake não está configurada durante a criação do stream Firehose. Se o perfil do Snowflake não estiver configurado, certifique-se de definir um perfil padrão para o usuário do Snowflake especificado.
- Código de erro: Snowflake. ExtraColumns: indica que a inserção no Snowflake foi rejeitada devido às colunas extras na carga de entrada. As colunas não presentes na tabela não devem ser especificadas. Observe que os nomes das colunas do Snowflake diferenciam maiúsculas de minúsculas. Se a entrega falhar com esse erro, apesar da coluna estar presente na tabela, certifique-se de que as letras maiúsculas e minúsculas do nome da coluna na carga útil de entrada correspondam ao nome da coluna declarado na definição da tabela.
- Código de erro: Snowflake. MissingColumns: indica que a inserção no Snowflake foi rejeitada devido à falta de colunas na carga de entrada. Certifique-se de que haja valores especificados para todas as colunas não anuláveis.
- Código de erro: Snowflake. InvalidInput: isso pode acontecer quando o Firehose falhou em analisar a carga de entrada fornecida em um formato JSON válido. Certifique-se de que a carga útil de json esteja bem formada, não tenha aspas duplas extras, aspas, caracteres de escape, etc. Atualmente, o Firehose oferece suporte a apenas um único item JSON como carga útil de registro, não havendo suporte a matrizes JSON.
- Código de erro: Snowflake. InvalidValue: indica que a entrega falhou devido ao tipo de dados incorreto na carga de entrada. Certifique-se de que os valores de JSON especificados na carga útil de entrada estejam de acordo com o tipo de dados declarado na definição da tabela do Snowflake.
- Código de erro: Snowflake. InvalidTableType: indica que o tipo de tabela configurado no stream do Firehose não é suportado. Consulte as limitações (em Limitações) do streaming de snowpipe para ver as tabelas, colunas e tipos de dados com suporte.

> **Note**
>
> Por qualquer motivo, se a definição da tabela ou as permissões do perfil forem alteradas no seu destino do Snowflake após a criação do fluxo do Firehose, poderá levar alguns minutos até que o Firehose detecte essas alterações. Se você estiver vendo erros de entrega devido a isso, tente excluir e recriar o fluxo do Firehose.

Solução de problemas de acessibilidade de endpoints do Firehose

Se a API do Firehose exceder um tempo limite, execute as etapas a seguir para testar a acessibilidade do endpoint:

- Verifique se as solicitações de API são feitas de um host em uma VPC. Todo o tráfego de uma VPC exige a configuração de um endpoint da VPC do Firehose. Para obter mais informações, consulte Usando o Firehose com. AWS PrivateLink
- Se o tráfego estiver vindo de uma rede pública ou VPC com o endpoint da VPC do Firehose configurado em uma sub-rede específica, execute os comandos a seguir no host para verificar a conectividade da rede. O endpoint do Firehose pode ser encontrado em Endpoints e cotas do Firehose.
  - Use ferramentas como traceroute ou tcping para verificar se a configuração da rede está correta. Se isso falhar, verifique sua configuração de rede:

    Por exemplo:

    ```
    traceroute firehose.us-east-2.amazonaws.com
    ```

    or

    ```
    tcping firehose.us-east-2.amazonaws.com 443
    ```

  - Se parecer que a configuração de rede está correta e o comando a seguir falhar, verifique se a Amazon CA (Autoridade Certificadora) está na cadeia de confiança.

    Por exemplo:

    ```
    curl firehose.us-east-2.amazonaws.com
    ```

  Se os comandos acima tiverem êxito, tente usar a API novamente para ver se há uma resposta retornada da API.

Solução de problemas de endpoints de HTTP

Esta seção descreve etapas comuns de solução de problemas ao lidar com o Amazon Data Firehose entregando dados para destinos genéricos de endpoints HTTP e destinos de parceiros, incluindo

Datadog, Dynatrace, LogicMonitor MongoDB, New Relic, Splunk ou Sumo Logic. Para os fins desta seção, todos os destinos aplicáveis são chamados de endpoints de HTTP. Certifique-se de que o perfil do IAM especificado no seu fluxo do Firehose tenha acesso ao bucket de backup do S3 e à função do Lambda para transformação de dados (se a transformação de dados estiver habilitada). Além disso, certifique-se de que a função do IAM tenha acesso ao grupo de CloudWatch registros e aos fluxos de registros para verificar os registros de erros. Para obter mais informações, consulte Concessão ao Firehose de acesso a um destino de endpoint de HTTP.

> **Note**
>
> As informações nesta seção não se aplicam aos seguintes destinos: Splunk, OpenSearch Service, S3 e Redshift.

CloudWatch Registros

É altamente recomendável que você habilite o CloudWatch Logging for. Os registros só são publicados quando há erros na entrega ao seu destino.

Exceções de destino

ErrorCode: HttpEndpoint.DestinationException

```
{
    "deliveryStreamARN": "arn:aws:firehose:us-east-1:123456789012:deliverystream/
ronald-test",
    "destination": "custom.firehose.endpoint.com...",
    "deliveryStreamVersionId": 1,
    "message": "The following response was received from the endpoint destination.
 413: {\"requestId\": \"43b8e724-dbac-4510-adb7-ef211c6044b9\", \"timestamp\":
 1598556019164, \"errorMessage\": \"Payload too large\"}",
    "errorCode": "HttpEndpoint.DestinationException",
    "processor": "arn:aws:lambda:us-east-1:379522611494:function:httpLambdaProcessing"
}
```

As exceções de destino indicam que o Firehose é capaz de estabelecer uma conexão com o endpoint e fazer uma solicitação HTTP, mas não recebeu um código de resposta 200. As respostas 2xx que não sejam as respostas 200 também resultarão em uma exceção de destino. O Amazon

Data Firehose registra o código de resposta e uma carga de resposta truncada recebida do endpoint configurado no Logs. CloudWatch Como o Amazon Data Firehose registra em log o código de resposta e a carga útil sem modificação ou interpretação, cabe ao endpoint fornecer o motivo exato da rejeição da solicitação do Amazon Data Firehose de entrega HTTP. Veja a seguir as recomendações de solução de problemas mais comuns para essas exceções:

- 400: indica que você está enviando uma solicitação inválida devido a uma configuração incorreta do Amazon Data Firehose. Certifique-se de ter o URL, os atributos comuns, a codificação do conteúdo, a chave de acesso e as sugestões de buffer corretos para o seu destino. Consulte a documentação específica do destino sobre a configuração necessária.
- 401: indica que a chave de acesso que você configurou para o fluxo do Firehose está incorreta ou ausente.
- 403: indica que a chave de acesso que você configurou para o fluxo do Firehose não tem permissões para entregar dados ao endpoint configurado.
- 413: indica que a carga útil da solicitação que o Amazon Data Firehose envia para o endpoint é muito grande para ser processada pelo endpoint. Tente reduzir a sugestão de buffer para o tamanho recomendado para o destino.
- 429: indica que o Amazon Data Firehose está enviando solicitações em uma taxa maior do que a capacidade do destino. Ajuste sua dica de buffer aumentando o tempo and/or de buffer, aumentando o tamanho do buffer (mas ainda dentro do limite do seu destino).
- 5xx: indica que há um problema com o destino. O serviço do Amazon Data Firehose ainda está funcionando corretamente.

> **Important**
>
> Importante: embora essas sejam as recomendações comuns de solução de problemas, endpoints específicos podem ter motivos diferentes para fornecer os códigos de resposta e as recomendações específicas do endpoint devem ser seguidas primeiro.

Resposta inválida

ErrorCode: HttpEndpoint.InvalidResponseFromDestination

```
{
```

```
    "deliveryStreamARN": "arn:aws:firehose:us-east-1:123456789012:deliverystream/
ronald-test",
    "destination": "custom.firehose.endpoint.com...",
    "deliveryStreamVersionId": 1,
    "message": "The response received from the specified endpoint is invalid.
 Contact the owner of the endpoint to resolve the issue. Response for request
 2de9e8e9-7296-47b0-bea6-9f17b133d847 is not recognized as valid JSON or has unexpected
 fields. Raw response received: 200 {\"requestId\": null}",
    "errorCode": "HttpEndpoint.InvalidResponseFromDestination",
    "processor": "arn:aws:lambda:us-east-1:379522611494:function:httpLambdaProcessing"
}
```

Exceções de respostas inválidas indicam que o Amazon Data Firehose recebeu uma resposta inválida do destino do endpoint. A resposta deve estar em conformidade com as especificações de resposta ou o Amazon Data Firehose considerará a tentativa de entrega uma falha e entregará novamente os mesmos dados até que o período configurado para novas tentativas seja excedido. O Amazon Data Firehose trata as respostas que não seguem as especificações de resposta como falhas, mesmo que a resposta tenha o status 200. Se você estiver desenvolvendo um endpoint compatível com o Amazon Data Firehose, siga as especificações de resposta para garantir que os dados sejam entregues com êxito.

Veja abaixo alguns dos tipos comuns de respostas inválidas e como corrigi-las:

- Campos JSON inválidos ou inesperados: indica que a resposta não pode ser desserializada adequadamente como JSON ou tem campos inesperados. Certifique-se de que a resposta não seja codificada por conteúdo.
- Ausente RequestId: indica que a resposta não contém um RequestiD.
- RequestId não corresponde: indica que o RequestiD na resposta não corresponde ao RequestiD de saída.
- Timestamp ausente: indica que a resposta não contém um campo de timestamp. O campo de timestamp deve ser um número e não uma string.
- Cabeçalho de tipo de conteúdo ausente: indica que a resposta não contém um cabeçalho "content-type: application/json". Nenhum outro tipo de conteúdo é aceito.

> **Important**
>
> Importante: o Amazon Data Firehose só pode entregar dados a endpoints que sigam as especificações de resposta e de solicitações do Firehose. Se você estiver configurando o

destino para um serviço de terceiros, certifique-se de usar o endpoint correto compatível com o Amazon Data Firehose, que provavelmente será diferente do endpoint público de ingestão. Por exemplo, o endpoint do Amazon Data Firehose do Datadog é `https://aws-kinesis-http-intake.logs.datadoghq.com/`, enquanto seu endpoint público é `https://api.datadoghq.com/`.

Outros erros comuns

Os códigos de erro e as definições adicionais estão listados abaixo.

- Código de erro: HttpEndpoint. RequestTimeout - Indica que o endpoint demorou mais de 3 minutos para responder. Se você for o proprietário do destino, diminua o tempo de resposta do endpoint de destino. Se você não for o proprietário do destino, entre em contato com o proprietário e pergunte se algo pode ser feito para reduzir o tempo de resposta (ou seja, diminuir a sugestão de buffer para que haja menos dados sendo processados por solicitação).
- Código de erro: HttpEndpoint. ResponseTooLarge - Indica que a resposta é muito grande. A resposta deve ter menos do que 1 MiB, incluindo cabeçalhos.
- Código de erro: HttpEndpoint. ConnectionFailed - Indica que não foi possível estabelecer uma conexão com o endpoint configurado. Isso pode ser devido a um erro de digitação no URL configurado, ao endpoint não estar acessível ao Amazon Data Firehose ou ao endpoint demorar muito para responder à solicitação de conexão.
- Código de erro: HttpEndpoint. ConnectionReset - Indica que uma conexão foi estabelecida, mas foi reiniciada ou fechada prematuramente pelo endpoint.
- Código de erro: HttpEndpoint. SSLHandshakeFalha - indica que um handshake SSL não pôde ser concluído com êxito com o endpoint configurado.

Solução de problemas do MSK como fonte

Esta seção descreve as etapas comuns de solução de problemas ao usar o MSK como fonte

> **Note**
>
> Para solucionar problemas de processamento, transformação ou entrega do S3, consulte as seções anteriores

Falha da criação do hose

Verifique o seguinte se sua mangueira com MSK As Source estiver falhando na criação:

- Verifique se o cluster do MSK de origem está no estado Ativo.
- Se você estiver usando conectividade privada, verifique se o link privado no cluster está ativado.

  Se você estiver usando a conectividade pública, verifique se o acesso público no cluster está ativado.
- Se você estiver usando conectividade privada, certifique-se de adicionar uma política baseada em recursos que permita que o Firehose crie um link privado. Consulte também: Permissões entre contas do MSK.
- Certifique-se de que a função na configuração de origem tenha permissão para ingerir dados do tópico do cluster.
- Certifique-se de que seus grupos de segurança de VPC permitam tráfego de entrada nas portas usadas pelos servidores de bootstrap do cluster.

Hose suspenso

Verifique os pontos a seguir se o hose estiver no estado SUSPENSO

- Verifique se o cluster do MSK de origem está no estado Ativo.
- Verifique se o tópico de origem existe. Caso o tópico tenha sido excluído e recriado, você também precisará excluir e recriar o fluxo do Firehose.

Hose com contrapressão

O valor de DataReadFromSource .Backpressured será 1 quando BytesPerSecondLimit cada partição for excedida ou se o fluxo normal de entrega for lento ou interrompido.

- Se você estiver pressionando BytesPerSecondLimit , verifique a métrica DataReadFromSource .Bytes e solicite um aumento de limite.
- Verifique os CloudWatch registros, as métricas de destino, as métricas de transformação de dados e as métricas de conversão de formato para identificar os gargalos.

Atualidade incorreta de dados

A atualidade dos dados parece incorreta

- O Firehose calcula a atualidade dos dados com base no timestamp do registro consumido. Para garantir que esse timestamp seja registrado corretamente quando o registro do produtor persiste nos registros do agente do Kafka, defina a configuração do tipo de timestamp do tópico Kafka como `message.timestamp.type=LogAppendTime`.

Problemas de conexão de cluster do MSK

O procedimento a seguir explica como é possível validar a conectividade com clusters do MSK. Para obter detalhes sobre a configuração do cliente do Amazon MSK, consulte Conceitos básicos de uso do Amazon MSK no Guia do desenvolvedor do Amazon Managed Streaming for Apache Kafka.

Para validar a conectividade com clusters do MSK

1. Crie uma instância AL2 Amazon EC2 baseada em UNIX (preferencialmente). Se você tiver somente a conectividade VPC habilitada em seu cluster, certifique-se de que sua EC2 instância seja executada na mesma VPC. Faça SSH para a instância quando disponível. Para obter mais informações, consulte este tutorial no Guia do EC2 usuário da Amazon.
2. Instale o Java usando o gerenciador de pacotes Yum executando o comando a seguir. Para obter mais informações, consulte as instruções de instalação no Guia do usuário do Amazon Corretto 8.

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

3. Instale o cliente da AWS executando o comando a seguir.

   ```
   curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
   unzip awscliv2.zip
   sudo ./aws/install
   ```

4. Faça o download da versão 2.6* do cliente do Apache Kafka executando o comando a seguir.

   ```
   wget https://archive.apache.org/dist/kafka/2.6.2/kafka_2.12-2.6.2.tgz
   tar -xzf kafka_2.12-2.6.2.tgz
   ```

5. Acesse o diretório `kafka_2.12-2.6.2/libs` e execute o seguinte comando para baixar o arquivo JAR do IAM do Amazon MSK.

```
wget https://github.com/aws/aws-msk-iam-auth/releases/download/v1.1.3/aws-msk-iam-auth-1.1.3-all.jar
```

6. Crie o arquivo `client.properties` na pasta bin do Kafka.

7. Substitua `awsRoleArn` pelo ARN do perfil que você usou na sua `SourceConfiguration` do Firehose e verifique a localização do certificado. Permita que seu usuário AWS cliente assuma a função`awsRoleArn`. AWS o usuário cliente tentará assumir a função que você especificou aqui.

```
[ec2-user@ip-xx-xx-xx-xx bin]$ cat client.properties
security.protocol=SASL_SSL
sasl.mechanism=AWS_MSK_IAM
sasl.jaas.config=software.amazon.msk.auth.iam.IAMLoginModule required
 awsRoleArn="<role arn>" awsStsRegion="<region name>";
sasl.client.callback.handler.class=software.amazon.msk.auth.iam.IAMClientCallbackHandler
awsDebugCreds=true
ssl.truststore.location=/usr/lib/jvm/java-1.8.0-
openjdk-1.8.0.342.b07-1.amzn2.0.1.x86_64/jre/lib/security/cacerts
ssl.truststore.password=changeit
```

8. Execute o comando do Kafka a seguir para listar tópicos. Se sua conexão for pública, use os servidores de Bootstrap de endpoints públicos. Se sua conexão for privada, use os servidores de Bootstrap de endpoints privados.

```
bin/kafka-topics.sh --list --bootstrap-server <bootstrap servers> --command-config
 bin/client.properties
```

Se a conexão obtiver êxito, você verá um resultado semelhante ao exemplo a seguir.

```
[ec2-user@ip-xx-xx-xx-xx kafka_2.12-2.6.2]$ bin/kafka-topics.sh --list --bootstrap-
server <bootstrap servers> --command-config bin/client.properties

[xxxx-xx-xx 05:49:50,877] WARN The configuration 'awsDebugCreds' was supplied but
 isn't a known config. (org.apache.kafka.clients.admin.AdminClientConfig)
[xxxx-xx-xx 05:49:50,878] WARN The configuration 'ssl.truststore.location' was
 supplied but isn't a known config.
 (org.apache.kafka.clients.admin.AdminClientConfig)
```

```
[xxxx-xx-xx 05:49:50,878] WARN The configuration 'sasl.jaas.config' was supplied
 but isn't a known config. (org.apache.kafka.clients.admin.AdminClientConfig)
[xxxx-xx-xx 05:49:50,878] WARN The configuration
 'sasl.client.callback.handler.class' was supplied but isn't a known config.
 (org.apache.kafka.clients.admin.AdminClientConfig)
[xxxx-xx-xx 05:49:50,878] WARN The configuration 'ssl.truststore.password' was
 supplied but isn't a known config.
 (org.apache.kafka.clients.admin.AdminClientConfig)
[xxxx-xx-xx 05:50:21,629] WARN [AdminClient clientId=adminclient-1] Connection to
 node...
__amazon_msk_canary
__consumer_offsets
```

9. Se você tiver algum problema ao executar o script anterior, verifique se os servidores de bootstrap fornecidos estão acessíveis na porta especificada. Para fazer isso, é possível baixar e usar o telnet ou um utilitário similar, conforme mostrado no comando a seguir.

   ```
   sudo yum install telnet
   telnet <bootstrap servers><port>
   ```

   Se o teste obtiver êxito, você verá a resultado a seguir. Isso significa que você é capaz de se conectar ao seu cluster do MSK em sua VPC local e que os servidores de bootstrap estão íntegros na porta especificada.

   ```
   Connected to ..
   ```

10. Se a solicitação não obtiver êxito, verifique as regras de entrada no grupo de segurança da sua VPC. Como exemplo, é possível usar as propriedades a seguir na regra de entrada.

    ```
    Type: All traffic
    Port: Port used by the bootstrap server (e.g. 14001)
    Source: 0.0.0.0/0
    ```

    Tente novamente a conexão telnet conforme mostrado na etapa anterior. Se você ainda não conseguir se conectar ou se sua conexão do Firehose ainda estiver falhando, entre em contato com o suporte da AWS.

Cota do Amazon Data Firehose

Esta seção descreve as cotas atuais, antes chamadas de limites, no Amazon Data Firehose. Salvo indicação em contrário, cada cota aplica-se por região.

O console Service Quotas é um local central onde você pode visualizar e gerenciar suas cotas de AWS serviços e solicitar um aumento de cota para muitos dos recursos que você usa. Use as informações de cotas que fornecemos para gerenciar sua AWS infraestrutura. Planeje a solicitação de aumentos das cotas com antecedência antes que sejam necessários.

Para obter mais informações, consulte Endpoints e cotas do Amazon Data Firehose na Referência geral da Amazon Web Services.

A seção a seguir mostra que o Amazon Data Firehose tem a cota a seguir.

- Com o Amazon MSK como fonte do fluxo do Firehose, cada fluxo do Firehose tem uma cota padrão de 10 MB/s de throughput de leitura por partição e um tamanho máximo de registro de 10 MB. É possível usar o aumento de Service Quota para solicitar um aumento da cota padrão de 10 MB/s de taxa de throughput de leitura por partição.
- Com o Amazon MSK como fonte para o stream do Firehose, há um tamanho máximo de registro de 6 MB se o AWS Lambda estiver ativado e um tamanho máximo de registro de 10 MB se o Lambda estiver desativado. AWS O Lambda limita seu registro de entrada para 6 MB, e o Amazon Data Firehose encaminha registros acima de 6 MB para um bucket S3 com erro. Se o Lambda estiver desabilitado, o Firehose limitará seu registro de entrada em 10 MB. Se o Amazon Data Firehose receber um tamanho de registro do Amazon MSK maior que 10 MB, o Amazon Data Firehose entregará esse registro ao bucket de erros do S3 e emitirá métricas do Cloudwatch para a sua conta. Para obter mais informações sobre os limites do AWS Lambda, consulte: https://docs.aws.amazon.com/lambda/ latest/dg/gettingstarted -limits.html.
- Quando o particionamento dinâmico em um fluxo do Firehose está habilitado, há uma cota padrão de 500 partições ativas que podem ser criadas para esse fluxo do Firehose. A quantidade de partições ativas é o número total de partições ativas dentro do buffer de entrega. Por exemplo, se a consulta de particionamento dinâmico monta 3 partições por segundo e você tiver uma configuração de sugestão de buffer que aciona a entrega a cada 60 segundos, então, em média, você teria 180 partições ativas. Depois que os dados são entregues em uma partição, essa partição deixa de estar ativa. É possível usar o formulário de limites do Amazon Data Firehose para solicitar um aumento dessa cota para até 5.000 partições ativas por cada fluxo do Firehose.

Se você precisar de mais partições, será possível criar mais fluxos do Firehose e distribuir as partições ativas entre eles.

- Quando o particionamento dinâmico está habilitado em um fluxo do Firehose, um throughput máximo de 1 GB por segundo é possível para cada partição ativa.
- Cada conta terá a cota a seguir para o número de fluxos do Firehose por região:
  - Leste dos EUA (Norte da Virgínia), Leste dos EUA (Ohio), Oeste dos EUA (Oregon), Europa (Irlanda), Ásia-Pacífico (Tóquio): 5.000 fluxos do Firehose
  - Europa (Frankfurt), Europa (Londres), Ásia-Pacífico (Cingapura), Ásia-Pacífico (Sydney), Ásia-Pacífico (Seul), Ásia-Pacífico (Mumbai) AWS GovCloud , (Oeste dos EUA), Canadá (Oeste), Canadá (Central): 2.000 fluxos Firehose
  - Europa (Paris), Europa (Milão), Europa (Estocolmo), Ásia-Pacífico (Hong Kong), Ásia-Pacífico (Osaka), América do Sul (São Paulo), China (Ningxia), China (Pequim), Oriente Médio (Bahrein), (Leste dos EUA), África AWS GovCloud (Cidade do Cabo): 500 fluxos Firehose
  - Europa (Zurique), Europa (Espanha), Ásia-Pacífico (Hyderabad), Ásia-Pacífico (Jacarta), Ásia-Pacífico (Melbourne), Oriente Médio (EAU), Israel (Tel Aviv), Oeste do Canadá (Calgary), Canadá (Central), Ásia-Pacífico (Malásia), Ásia-Pacífico (Tailândia), México (Central): 100 Firehose strestrees amuletos
  - Se você exceder esse número, uma chamada a CreateDeliveryStream resultará em uma exceção `LimitExceededException`. Para aumentar essa cota, é possível usar o Service Quotas, caso esteja disponíveis na sua região. Para obter mais informações sobre o uso de Service Quotas, consulte Solicitar um aumento de cota. Se as Service Quotas não estiverem disponíveis na sua região, será possível usar o formulário de limites do Amazon Data Firehose para solicitar um aumento.
- Quando o Direct PUT é configurado como fonte de dados, cada stream do Firehose fornece a seguinte cota e solicitações combinadas: PutRecordPutRecordBatch
  - Para o Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon) e Europa (Irlanda): 500.000. records/second, 2,000 requests/second, and 5 MiB/second
  - Para outros Regiões da AWS: 100.000records/second, 1,000 requests/second, and 1 MiB/second.

  Se um stream Direct PUT sofrer limitação devido a maiores volumes de ingestão de dados que excedem a capacidade de taxa de transferência de um stream do Firehose, o Amazon Data Firehose aumenta automaticamente o limite de taxa de transferência do fluxo até que a limitação seja contida. Dependendo do aumento da taxa de transferência e da limitação, pode levar mais tempo para que o Firehose aumente a taxa de transferência de um stream até os níveis desejados.

Por esse motivo, continue tentando novamente os registros de ingestão de dados com falha. Se você espera que o volume de dados aumente em grandes rajadas repentinas ou se seu novo stream precisar de uma taxa de transferência maior do que o limite de taxa de transferência padrão, solicite o aumento do limite de taxa de transferência.

Para solicitar um aumento na cota, use o formulário de limites do Amazon Data Firehose. As três cotas são escaladas proporcionalmente. Por exemplo, se você aumentar a cota de taxa de transferência no Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon) ou Europa (Irlanda) para 10. MiB/second, the other two quota increase to 4,000 requests/second and 1,000,000 records/second

> **Note**
>
> Não use limites e cotas em nível de recursos como forma de controlar o uso do serviço.

> **Important**
>
> Se a cota aumentada for muito maior do que o tráfego em execução, isso ocasionará lotes de entrega pequenos para os destinos. Isso não é eficaz e pode ser dispendioso nos serviços de destino. Certifique-se de aumentar a cota apenas para que corresponda ao tráfego atual e aumente-a ainda mais se o tráfego aumentar.

> **Important**
>
> Observe que registros de dados menores podem levar a custos mais altos. A definição de preços da ingestão do Firehose é baseada no número de registros de dados que você envia para o serviço, multiplicado pelo tamanho de cada registro, arredondado para os 5 KB (5.120 bytes) mais próximos. Portanto, para o mesmo volume de dados recebidos (bytes), se o número maior de registros recebidos for maior, o custo incorrido será maior. Por exemplo, se o volume total de dados recebidos for 5 MiB, enviar 5 MiB de dados em 5.000 registros custa mais do que enviar a mesma quantidade de dados usando 1.000 registros. Para obter mais informações, consulte Amazon Data Firehose na Calculadora de Preços da AWS.

> **Note**
>
> Quando o Kinesis Data Streams está configurado como a fonte de dados, essa cota não é aplicada, e o Amazon Data Firehose diminui ou aumenta a escala verticalmente sem nenhum limite.

- Cada stream do Firehose armazena registros de dados por até 24 horas, caso o destino da entrega não esteja disponível e a origem esteja. DirectPut Se a fonte for o Kinesis Data Streams (KDS) e o destino não estiver disponível, os dados serão retidos de acordo com a configuração do KDS.
- O tamanho máximo de um registro enviado para o Amazon Data Firehose, antes da codificação na base64, é de 1.000 KiB.
- A operação PutRecordBatch pode armazenar até 500 registros por chamada ou 4 MiB por chamada, sendo aplicável a menor opção. Essa cota não pode ser alterada.
- Cada uma das operações a seguir podem fornecer até cinco invocações por segundo , o que é um limite fixo.
  - `CreateDeliveryStream`
  - `DeleteDeliveryStream`
  - `DescribeDeliveryStream`
  - `ListDeliveryStreams`
  - `UpdateDestination`
  - `TagDeliveryStream`
  - `UntagDeliveryStream`
  - `ListTagsForDeliveryStream`
  - `StartDeliveryStreamEncryption`
  - `StopDeliveryStreamEncryption`
- As dicas de intervalo do buffer variam de 60 a 900 segundos.
- Para a entrega do Amazon Data Firehose ao Amazon Redshift, há suporte somente para clusters do Amazon Redshift.
- O intervalo de duração da nova tentativa é de 0 segundos a 7.200 segundos para o Amazon Redshift OpenSearch e a entrega de serviços.

- O Firehose oferece suporte ao Elasticsearch versões 1.5, 2.3, 5.1, 5.3, 5.5, 5.6, bem como todas as versões 6.*, 7.* e 8.*. O Firehose oferece suporte ao Amazon OpenSearch Service 2.x até 2.11.
- Quando o destino é Amazon S3, Amazon Redshift ou OpenSearch Service, o Amazon Data Firehose permite até 5 invocações Lambda pendentes por fragmento. Para o Splunk, a cota é de 10 invocações pendentes do Lambda por fragmento.
- É possível usar um CMK do tipo `CUSTOMER_MANAGED_CMK` para criptografar até 500 fluxos do Firehose.

Histórico do documento

A tabela a seguir descreve as alterações importantes feitas na documentação do Amazon Data Firehose.

Alteração	Descrição	Alterado em
Foi adicionado o suporte para a hierarquia de vários catálogos do Glue	Isso simplifica a integração do Firehose com as tabelas do Amazon S3 sem a necessidade de links de recursos entre o catálogo de dados padrão e. `S3TablesCatalog` Consulte Configurar um fluxo do Firehose para tabelas do Amazon S3.	14 de maio de 2025
Adicionada base de dados como fonte (pré-visualização pública)	Agora você pode replicar as alterações do banco de dados para tabelas do Apache Iceberg no Amazon S3. Consulte Replique as alterações do banco de dados no Apache Iceberg.	15 de novembro de 2024
Versão de disponibilidade geral (GA) para tabelas do Apache Iceberg adicionadas como destino	É possível criar um fluxo do Firehose com tabelas do Apache Iceberg como destino. Consulte Entrega de dados às tabelas do Apache Iceberg.	30 de setembro de 2024
Foram adicionados exemplos de tipos de dados	Foram adicionados exemplos de tipos de dados com suporte a tabelas do Apache Iceberg. Consulte Noções básicas sobre os tipos de dados com suporte.	22 de agosto de 2024
Lançamento de nova região	O Amazon Data Firehose está agora disponível na região Ásia-Pacífico (Malásia). Consulte Cota do Amazon Data Firehose.	22 de agosto de 2024
Foram adicionadas tabelas do Apache Iceberg como	É possível criar um fluxo do Firehose com tabelas do Apache Iceberg como destino. Consulte Entrega de dados às tabelas do Apache Iceberg.	25 de julho de 2024

Alteração	Descrição	Alterado em
destino (pré-visualização pública)		
Sugestões de armazenamento em buffer para o Snowflake	O Snowflake agora oferece suporte a sugestões de armazenamento em buffer. Consulte the section called “Definição de configurações de destino para o Snowflake”.	25 de julho de 2024
Snowflake como destino em novas regiões	O Snowflake agora está disponível como destino nas regiões Ásia-Pacífico (Singapura), Ásia-Pacífico (Seul) e Ásia-Pacífico (Sydney). Consulte the section called “Definição de configurações de destino para o Snowflake”.	25 de julho de 2024
Seções do guia do usuário reestruturadas	Navegação simplificada para seções no guia do usuário. Consulte Envio de dados a um fluxo do Firehose e Solucionar erros.	5 de julho de 2024
O Amazon Data Firehose se integra ao AWS Secrets Manager	Agora é possível acessar seus segredos e automatizar a alternância de credenciais com segurança com o Secrets Manager. Consulte the section called “Autenticação com o AWS Secrets Manager”.	6 de junho de 2024
Foi adicionado suporte para ingestão de logs para o Dynatrace	Agora é possível enviar logs e eventos para a Dynatrace para análise posterior. Consulte the section called “Definições de configurações de destino para o Dynatrace”.	18 de abril de 2024
Versão de disponibilidade geral (GA) do Snowflake como destino	O Snowflake agora está disponível ao público em geral. Consulte the section called “Definição de configurações de destino para o Snowflake”.	17 de abril de 2024

Alteração	Descrição	Alterado em
O Amazon Kinesis Data Firehose agora é conhecido como Amazon Data Firehose	O Amazon Kinesis Data Firehose foi rebatizado como Amazon Data Firehose. Consulte O que é o Amazon Data Firehose	9 de fevereiro de 2024
Foi adicionado o Snowflake como destino (pré-visualização pública)	É possível criar um fluxo do Firehose com o Snowflake como destino. Consulte the section called “Definição de configurações de destino para o Snowflake”.	19 de janeiro de 2024
Foi adicionada a descompactação automática dos registros CloudWatch	É possível ativar a descompactação em fluxos novos ou existentes para enviar CloudWatch dados descompactados do Logs para destinos do Firehose. Consulte the section called “Enviar CloudWatch registros para o Firehose”.	15 de dezembro de 2023
Adicionada a Splunk Observability Cloud como destino	É possível criar um fluxo do Firehose com a Splunk Observability Cloud como destino. Consulte the section called “Definição de configurações de destino para a Splunk Observability Cloud”.	3 de outubro de 2023
Adicionado o Amazon Managed Streaming for Apache Kafka como fonte de dados	É possível agora configurar o Amazon MSK para enviar informações para um fluxo do Firehose. Consulte the section called “Definição de configurações de fonte para o Amazon MSK”.	26 de setembro de 2023

Alteração	Descrição	Alterado em
Adicionada compatibilidade com o tipo DocumentID para destino do Serviço OpenSearch	Se OpenSearch Service for o destino do seu fluxo do Firehose, o tipo DocumentID indicará o método para configurar o ID do documento. Os métodos com suporte são ID do documento gerado pelo Firehose e OpenSearch ID do documento gerado pelo documento gerado pelo documento gerado pelo Firehose. Consulte the section called “Definição de configurações do destino”.	10 de maio de 2023
Adicionada compatibilidade com particion amento dinâmico	Foi adicionado suporte ao com particionamento dinâmico contínuo dos dados em streaming no Amazon Data Firehose. Consulte Partição de dados de streaming.	31 de agosto de 2021
Adicione um tópico sobre prefixos personalizados.	Adicionado um tópico sobre as expressões que podem ser usadas ao criar um prefixo personalizado para dados entregues ao Amazon S3. Consulte the section called “Noções básicas de prefixos personalizados para objetos do Amazon S3”.	20 de dezembro de 2018
Foi adicionado um novo tutorial do Amazon Data Firehose	Foi adicionado um tutorial que demonstra como enviar logs de fluxo da Amazon VPC ao Splunk por meio do Amazon Data Firehose. Consulte Ingestão de logs de fluxo da VPC no Splunk usando o Amazon Data Firehose.	30 de outubro de 2018
Quatro novas regiões do Amazon Data Firehose foram adicionadas	Adicionadas Paris, Mumbai, São Paulo e Londres. Para obter mais informações, consulte Cota do Amazon Data Firehose.	27 de junho de 2018
Duas novas regiões do Amazon Data Firehose foram adicionadas	Adicionadas Seul e Montreal. Para obter mais informações, consulte Cota do Amazon Data Firehose.	13 de junho de 2018

Alteração	Descrição	Alterado em
Novo recurso Kinesis Streams como fonte	Foram adicionados fluxos do Kinesis como uma fonte potencial para registros para um fluxo do Firehose. Para obter mais informações, consulte Escolha da fonte e do destino para seu fluxo do Firehose.	18 de agosto de 2017
Fazer a atualização para a documentação do console	O assistente de criação de fluxo do Firehose foi atualizado. Para obter mais informações, consulte Tutorial: Criação de um fluxo do Firehose a partir do console.	19 de julho de 2017
Nova transformação de dados	É possível configurar o Amazon Data Firehose para transformar os dados antes da entrega. Para obter mais informações, consulte Transformação de dados da fonte no Amazon Data Firehose.	19 de dezembro de 2016
Nova tentativa de COPY do Amazon Redshift	É possível configurar o Amazon Data Firehose para tentar executar novamente um comando COPY no cluster do Amazon Redshift se ele falhar. Para ter mais informações, consulte Tutorial: Criação de um fluxo do Firehose a partir do console, Noções básicas sobre entrega de dados no Amazon Data Firehose e Cota do Amazon Data Firehose.	18 de maio de 2016
Novo destino do Amazon Data Firehose, o Amazon Service OpenSearch	É possível criar um fluxo do Firehose com o Amazon OpenSearch Service como destino. Para ter mais informações, consulte Tutorial: Criação de um fluxo do Firehose a partir do console, Noções básicas sobre entrega de dados no Amazon Data Firehose e Conceder ao Firehose acesso a um destino de serviço público OpenSearch .	19 de abril de 2016
Novas CloudWatch métricas e recursos de solução de problemas aprimorados	Atualização do Monitoramento do Amazon Data Firehose e do Solução de problemas de erros no Amazon Data Firehose.	19 de abril de 2016

Alteração	Descrição	Alterado em
Novo agente do Kinesis aprimorado	Atualizado Configurar o agente do Kinesis para enviar dados.	11 de abril de 2016
Novos agentes do Kinesis	Adição do Configurar o agente do Kinesis para enviar dados.	2 de outubro de 2015
Lançamento inicial	Versão inicial do Guia do desenvolvedor do Amazon Data Firehose.	4 de outubro de 2015