aws

Guía para desarrolladores

Amazon Data Firehose

Amazon Data Firehose: Guía para desarrolladores

Table of Contents

Las traducciones son generadas a través de traducción automática. En caso de conflicto entre la traducción y la version original de inglés, prevalecerá la version en inglés.

¿Qué es Amazon Data Firehose?

Amazon Data Firehose es un servicio totalmente administrado para entregar datos de streaming en tiempo real en destinos como Amazon Simple Storage Service (Amazon S3), Amazon Redshift, Amazon OpenSearch Service, Amazon OpenSearch sin servidor, Splunk, Apache Iceberg Tables y cualquier punto de conexión HTTP personalizado o puntos de conexión HTTP de proveedores de servicios de terceros compatibles, tales como Datadog, Dynatrace, LogicMonitor, MongoDB, New Relic, Coralogix y Elastic. Con Amazon Data Firehose, no es necesario escribir aplicaciones ni administrar recursos. Configure los productores de datos para que envíen datos a Amazon Data Firehose y este los entrega automáticamente al destino que haya especificado. También puede configurar Amazon Data Firehose para transformar los datos antes de entregarlos.

Para obtener más información sobre las soluciones de macrodatos de AWS, consulte Macrodatos en AWS. Para obtener más información sobre las soluciones de datos de streaming de AWS, consulte ¿Qué son los datos de streaming?.

Conozca los conceptos clave

Al empezar a utilizar Amazon Data Firehose, es recomendable comprender los siguientes conceptos.

Flujo de Firehose

: Entidad subyacente de Amazon Data Firehose. Para usar Amazon Data Firehose, se crea un flujo de Firehose y, a continuación, se le envían datos. Para obtener más información, consulte Tutorial: Crear un flujo de Firehose desde la consola y Enviar datos a un flujo de Firehose.

Registro

: Datos de interés que el productor de datos envía a un flujo de Firehose. Cada registro puede pesar hasta 1000 KB.

Productor de datos

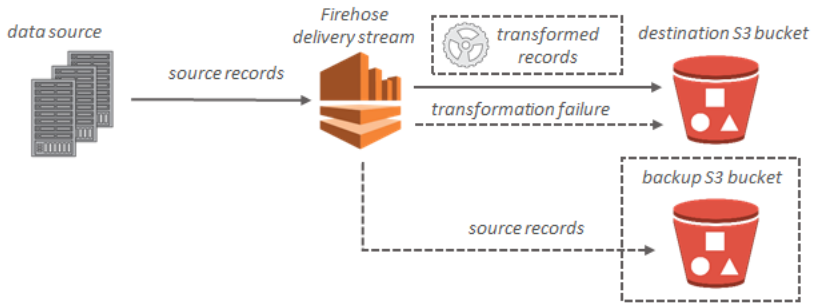
: Los productores envían los registros a los flujos de Firehose. Por ejemplo, un servidor web que envía datos de registro a un flujo de Firehose es un productor de datos. También puede configurar el flujo de Firehose para que lea automáticamente los datos de un flujo de datos de Kinesis existente y los cargue en los destinos. Para obtener más información, consulte Enviar datos a un flujo de Firehose.

Tamaño e intervalo del búfer

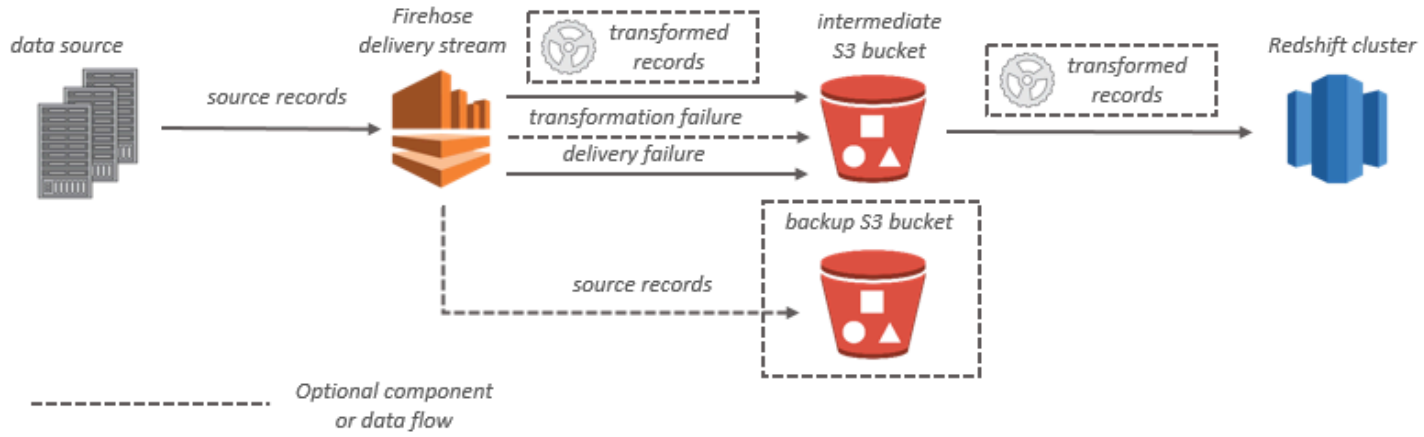
Amazon Data Firehose almacena una cantidad determinada de datos de streaming de entrada en búfer durante un periodo determinado antes de entregarlos en los destinos. Buffer Size se expresa en MB y Buffer Interval, en segundos.

Descripción del flujo de datos en Amazon Data Firehose

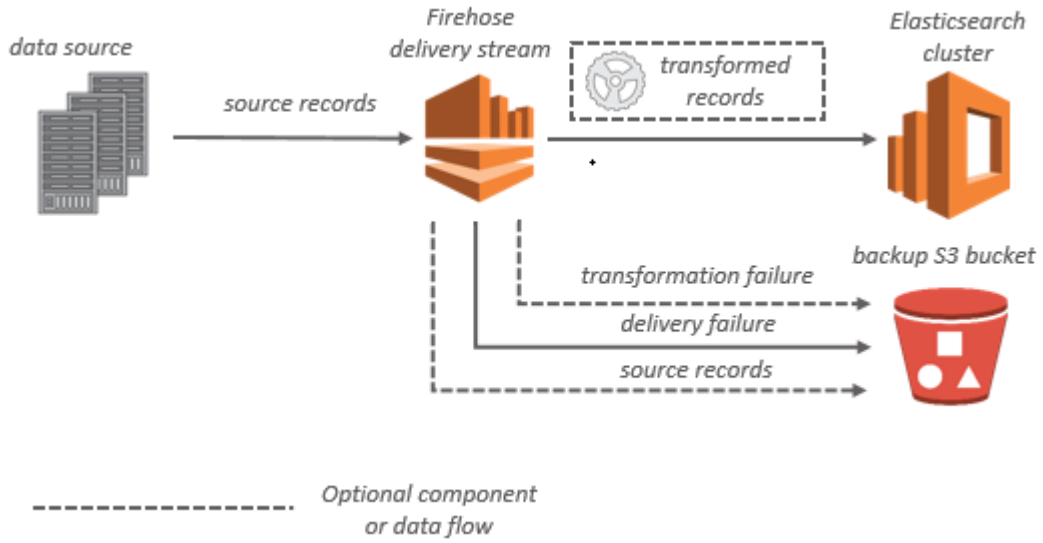
En el caso de los destinos de Amazon S3, los datos de streaming se entregan en el bucket de S3. Si habilita la transformación de datos, puede realizar una copia de seguridad de los datos de origen en otro bucket de Amazon S3.



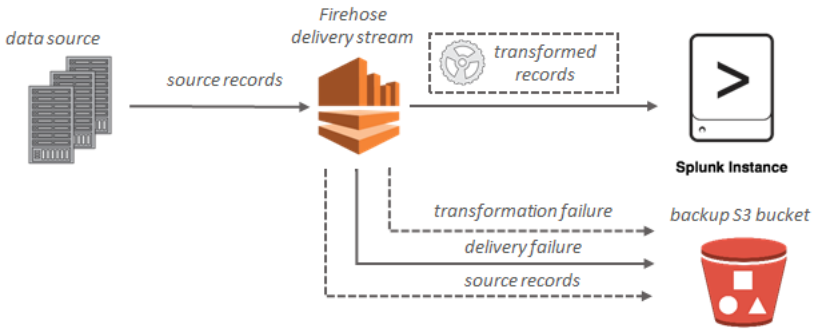
En el caso de los destinos de Amazon Redshift, los datos de streaming se entregan primero en el bucket de S3. A continuación, Amazon Data Firehose emite un comando COPY de Amazon Redshift para cargar los datos del bucket de S3 en el clúster de Amazon Redshift. Si habilita la transformación de datos, puede realizar una copia de seguridad de los datos de origen en otro bucket de Amazon S3.



En el caso de los destinos de OpenSearch Service, los datos de streaming se entregan en el clúster de OpenSearch Service y se puede hacer una copia de seguridad de ellos en el bucket de S3 simultáneamente.



Si el destino es Splunk, los datos de streaming se entregan a Splunk y se puede hacer un backup de ellos en el bucket de S3 simultáneamente.



Uso de Firehose con un SDK de AWS

Los kits de desarrollo de software (SDK) de AWS se encuentran disponibles en muchos lenguajes de programación populares. Cada SDK proporciona una API, ejemplos de código y documentación que facilitan a los desarrolladores la creación de aplicaciones en su lenguaje preferido.

Documentación de SDK	Ejemplos de código
AWS SDK para C++	AWS SDK para C++ Ejemplos de código de la

Documentación de SDK	Ejemplos de código
AWS CLI	AWS CLI Ejemplos de código de la
AWS SDK para Go	AWS SDK para Go Ejemplos de código de la
AWS SDK para Java	AWS SDK para Java Ejemplos de código de la
AWS SDK para JavaScript	AWS SDK para JavaScript Ejemplos de código de la
AWS SDK para Kotlin	AWS SDK para Kotlin Ejemplos de código de la
AWS SDK para .NET	AWS SDK para .NET Ejemplos de código de la
AWS SDK para PHP	AWS SDK para PHP Ejemplos de código de la
Herramientas de AWS para PowerShell	Herramientas de AWS para PowerShell Ejemplos de código de la
AWS SDK para Python (Boto3)	AWS SDK para Python (Boto3) Ejemplos de código de la
AWS SDK para Ruby	AWS SDK para Ruby Ejemplos de código de la
AWS SDK para Rust	AWS SDK para Rust Ejemplos de código de la
AWS SDK para SAP ABAP	AWS SDK para SAP ABAP Ejemplos de código de la
AWS SDK para Swift	AWS SDK para Swift Ejemplos de código de la

Ejemplo de disponibilidad

¿No encuentra lo que necesita? Solicite un ejemplo de código a través del enlace de Enviar comentarios que se encuentra al final de esta página.

Complete los requisitos previos para configurar Amazon Data Firehose

Antes de usar Amazon Data Firehose por primera vez, lleve a cabo las siguientes tareas.

Tareas

- Inscripción en AWS
- (Opcional) Descargar bibliotecas y herramientas

Inscripción en AWS

Al registrarse en Amazon Web Services (AWS), la cuenta de AWS se registra automáticamente en todos los servicios de AWS, incluido Amazon Data Firehose. Solo se le cobrará por los servicios que utilice.

Si ya dispone de una cuenta de AWS, pase a la siguiente tarea. Si no dispone de una cuenta de AWS, utilice el siguiente procedimiento para crear una.

Registro en una cuenta de AWS

1. Abra https://portal.aws.amazon.com/billing/signup.
2. Siga las instrucciones que se le indiquen.

   Parte del procedimiento de registro consiste en recibir una llamada telefónica o mensaje de texto e indicar un código de verificación en el teclado del teléfono.

   Al registrarse en una Cuenta de AWS, se crea un Usuario raíz de la cuenta de AWS. El usuario raíz tendrá acceso a todos los Servicios de AWS y recursos de esa cuenta. Como práctica recomendada de seguridad, asigne acceso administrativo a un usuario y utilice únicamente el usuario raíz para realizar tareas que requieren acceso de usuario raíz.

(Opcional) Descargar bibliotecas y herramientas

Las siguientes bibliotecas y herramientas lo ayudarán a trabajar con Amazon Data Firehose mediante programación y desde la línea de comandos:

- Las Operaciones de la API de Firehose es el conjunto básico de operaciones que admite Amazon Data Firehose.
- Los AWS SDK para Go, Java, .NET, Node.js, Python y Ruby incluyen soporte y muestras para Amazon Data Firehose.

  Si su versión de AWS SDK para Java no incluye muestras para Amazon Data Firehose, también puede descargar la versión más reciente de AWS SDK de GitHub.
- La AWS Command Line Interface es compatible con Amazon Data Firehose. AWS CLI permite controlar varios servicios de AWS desde la línea de comandos y automatizarlos mediante scripts.

Tutorial: Crear un flujo de Firehose desde la consola

Puedes usar el Consola de administración de AWS o un AWS SDK para crear una transmisión de Firehose con destino al destino que elijas.

Puedes actualizar la configuración de tu transmisión de Firehose en cualquier momento después de crearla, mediante la consola Amazon Data Firehose o. UpdateDestination El flujo de Firehose permanecerá en estado `Active` mientras se actualice la configuración, y podrá continuar enviando datos. La configuración actualizada suele entrar en vigor transcurridos unos minutos. El número de versión de un flujo de Firehose aumenta un valor de 1 después de actualizar la configuración. Se refleja en el nombre del objeto de Amazon S3 entregado. Para obtener más información, consulte Configuración del formato de nombres de objetos de Amazon S3.

Siga los pasos de los temas siguientes para crear un flujo de Firehose.

Temas

Elija el origen y el destino del flujo de Firehose

1. Abra la consola de Firehose en https://console.aws.amazon.com/firehose/.
2. Seleccione Crear flujo de Firehose.
3. En la página Crear flujo de Firehose, elija una fuente para su flujo de Firehose de una de las siguientes opciones.

    - Direct PUT: elija esta opción para crear un flujo de Firehose en el que las aplicaciones de los productores escriban directamente. Esta es una lista de servicios y agentes de AWS y servicios de código abierto que se integran con Direct PUT en Amazon Data Firehose. Esta lista no es exhaustiva y es posible que haya servicios adicionales que se puedan utilizar para enviar datos directamente a Firehose.

- AWS SDK
- AWS Lambda
- Registros de AWS CloudWatch
- Eventos de AWS CloudWatch
- Flujos de métricas en la nube de AWS
- AWS IoT
- AWS EventBridge
- Amazon Simple Email Service
- Amazon SNS
- Registros de ACL web de AWS WAF
- Amazon API Gateway: registros de acceso
- Amazon Pinpoint
- Registro de agente de Amazon MSK
- Registros de consultas de Amazon Route 53 Resolver
- Registros de alertas de AWS Network Firewall
- Registros de flujos de AWS Network Firewall
- Amazon ElastiCache Redis SLOWLOG
- Agente de Kinesis (Linux)
- Kinesis Tap (Windows)
- Fluentbit
- Fluentd
- Apache Nifi
- Snowflake

- Amazon Kinesis Data Streams seleccione esta opción para configurar un flujo de Firehose que utilice un flujo de datos de Kinesis como origen de datos. A continuación, puede usar Firehose para leer fácilmente los datos de un flujo de datos de Kinesis existente y cargarlos en los destinos. Para obtener más información sobre el uso de Kinesis Data Streams como origen de datos, consulte Enviar datos a un flujo de Firehose con Kinesis Data Streams.
- Amazon MSK: seleccione esta opción para configurar un flujo de Firehose que utilice Amazon MSK como origen de datos. A continuación, puede utilizar Firehose para leer fácilmente los datos de un clúster existente de Amazon MSK y cargarlos en los buckets de S3 especificados.

Para obtener más información, consulte Envío de datos a un flujo de Firehose con Amazon MSK.

4. Elija un destino para su flujo de Firehose de uno de los siguientes destinos compatibles con Firehose.

   - Amazon OpenSearch Service
   - Amazon OpenSearch sin servidor
   - Amazon Redshift
   - Amazon S3
   - Tablas de Apache Iceberg
   - Coralogix
   - Datadog
   - Dynatrace
   - Elastic
   - Punto de conexión HTTP
   - Honeycomb
   - Logic Monitor
   - Logz.io
   - MongoDB Cloud
   - New Relic
   - Splunk
   - Splunk Observability Cloud
   - Sumo Logic
   - Snowflake

5. Para el nombre del flujo de Firehose, puede usar el nombre que la consola genere para usted o añadir el flujo de Firehose que elija.

Configurar los ajustes del origen

Puede configurar los ajustes del origen en función del origen que elija para enviar la información a un flujo de Firehose desde la consola. Puede configurar los ajustes del origen para Amazon MSK y

Amazon Kinesis Data Streams como origen. No hay ninguna configuración de origen disponible para Direct PUT como origen.

Configuración de los ajustes de origen para Amazon MSK

Cuando elige Amazon MSK para enviar información a un flujo de Firehose, puede elegir entre clústeres aprovisionados por MSK y clústeres sin servidor de MSK. A continuación, puede utilizar Firehose para leer fácilmente los datos de un tema y un clúster de Amazon MSK específicos y cargarlos en el destino de S3 especificado.

En la sección Configuración de origen de la página, proporcione valores para los siguientes campos.

Conectividad de clústeres de Amazon MSK

Elija la opción Agentes de arranque privados (recomendada) o Agentes de arranque públicos en función de la configuración del clúster. Los agentes de arranque es lo que el cliente de Apache Kafka utiliza como punto de partida para conectarse al clúster. Los agentes de arranque públicos están diseñados para el acceso público desde fuera de AWS, mientras que los agentes de arranque privados están diseñados para acceder desde dentro de AWS. Para obtener más información sobre Amazon MSK, consulte Amazon Managed Streaming para Apache Kafka.

Para conectarse a un clúster de Amazon MSK aprovisionado o sin servidor a través de agentes de arranque privados, el clúster debe cumplir todos los requisitos siguientes.

- El clúster debe estar activo.
- El clúster debe tener IAM como uno de sus métodos de control de acceso.
- La conectividad privada de múltiples VPC debe estar habilitada para el método de control de acceso de IAM.
- Debe agregar a este clúster una política basada en recursos que conceda a la entidad principal del servicio Firehose el permiso para invocar la operación `CreateVpcConnection` de la API de Amazon MSK.

Para conectarse a un clúster de Amazon MSK aprovisionado a través de agentes de arranque públicos, el clúster debe cumplir todos los requisitos siguientes.

- El clúster debe estar activo.
- El clúster debe tener IAM como uno de sus métodos de control de acceso.
- El clúster debe ser de acceso público.

Cuenta del clúster de MSK

Puede elegir la cuenta en la que reside el clúster de Amazon MSK. Puede ser una de las siguientes.

- Cuenta actual: le permite ingerir datos de un clúster de MSK de la cuenta de AWS actual. Para ello, debe especificar el ARN del clúster de Amazon MSK desde el que el flujo de Firehose leerá los datos.
- Cuenta cruzada: le permite ingerir datos de un clúster de MSK de otra cuenta de AWS. Para obtener más información, consulte Entrega entre cuentas desde Amazon MSK.

Tema

Especifique el tema de Apache Kafka del que desea que su flujo de Firehose ingiera datos. No podrá actualizar este tema una vez finalizada la creación del flujo de Firehose.

> **Note**
>
> Firehose descomprime automáticamente los mensajes de Apache Kafka.

Configuración de los ajustes de origen de Amazon Kinesis Data Streams

Configure los ajustes de origen de Amazon Kinesis Data Streams de la siguiente manera para que envíe información a un flujo de Firehose.

> **Important**
>
> Si utiliza Kinesis Producer Library (KPL) para escribir datos en un flujo de datos de Kinesis, puede utilizar la agregación para combinar los registros que escriba en ese flujo de datos de Kinesis. Si después utiliza ese flujo de datos como origen del flujo de Firehose, Amazon Data Firehose desagrega los registros antes de entregarlos en el destino. Si configura el flujo de Firehose de modo que transforme los datos, Amazon Data Firehose desagrega los registros antes de entregarlos en AWS Lambda. Para obtener más información, consulte Desarrollar productores en Amazon Kinesis Data Streams con la Kinesis Producer Library y Agregación.

En la configuración de origen, elija un flujo existente en la lista de flujos de datos de Kinesis o introduzca un ARN de flujo de datos en el formato `arn:aws:kinesis:[Region]:[AccountId]:stream/[StreamName]`.

Si no tiene un flujo de datos existente, elija Create (Crear) para crear uno nuevo desde la consola de Amazon Kinesis. Es posible que necesite un rol de IAM que tenga los permisos necesarios en el flujo de Kinesis. Para obtener más información, consulte ???. Después de crear un nuevo flujo, elija el ícono de actualización para actualizar la lista Flujo de Kinesis. Si tiene un gran número de flujos, filtre la lista con Filter by name.

> **Note**
>
> Al configurar un flujo de datos de Kinesis como origen de un flujo de Firehose, las operaciones `PutRecord` y `PutRecordBatch` de Amazon Data Firehose se deshabilitan. Para agregar datos al flujo de Firehose en este caso, use las operaciones `PutRecord` y `PutRecords` de Kinesis Data Streams.

Amazon Data Firehose comienza a leer los datos desde la posición `LATEST` del flujo de Kinesis. Para obtener más información sobre las posiciones de Kinesis Data Streams, consulte GetShardIterator.

Amazon Data Firehose llama a la operación GetRecords del flujo de datos de Kinesis una vez por segundo para cada partición. Sin embargo, cuando la copia de seguridad completa está habilitada, Firehose llama a la operación `GetRecords` de los flujos de datos de Kinesis dos veces por segundo para cada partición, una para el destino de entrega principal y otra para la copia de seguridad completa.

Se pueden leer varios flujos de Firehose desde el mismo flujo de Kinesis. Otras aplicaciones de Kinesis (consumidores) pueden leer también datos del mismo flujo. Cada llamada desde un flujo de Firehose u otra aplicación consumidora cuenta al calcular la limitación controlada total de solicitudes de la partición. Para evitar las limitaciones, planee sus aplicaciones con especial cuidado. Para obtener más información sobre los límites de Kinesis Data Streams, consulte Amazon Kinesis Streams Limits.

Continúe con el siguiente paso para configurar la transformación de registros y la conversión de formato.

(Opcional) Configuración de la transformación de registros y de la conversión de formato

Configure Amazon Data Firehose para transformar y convertir los datos de sus registros.

Si elige Amazon MSK como fuente del flujo de Firehose

En la sección Transforme los registros de origen con AWS Lambda, proporcione valores para el siguiente campo.

1. Transformación de datos

   Para crear un flujo de Firehose que no transforme los datos de entrada, no marque la casilla Habilitar la transformación de datos.

   Para especificar una función de Lambda que Firehose pueda invocar y utilizar para transformar los datos de entrada antes de entregarlos, marque la casilla Habilitar la transformación de datos. Puede configurar una nueva función de Lambda con uno de los esquemas de Lambda o puede elegir una función de Lambda existente. La función de Lambda debe contener el modelo de estados que necesita Firehose. Para obtener más información, consulte Transformación de los datos de origen en Amazon Data Firehose.
2. En la sección Convert record format (Convertir formato de registros), proporcione valores para el siguiente campo:

   Record format conversion (Conversión del formato de registros

   Para crear un flujo de Firehose que no convierta el formato de los registros de datos entrantes, elija Deshabilitada.

   Para convertir el formato de los registros entrantes, elija Enabled (Habilitada) y, a continuación, especifique el formato de salida que desee. Debe especificar una tabla de AWS Glue que contenga el esquema que desee que Firehose utilice para convertir el formato de sus registros. Para obtener más información, consulte Conversión del formato de los datos de entrada.

   Para ver un ejemplo de cómo configurar la conversión de formatos de registros con CloudFormation, consulte AWS::KinesisFirehose::DeliveryStream.

Si elige Amazon Kinesis Data Streams o Direct PUT como origen del flujo de Firehose

En la sección Configuración de origen, proporcione valores para los siguientes campos.

1. En Transformar registros, elija una de las siguientes opciones:

a. Si su destino es Amazon S3 o Splunk, en la sección Descomprimir los registros de origen de Registros de Amazon CloudWatch, seleccione Activar la descompresión.

b. En la sección Transforme los registros de origen con AWS Lambda, proporcione valores para el siguiente campo:

   Transformación de datos

   Para crear un flujo de Firehose que no transforme los datos de entrada, no marque la casilla Habilitar la transformación de datos.

   Para especificar una función de Lambda que Amazon Data Firehose pueda invocar y utilizar para transformar los datos de entrada antes de entregarlos, marque la casilla Habilitar la transformación de datos. Puede configurar una nueva función de Lambda con uno de los esquemas de Lambda o puede elegir una función de Lambda existente. La función de Lambda debe contener el modelo de estados que necesita Amazon Data Firehose. Para obtener más información, consulte Transformación de los datos de origen en Amazon Data Firehose.

2. En la sección Convert record format (Convertir formato de registros), proporcione valores para el siguiente campo:

   Record format conversion (Conversión del formato de registros

   Para crear un flujo de Firehose que no convierta el formato de los registros de datos entrantes, elija Deshabilitada.

   Para convertir el formato de los registros entrantes, elija Enabled (Habilitada) y, a continuación, especifique el formato de salida que desee. Debe especificar una tabla de AWS Glue que contenga el esquema que desee que Amazon Data Firehose utilice para convertir el formato de sus registros. Para obtener más información, consulte Conversión del formato de los datos de entrada.

   Para ver un ejemplo de cómo configurar la conversión de formatos de registros con CloudFormation, consulte AWS::KinesisFirehose::DeliveryStream.

Configuración de los ajustes de destino

En esta sección, se describen los ajustes que debe configurar para su flujo de Firehose en función del destino que seleccione.

Temas

Configuración de los ajustes de destino de Amazon S3

Debe especificar la siguiente configuración para poder utilizar Amazon S3 como destino del flujo de Firehose.

- Escriba valores en los siguientes campos:

Bucket de S3

Seleccione el bucket de S3 de su propiedad adonde se deben entregar los datos de streaming. Puede crear un bucket de S3 nuevo o elegir uno disponible.

Delimitador de nueva línea

Puede configurar el flujo de Firehose para agregar un delimitador de nueva línea entre los registros de los objetos que se entregan en Amazon S3. Para ello, elija Habilitado. Para no agregar un delimitador de nueva línea entre los registros de los objetos que se entregan en Amazon S3, seleccione Deshabilitado. Si planea usar Athena para consultar objetos de S3 con registros agregados, active esta opción.

Particionamiento dinámico

Seleccione Habilitado para habilitar y configurar el particionamiento dinámico.

Desagregación de varios registros

Se trata del proceso de analizar los registros del flujo de Firehose y separarlos en función de un JSON válido o del delimitador de nueva línea especificado.

Si agregas varios eventos, registros o registros en una sola PutRecord llamada a la PutRecordBatch API, aún puedes habilitar y configurar la partición dinámica. Con los datos agregados, al habilitar el particionamiento dinámico, Amazon Data Firehose analiza los registros y busca varios objetos JSON válidos en cada llamada a la API. Cuando el flujo de Firehose se configura con Kinesis Data Streams como origen, también puede utilizar la agregación integrada en Kinesis Producer Library (KPL). La funcionalidad de partición de datos se ejecuta después de desagregar los datos. Por lo tanto, cada registro de cada llamada a la API se puede entregar en distintos prefijos de Amazon S3. También puede utilizar la integración de la función de Lambda para realizar cualquier otra desagregación o cualquier otra transformación antes de la funcionalidad de particionamiento de datos.

> **Important**
>
> Si sus datos están agregados, el particionamiento dinámico solo se puede aplicar una vez completada la desagregación de los datos. Por lo tanto, si habilita el particionamiento dinámico en sus datos agregados, debe seleccionar Habilitado para habilitar la desagregación de varios registros.

El flujo de Firehose completa los siguientes pasos de procesamiento en el siguiente orden: desagregación de KPL (protobuf), desagregación de JSON o delimitador, procesamiento de Lambda, particionamiento de datos, conversión de formato de datos y entrega en Amazon S3.

Tipo de desagregación de varios registros

Si ha habilitado la desagregación de varios registros, debe especificar el método para que Firehose desagregue los datos. Utilice el menú desplegable para elegir JSON o Delimitado.

Análisis en línea

Este es uno de los mecanismos admitidos para particionar dinámicamente los datos vinculados a Amazon S3. A fin de usar el análisis en línea para el particionamiento dinámico de sus datos, debe especificar los parámetros de registro de datos que se utilizarán como claves de particionamiento y proporcionar un valor para cada clave de particionamiento especificada. Seleccione Habilitado para habilitar y configurar el análisis en línea.

> **Important**
>
> Si especificó una función AWS Lambda en los pasos anteriores para transformar los registros fuente, puede utilizarla para particionar dinámicamente los datos enlazados a S3 y seguir creando las claves de partición con el análisis en línea. Con el particionamiento dinámico, puede utilizar el análisis en línea o la función AWS Lambda para crear las claves de particionamiento. O bien, puede usar el análisis en línea y la función AWS Lambda al mismo tiempo para crear las claves de partición.

Claves de particionamiento dinámico

Puede usar los campos Clave y Valor para especificar los parámetros de registro de datos que se utilizarán como claves de particionamiento dinámico y las consultas jq para generar valores de claves de particionamiento dinámico. Firehose solo admite jq 1.6. Puede especificar hasta 50 claves de particionamiento dinámico. Debe ingresar expresiones jq válidas para los valores de las claves de particionamiento dinámico a fin de configurar correctamente el particionamiento dinámico para su flujo de Firehose.

Prefijo de bucket de S3

Al habilitar y configurar el particionamiento dinámico, debe especificar los prefijos de buckets de S3 en los que Amazon Data Firehose entregará los datos particionados.

Para que el particionamiento dinámico se configure correctamente, el número de prefijos de bucket de S3 debe ser idéntico al número de claves de particionamiento especificadas.

Puede particionar los datos de origen con el análisis en línea o con la función Lambda AWS que especifique. Si especificó una función de AWS Lambda para crear claves de partición para los datos de origen, debe escribir manualmente los valores del prefijo del bucket S3 con el siguiente formato: "Lambda:keyID». partitionKeyFrom Si utiliza el análisis en línea para especificar las claves de partición para sus datos de origen, puede escribir manualmente los valores de vista previa del bucket de S3 con el siguiente formato: «partitionKeyFromquery:keyID» o puede elegir el botón Aplicar claves de partición dinámica para usar sus pares de particiones dinámicas para generar automáticamente los prefijos de su bucket de S3. key/value Al particionar sus datos con análisis en línea o AWS Lambda, también puede usar las siguientes formas de expresión en el prefijo de su bucket de S3:! {namespace:value}, donde el espacio de nombres puede ser Query o Lambda. partitionKeyFrom partitionKeyFrom

Zona horaria del prefijo del resultado del bucket S3 y el error de S3

Elija la zona horaria que desee usar para la fecha y la hora en los prefijos personalizados de los objetos de Amazon S3. De forma predeterminada, Firehose añade un prefijo de hora en UTC. Puede cambiar la zona horaria utilizada en los prefijos de S3 si desea utilizar una zona horaria diferente.

Sugerencias de almacenamiento en búfer

Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía de un proveedor de servicios a otro.

Compresión en S3

Elija la compresión de datos GZIP, Snappy, Zip o Snappy compatible con Hadoop, o sin compresión de datos. Las compresiones Snappy, Zip y Snappy compatible con Hadoop no están disponibles para los flujos de Firehose con Amazon Redshift como destino.

Formato de extensión de archivo de S3 (opcional)

Especifique un formato de extensión de archivo para los objetos entregados al bucket de destino de Amazon S3. Si habilita esta característica, la extensión de archivo especificada anulará las extensiones de archivo predeterminadas incorporadas por las funciones de conversión de formato de datos o de compresión en S3, como .parquet o .gz. Asegúrese de haber configurado la extensión de archivo correcta cuando utilice esta característica con la conversión de formato de datos o la compresión en S3. La extensión del archivo debe empezar con un punto (.) y puede contener los caracteres permitidos: 0-9a-z!-_.*‘(). La extensión del archivo no puede superar los 128 caracteres.

Cifrado de S3

Firehose admite el cifrado del lado del servidor de Amazon S3 con AWS Key Management Service (SSE-KMS) para cifrar los datos entregados en Amazon S3. Puede optar por utilizar el tipo de cifrado predeterminado especificado en el depósito S3 de destino o cifrar con una clave de la lista de claves de su propiedad. AWS KMS Si cifra los datos con AWS KMS claves, puede usar la clave AWS administrada predeterminada (aws/s3) o una clave administrada por el cliente. Para obtener más información, consulte Protección de datos mediante el cifrado del lado del servidor con claves administradas por AWS KMS (SSE-KMS).

Configuración de los ajustes de destino de las tablas de Apache Iceberg

Firehose admite las tablas Apache Iceberg como destino en todas las regiones, excepto en Regiones de AWSChina AWS GovCloud (US) Regions, y en Asia Pacífico (Malasia).

Para obtener más información sobre las tablas de Apache Iceberg como destino, consulte Entrega de datos a tablas de Apache Iceberg con Amazon Data Firehose.

Configuración de las opciones de destino de Amazon Redshift

En esta sección, se describe la configuración para usar Amazon Redshift como destino del flujo de Firehose.

Elija uno de los siguientes procedimientos en función de si tiene un clúster aprovisionado de Amazon Redshift o un grupo de trabajo de Amazon Redshift sin servidor.

- Clúster aprovisionado de Amazon Redshift

- Configuración de las opciones de destino del grupo de trabajo de Amazon Redshift sin servidor

> **Note**
>
> Firehose no puede escribir en los clústeres de Amazon Redshift que utilizan el enrutamiento de VPC mejorado.

Clúster aprovisionado de Amazon Redshift

En esta sección, se describe la configuración para usar el clúster aprovisionado de Amazon Redshift como destino del flujo de Firehose.

- Escriba valores en los siguientes campos:

  Clúster

  Clúster de Amazon Redshift en el que se copian los datos del bucket de S3. Configure el clúster de Amazon Redshift para que sea accesible al público y desbloquee direcciones IP de Amazon Data Firehose. Para obtener más información, consulte Concesión a Firehose de acceso a un destino de Amazon Redshift .

  Autenticación

  Puede elegir entre introducirlo username/password directamente o recuperar el secreto AWS Secrets Manager para acceder al clúster de Amazon Redshift.

  - Nombre de usuario

    Especifique un usuario de Amazon Redshift con permisos para acceder al clúster de Amazon Redshift. Este usuario debe tener el permiso `INSERT` de Amazon Redshift para copiar datos del bucket de S3 en el clúster de Amazon Redshift.

  - Contraseña

    Especifique la contraseña del usuario con permisos para obtener acceso al clúster.

  - Secret

    Seleccione un secreto AWS Secrets Manager que contenga las credenciales del clúster de Amazon Redshift. Si no ve su secreto en la lista desplegable, cree uno en AWS Secrets

Manager para sus credenciales de Amazon Redshift. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

Base de datos

Base de datos de Amazon Redshift en la que se copian los datos.

Tabla

Tabla de Amazon Redshift en la que se copian los datos.

Columnas

Las columnas específicas de la tabla donde se copian los datos (opcional). Utilice esta opción si la cantidad de columnas definida en los objetos de Amazon S3 es menor que la cantidad de columnas en la tabla Amazon Redshift.

Destino de S3 intermedio

Firehose primero entrega los datos en el bucket de S3 y, a continuación, emite un comando COPY de Amazon Redshift para cargar los datos en el clúster de Amazon Redshift. Especifique el bucket de S3 de su propiedad adonde se deben entregar los datos de streaming. Cree un nuevo bucket de S3 o elija uno que le pertenezca.

Firehose no elimina los datos de su bucket de S3 después de cargarlos en el clúster de Amazon Redshift. Puede administrar los datos en el bucket de S3 utilizando una configuración del ciclo de vida. Para obtener más información, consulte Administración del ciclo de vida de los objetos en la Guía del usuario de Amazon Simple Storage Service.

Prefijo de S3 intermedio

(Opcional) Para utilizar el prefijo predeterminado de los objetos de Amazon S3, deje esta opción en blanco. Firehose utiliza automáticamente un prefijo en formato de tiempo “YYYY/MM/dd/HH” UTC para objetos de Amazon S3 entregados. Puede añadir más elementos al comienzo de este prefijo. Para obtener más información, consulte Configuración del formato de nombres de objetos de Amazon S3.

Opciones de COPY

Parámetros que puede especificar en el comando COPY de Amazon Redshift. Podrían ser necesarios para la configuración. Por ejemplo, se requiere `GZIP` «» si la compresión de datos de Amazon S3 está habilitada. Se requiere `REGION` «» si el bucket de S3 no se encuentra en la misma AWS región que el clúster de Amazon Redshift. Para más

información, consulte COPY en la Guía para desarrolladores de bases de datos de Amazon Redshift.

COPY command

Comando COPY de Amazon Redshift. Para más información, consulte COPY en la Guía para desarrolladores de bases de datos de Amazon Redshift.

Retry duration

Tiempo (entre 0 y 7200 segundos) para que Firehose vuelva a intentarlo si falla el comando COPY sobre los datos en el clúster de Amazon Redshift. Firehose hace reintentos cada 5 minutos hasta que finaliza el tiempo de reintento. Si establece el tiempo de reintento en 0 (cero) segundos, Firehose no lo reintenta tras producirse un error en el comando COPY.

Sugerencias de almacenamiento en búfer

Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía de un proveedor de servicios a otro.

Compresión en S3

Elija la compresión de datos GZIP, Snappy, Zip o Snappy compatible con Hadoop, o sin compresión de datos. Las compresiones Snappy, Zip y Snappy compatible con Hadoop no están disponibles para los flujos de Firehose con Amazon Redshift como destino.

Formato de extensión de archivo de S3 (opcional)

Formato de extensión de archivo S3 (opcional): especifique un formato de extensión de archivo para los objetos entregados al bucket de destino de Amazon S3. Si habilita esta característica, la extensión de archivo especificada anulará las extensiones de archivo predeterminadas incorporadas por las funciones de conversión de formato de datos o de compresión en S3, como .parquet o .gz. Asegúrese de haber configurado la extensión de archivo correcta cuando utilice esta característica con la conversión de formato de datos o la compresión en S3. La extensión del archivo debe empezar con un punto (.) y puede contener los caracteres permitidos: 0-9a-z!-_.*‘(). La extensión del archivo no puede superar los 128 caracteres.

Cifrado de S3

Firehose admite el cifrado del lado del servidor de Amazon S3 con AWS Key Management Service (SSE-KMS) para cifrar los datos entregados en Amazon S3. Puede optar por utilizar

el tipo de cifrado predeterminado especificado en el depósito S3 de destino o cifrar con una clave de la lista de claves de su propiedad. AWS KMS Si cifra los datos con AWS KMS claves, puede usar la clave AWS administrada predeterminada (aws/s3) o una clave administrada por el cliente. Para obtener más información, consulte Protección de datos mediante el cifrado del lado del servidor con claves administradas por AWS KMS (SSE-KMS).

Configuración de las opciones de destino del grupo de trabajo de Amazon Redshift sin servidor

En esta sección, se describe la configuración para usar el grupo de trabajo de Amazon Redshift sin servidor como destino del flujo de Firehose.

- Escriba valores en los siguientes campos:

  Nombre del grupo de trabajo

  El grupo de trabajo de Amazon Redshift sin servidor en el que se copian los datos del bucket de S3. Configure el grupo de trabajo de Amazon Redshift sin servidor para que sea accesible al público y desbloquee las direcciones IP de Firehose. Para obtener más información, consulte la sección Conectarse a una instancia de Amazon Redshift sin servidor accesible públicamente en Conexión a Amazon Redshift sin servidor y también Concesión a Firehose de acceso a un destino de Amazon Redshift .

  Autenticación

  Puede elegir entre introducirlo username/password directamente o recuperar el secreto para acceder AWS Secrets Manager al grupo de trabajo Amazon Redshift Serverless.

  - Nombre de usuario

    Especifique el usuario de Amazon Redshift con permisos para acceder al grupo de trabajo de Amazon Redshift sin servidor. Este usuario debe tener el permiso `INSERT` de Amazon Redshift para copiar datos del bucket de S3 en el grupo de trabajo de Amazon Redshift sin servidor.

  - Contraseña

    Especifique la contraseña del usuario que tiene permisos para acceder al grupo de trabajo de Amazon Redshift sin servidor.

- Secret

  Seleccione un secreto AWS Secrets Manager que contenga las credenciales del grupo de trabajo Amazon Redshift Serverless. Si no ve su secreto en la lista desplegable, cree uno en AWS Secrets Manager para sus credenciales de Amazon Redshift. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

Base de datos

Base de datos de Amazon Redshift en la que se copian los datos.

Tabla

Tabla de Amazon Redshift en la que se copian los datos.

Columnas

Las columnas específicas de la tabla donde se copian los datos (opcional). Utilice esta opción si la cantidad de columnas definida en los objetos de Amazon S3 es menor que la cantidad de columnas en la tabla Amazon Redshift.

Destino de S3 intermedio

Amazon Data Firehose primero entrega los datos en el bucket de S3 y, a continuación, emite un comando COPY de Amazon Redshift para cargar los datos en el grupo de trabajo de Amazon Redshift sin servidor. Especifique el bucket de S3 de su propiedad adonde se deben entregar los datos de streaming. Cree un nuevo bucket de S3 o elija uno que le pertenezca.

Firehose no elimina los datos de su bucket de S3 después de cargarlos en el grupo de trabajo de Amazon Redshift sin servidor. Puede administrar los datos en el bucket de S3 utilizando una configuración del ciclo de vida. Para obtener más información, consulte Administración del ciclo de vida de los objetos en la Guía del usuario de Amazon Simple Storage Service.

Prefijo de S3 intermedio

(Opcional) Para utilizar el prefijo predeterminado de los objetos de Amazon S3, deje esta opción en blanco. Firehose utiliza automáticamente un prefijo en formato de tiempo “YYYY/MM/dd/HH” UTC para objetos de Amazon S3 entregados. Puede añadir más elementos al comienzo de este prefijo. Para obtener más información, consulte Configuración del formato de nombres de objetos de Amazon S3.

Opciones de COPY

Parámetros que puede especificar en el comando COPY de Amazon Redshift. Podrían ser necesarios para la configuración. Por ejemplo, se requiere `GZIP` «» si la compresión de datos de Amazon S3 está habilitada. Se requiere `REGION` «» si su bucket de S3 no está en la misma AWS región que su grupo de trabajo Amazon Redshift Serverless. Para más información, consulte COPY en la Guía para desarrolladores de bases de datos de Amazon Redshift.

COPY command

Comando COPY de Amazon Redshift. Para más información, consulte COPY en la Guía para desarrolladores de bases de datos de Amazon Redshift.

Retry duration

Tiempo (entre 0 y 7200 segundos) para que Firehose vuelva a intentarlo si falla el comando COPY sobre los datos en el grupo de trabajo de Amazon Redshift sin servidor. Firehose hace reintentos cada 5 minutos hasta que finaliza el tiempo de reintento. Si establece el tiempo de reintento en 0 (cero) segundos, Firehose no lo reintenta tras producirse un error en el comando COPY.

Sugerencias de almacenamiento en búfer

Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía de un proveedor de servicios a otro.

Compresión en S3

Elija la compresión de datos GZIP, Snappy, Zip o Snappy compatible con Hadoop, o sin compresión de datos. Las compresiones Snappy, Zip y Snappy compatible con Hadoop no están disponibles para los flujos de Firehose con Amazon Redshift como destino.

Formato de extensión de archivo de S3 (opcional)

Formato de extensión de archivo S3 (opcional): especifique un formato de extensión de archivo para los objetos entregados al bucket de destino de Amazon S3. Si habilita esta característica, la extensión de archivo especificada anulará las extensiones de archivo predeterminadas incorporadas por las funciones de conversión de formato de datos o de compresión en S3, como .parquet o .gz. Asegúrese de haber configurado la extensión de archivo correcta cuando utilice esta característica con la conversión de formato de datos

o la compresión en S3. La extensión del archivo debe empezar con un punto (.) y puede contener los caracteres permitidos: 0-9a-z!-_.*‘(). La extensión del archivo no puede superar los 128 caracteres.

Cifrado de S3

Firehose admite el cifrado del lado del servidor de Amazon S3 con AWS Key Management Service (SSE-KMS) para cifrar los datos entregados en Amazon S3. Puede optar por utilizar el tipo de cifrado predeterminado especificado en el depósito S3 de destino o cifrar con una clave de la lista de claves de su propiedad. AWS KMS Si cifra los datos con AWS KMS claves, puede usar la clave AWS administrada predeterminada (aws/s3) o una clave administrada por el cliente. Para obtener más información, consulte Protección de datos mediante el cifrado del lado del servidor con claves administradas por AWS KMS (SSE-KMS).

OpenSearch Configure los ajustes de destino del servicio

Firehose admite las versiones de Elasticsearch 1.5, 2.3, 5.1, 5.3, 5.5, 5.6, así como todas las versiones 6.*, 7.* y 8.*. Firehose es compatible con Amazon OpenSearch Service desde la versión 2.x hasta la 2.11.

En esta sección, se describen las opciones para usar el OpenSearch Servicio en tu destino.

- Escriba valores en los siguientes campos:

  OpenSearch Dominio de servicio

  El dominio de OpenSearch servicio al que se envían sus datos.

  Índice

  El nombre del índice de OpenSearch servicios que se utilizará al indexar los datos en su clúster OpenSearch de servicios.

  Index rotation

  Elija si se debe rotar el índice OpenSearch de servicios y con qué frecuencia. Si la rotación de índices está habilitada, Amazon Data Firehose agrega la marca de tiempo correspondiente al nombre del índice especificado y lo rota. Para obtener más información, consulte Configuración de la rotación de indexación para OpenSearch Service.

Tipo

El nombre del tipo de OpenSearch servicio que se utilizará al indexar los datos en su clúster de OpenSearch servicios. Para Elasticsearch 7.x y OpenSearch 1.x, solo puede haber un tipo por índice. Si intenta especificar un tipo nuevo para un índice existente que ya tiene otro tipo, Firehose devuelve un error durante el tiempo de ejecución.

Para Elasticsearch 7.x, deje este campo vacío.

Retry duration

Tiempo que tarda Firehose en volver a intentarlo si se produce un error en una solicitud de indexación. OpenSearch Como duración del reintento, se puede establecer cualquier valor entre 0 y 7200 segundos. El valor de predeterminado para la duración de reintento es de 300 segundos. Firehose reintentará varias veces con una retirada exponencial hasta que se agote el tiempo de reintento.

Una vez transcurrido el tiempo de reintento, Firehose envía los datos a la cola de mensajes fallidos (DLQ), un bucket de errores de S3 configurado. En el caso de los datos que se envían a DLQ, hay que volver a llevarlos del depósito de errores de S3 configurado al destino. OpenSearch

Si quieres impedir que Firehose Stream entregue datos a DLQ debido a un tiempo de inactividad o al mantenimiento de OpenSearch los clústeres, puedes configurar la duración del reintento con un valor superior en segundos. Para aumentar el valor de la duración del reintento por encima de los 7200 segundos, comuníquese con el servicio de asistencia de AWS .

Tipo de DocumentID

Indica el método para configurar el ID de documento. Los métodos admitidos son el ID de documento generado por FireHose y el ID de documento generado por el OpenSearch servicio. El identificador del documento generado por Firehose es la opción predeterminada cuando el valor del identificador del documento no está establecido. OpenSearch La opción recomendada es el identificador de documento generado por el servicio, ya que permite realizar operaciones de escritura intensiva, como el análisis y la observabilidad de los registros, por lo que consume menos recursos de CPU en el dominio del OpenSearch servicio y, por lo tanto, mejora el rendimiento.

Destination VPC connectivity (Conectividad de VPC de destino)

Si su dominio OpenSearch de servicio está en una VPC privada, utilice esta sección para especificar esa VPC. Especifique también las subredes y subgrupos que desea que Amazon Data Firehose utilice cuando envíe datos a su dominio de servicio. OpenSearch Puede usar los mismos grupos de seguridad que usa el dominio del OpenSearch servicio. Si especifica diferentes grupos de seguridad, asegúrese de que permitan el tráfico HTTPS saliente al grupo de seguridad del dominio del OpenSearch servicio. Asegúrese también de que el grupo de seguridad del dominio de OpenSearch servicio permita el tráfico HTTPS desde los grupos de seguridad que especificó al configurar la transmisión de Firehose. Si utilizas el mismo grupo de seguridad tanto para la transmisión de Firehose como para el dominio de OpenSearch servicio, asegúrate de que la regla de entrada del grupo de seguridad permita el tráfico HTTPS. Para obtener más información acerca de las reglas de los grupos de seguridad, consulte Reglas del grupo de seguridad en la documentación de Amazon VPC.

> **Important**
>
> Cuando especifique subredes para entregar datos al destino en una VPC privada, asegúrese de tener una cantidad suficiente de direcciones IP libres en las subredes elegidas. Si no hay una dirección IP libre disponible en una subred específica, Firehose no podrá crear ni ENIs añadir para la entrega de datos en la VPC privada, y la entrega se degradará o fallará.

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía de un proveedor de servicios a otro.

Configure los ajustes de destino para Serverless OpenSearch

En esta sección se describen las opciones para usar OpenSearch Serverless como destino.

- Escriba valores en los siguientes campos:

OpenSearch Colección Serverless

El punto final de un grupo de índices OpenSearch sin servidor al que se envían los datos.

Índice

El nombre del índice OpenSearch Serverless que se utilizará al indexar los datos para su colección Serverless. OpenSearch

Destination VPC connectivity (Conectividad de VPC de destino)

Si su colección OpenSearch sin servidor está en una VPC privada, utilice esta sección para especificar esa VPC. Especifique también las subredes y subgrupos que desea que Amazon Data Firehose utilice cuando envíe datos a su colección Serverless. OpenSearch

> **Important**
>
> Cuando especifique subredes para entregar datos al destino en una VPC privada, asegúrese de tener una cantidad suficiente de direcciones IP libres en las subredes elegidas. Si no hay una dirección IP libre disponible en una subred específica, Firehose no podrá crear ni ENIs añadir para la entrega de datos en la VPC privada, y la entrega se degradará o fallará.

Retry duration

Tiempo que tarda Firehose en volver a intentarlo si se produce un error en una solicitud de índice a OpenSearch Serverless. Como duración del reintento, se puede establecer cualquier valor entre 0 y 7200 segundos. El valor de predeterminado para la duración de reintento es de 300 segundos. Firehose reintentará varias veces con una retirada exponencial hasta que se agote el tiempo de reintento.

Una vez transcurrido el tiempo de reintento, Firehose envía los datos a la cola de mensajes fallidos (DLQ), un bucket de errores de S3 configurado. En el caso de los datos que se envían a DLQ, hay que volver a llevarlos del depósito de errores de S3 configurado a un destino sin servidor. OpenSearch

Si quieres impedir que Firehose Stream entregue datos a DLQ debido a un tiempo de inactividad o al mantenimiento de los clústeres OpenSearch sin servidor, puedes configurar la duración del reintento con un valor superior en segundos. Para aumentar el valor de la

duración del reintento por encima de los 7200 segundos, comuníquese con el servicio de asistencia de AWS .

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía de un proveedor de servicios a otro.

Configuración de los ajustes de destino del punto de conexión HTTP

En esta sección se describen las opciones de uso de un punto de conexión HTTP como destino.

> **Important**
>
> Si elige un punto de conexión HTTP como destino, revise y siga las instrucciones que se ofrecen en Comprender las especificaciones de solicitudes y respuestas de entrega de puntos de conexión HTTP.

- Proporcione valores para los siguientes campos:

  Nombre del punto de conexión HTTP: opcional

  Especifique un nombre fácil de recordar para el punto de conexión HTTP. Por ejemplo, `My HTTP Endpoint Destination`.

  URL del punto de conexión HTTP

  Especifique la URL del punto de conexión HTTP en el siguiente formato: `https://xyz.httpendpoint.com`. La URL debe ser una URL HTTPS.

  Autenticación

  Puedes elegir entre introducir la clave de acceso directamente o recuperar el secreto para acceder AWS Secrets Manager al punto final HTTP.

  - (Opcional) Clave de acceso

    Póngase en contacto con el propietario del punto de conexión para obtener la clave de acceso a fin de permitir la entrega de datos en su punto de conexión desde Firehose.

- Secret

  Seleccione un secreto AWS Secrets Manager que contenga la clave de acceso del punto final HTTP. Si no ve su secreto en la lista desplegable, cree uno AWS Secrets Manager para la clave de acceso. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

Codificación de contenidos

Amazon Data Firehose utiliza la codificación de contenidos para comprimir el cuerpo de una solicitud antes de enviarla al destino. Selecciona GZIP o Desactivado para codificar enable/disable el contenido de tu solicitud.

Retry duration

Especifique durante cuánto tiempo Amazon Data Firehose reintenta el envío de datos al punto de conexión HTTP seleccionado.

Tras enviar los datos, Amazon Data Firehose espera primero una confirmación del punto de conexión HTTP. Si se produce un error o la confirmación no llega dentro del periodo de tiempo de espera de confirmación, Amazon Data Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento. Después de eso, Amazon Data Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Amazon Data Firehose envía datos al punto de conexión HTTP (ya sea en el intento inicial o en un reintento), reinicia el contador de tiempo de espera de confirmación y espera a que llegue una confirmación del punto de conexión HTTP.

Aunque se agote el tiempo de reintento, Amazon Data Firehose sigue esperando la confirmación hasta que la recibe o hasta que finaliza el periodo de tiempo de espera de confirmación. Si se agota el tiempo de espera de confirmación, Amazon Data Firehose determina si queda tiempo en el contador de reintento. Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una confirmación o determina que el tiempo de reintento se ha agotado.

Si no desea que Amazon Data Firehose vuelva a intentar el envío de datos, establezca este valor en 0.

Parámetros: opcional

Amazon Data Firehose incluye estos pares de clave-valor en cada llamada HTTP. Estos parámetros pueden ayudarlo a identificar y organizar sus destinos.

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía de un proveedor de servicios a otro.

> **Important**
>
> Para los destinos de punto final HTTP, si ves 413 códigos de respuesta del punto final de destino en CloudWatch Logs, reduce el tamaño de la sugerencia de almacenamiento en búfer en la transmisión de Firehose e inténtalo de nuevo.

Configuración de los ajustes de destino de Datadog

En esta sección se describen las opciones de uso de Datadog como destino. Para obtener más información sobre Datadog, consulta amazon_web_services/. https://docs.datadoghq.com/integrations/

- Proporcione valores para los siguientes campos.

  URL del punto de conexión HTTP

  Elija dónde desea enviar los datos desde una de las siguientes opciones del menú desplegable.

  - Registros de Datadog - US1
  - Registros de Datadog - US3
  - Registros de Datadog - US5
  - Registros de Datadog - AP1
  - Registros de Datadog: UE
  - Registros de Datadog: GOV
  - Métricas de Datadog: EE. UU.

- Métricas de Datadog - US5
- Métricas de Datadog - AP1
- Métricas de Datadog: UE
- Configuraciones de Datadog - US1
- Configuraciones de Datadog - US3
- Configuraciones de Datadog - US5
- Configuraciones de Datadog - AP1
- Configuraciones de Datadog: UE
- Configuraciones de Datadog: US GOV

Autenticación

Puede elegir entre introducir la clave de API directamente o recuperar el secreto para acceder AWS Secrets Manager a Datadog.

- Clave de API

  Póngase en contacto con Datadog para obtener la clave de API necesaria para permitir la entrega de datos en este punto de conexión desde Firehose.

- Secret

  Seleccione un secreto AWS Secrets Manager que contenga la clave de API de Datadog. Si no ve su secreto en la lista desplegable, cree uno en AWS Secrets Manager. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

Codificación de contenidos

Amazon Data Firehose utiliza la codificación de contenidos para comprimir el cuerpo de una solicitud antes de enviarla al destino. Selecciona GZIP o Disabled para codificar el enable/disable contenido de tu solicitud.

Retry duration

Especifique durante cuánto tiempo Amazon Data Firehose reintenta el envío de datos al punto de conexión HTTP seleccionado.

Tras enviar los datos, Amazon Data Firehose espera primero una confirmación del punto de conexión HTTP. Si se produce un error o la confirmación no llega dentro del periodo de tiempo de espera de confirmación, Amazon Data Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento.

Después de eso, Amazon Data Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Amazon Data Firehose envía datos al punto de conexión HTTP (ya sea en el intento inicial o en un reintento), reinicia el contador de tiempo de espera de confirmación y espera a que llegue una confirmación del punto de conexión HTTP.

Aunque se agote el tiempo de reintento, Amazon Data Firehose sigue esperando la confirmación hasta que la recibe o hasta que finaliza el periodo de tiempo de espera de confirmación. Si se agota el tiempo de espera de confirmación, Amazon Data Firehose determina si queda tiempo en el contador de reintento. Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una confirmación o determina que el tiempo de reintento se ha agotado.

Si no desea que Amazon Data Firehose vuelva a intentar el envío de datos, establezca este valor en 0.

Parámetros: opcional

Amazon Data Firehose incluye estos pares de clave-valor en cada llamada HTTP. Estos parámetros pueden ayudarlo a identificar y organizar sus destinos.

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía de un proveedor de servicios a otro.

Configuración de los ajustes de destino de Honeycomb

En esta sección se describen las opciones de uso de Honeycomb como destino. Para obtener más información sobre Honeycomb, consulte https://docs.honeycomb. -cloudwatch-metrics/. io/getting-data-in/metrics/aws

- Proporcione valores para los siguientes campos:

  Punto de conexión entre Honeycomb y Kinesis

  Especifique la URL del punto final HTTP en el siguiente formato: b.io/1/kinesis_events/{{dataset}} https://api.honeycom

Autenticación

Puede elegir entre introducir la clave de API directamente o recuperar el secreto para acceder a Honeycomb. AWS Secrets Manager

- Clave de API

  Póngase en contacto con Honeycomb para obtener la clave de API necesaria para permitir la entrega de datos en este punto de conexión desde Firehose.

- Secret

  Seleccione un secreto AWS Secrets Manager que contenga la clave de API de Honeycomb. Si no ve su secreto en la lista desplegable, cree uno en AWS Secrets Manager. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

Codificación de contenidos

Amazon Data Firehose utiliza la codificación de contenidos para comprimir el cuerpo de una solicitud antes de enviarla al destino. Elija GZIP para habilitar la codificación del contenido de su solicitud. Esta es la opción recomendada para el destino de Honeycomb.

Retry duration

Especifique durante cuánto tiempo Amazon Data Firehose reintenta el envío de datos al punto de conexión HTTP seleccionado.

Tras enviar los datos, Amazon Data Firehose espera primero una confirmación del punto de conexión HTTP. Si se produce un error o la confirmación no llega dentro del periodo de tiempo de espera de confirmación, Amazon Data Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento. Después de eso, Amazon Data Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Amazon Data Firehose envía datos al punto de conexión HTTP (ya sea en el intento inicial o en un reintento), reinicia el contador de tiempo de espera de confirmación y espera a que llegue una confirmación del punto de conexión HTTP.

Aunque se agote el tiempo de reintento, Amazon Data Firehose sigue esperando la confirmación hasta que la recibe o hasta que finaliza el periodo de tiempo de espera de confirmación. Si se agota el tiempo de espera de confirmación, Amazon Data Firehose

determina si queda tiempo en el contador de reintento. Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una confirmación o determina que el tiempo de reintento se ha agotado.

Si no desea que Amazon Data Firehose vuelva a intentar el envío de datos, establezca este valor en 0.

Parámetros: opcional

Amazon Data Firehose incluye estos pares de clave-valor en cada llamada HTTP. Estos parámetros pueden ayudarlo a identificar y organizar sus destinos.

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía de un proveedor de servicios a otro.

Configuración de los ajustes de destino de Coralogix

En esta sección se describen las opciones de uso de Coralogix como destino. Para obtener más información sobre Coralogix, consulte Introducción a Coralogix.

- Proporcione valores para los siguientes campos:

  URL del punto de conexión HTTP

  Elija la URL del punto de conexión HTTP entre las siguientes opciones del menú desplegable:

  - Coralogix: EE. UU.
  - Coralogix: SINGAPUR
  - Coralogix: IRLANDA
  - Coralogix: INDIA
  - Coralogix: ESTOCOLMO

  Autenticación

  Puede elegir entre introducir la clave privada directamente o recuperar el secreto para acceder AWS Secrets Manager a Coralogix.

- Clave privada

  Póngase en contacto con Coralogix para obtener la clave privada necesaria para permitir la entrega de datos en este punto de conexión desde Firehose.

- Secret

  Seleccione un secreto AWS Secrets Manager que contenga la clave privada de Coralogix. Si no ve su secreto en la lista desplegable, cree uno en AWS Secrets Manager. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

Codificación de contenidos

Amazon Data Firehose utiliza la codificación de contenidos para comprimir el cuerpo de una solicitud antes de enviarla al destino. Elija GZIP para habilitar la codificación del contenido de su solicitud. Esta es la opción recomendada para el destino de Coralogix.

Retry duration

Especifique durante cuánto tiempo Amazon Data Firehose reintenta el envío de datos al punto de conexión HTTP seleccionado.

Tras enviar los datos, Amazon Data Firehose espera primero una confirmación del punto de conexión HTTP. Si se produce un error o la confirmación no llega dentro del periodo de tiempo de espera de confirmación, Amazon Data Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento. Después de eso, Amazon Data Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Amazon Data Firehose envía datos al punto de conexión HTTP (ya sea en el intento inicial o en un reintento), reinicia el contador de tiempo de espera de confirmación y espera a que llegue una confirmación del punto de conexión HTTP.

Aunque se agote el tiempo de reintento, Amazon Data Firehose sigue esperando la confirmación hasta que la recibe o hasta que finaliza el periodo de tiempo de espera de confirmación. Si se agota el tiempo de espera de confirmación, Amazon Data Firehose determina si queda tiempo en el contador de reintento. Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una confirmación o determina que el tiempo de reintento se ha agotado.

Si no desea que Amazon Data Firehose vuelva a intentar el envío de datos, establezca este valor en 0.

Parámetros: opcional

Amazon Data Firehose incluye estos pares de clave-valor en cada llamada HTTP. Estos parámetros pueden ayudarlo a identificar y organizar sus destinos.

- applicationName: entorno en el que se ejecuta Data Firehose
- subsystemName: nombre de la integración de Data Firehose
- computerName: nombre del flujo de Firehose en uso

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía según el proveedor de servicios.

Configuración de los ajustes de destino de Dynatrace

En esta sección se describen las opciones de uso de Dynatrace como destino. Para obtener más información, consulte -metric-streams/. https://www.dynatrace.com/support/ help/technology-support/cloud-platforms/amazon-web-services/integrations/cloudwatch

- Elija opciones para usar Dynatrace como destino de su flujo de Firehose.

  Tipo de ingesta

  Elija si quiere entregar métricas o registros (predeterminado) en Dynatrace para su posterior análisis y procesamiento.

  URL del punto de conexión HTTP

  Elija la URL del punto de conexión HTTP (Dynatrace EE. UU., Dynatrace UE o Dynatrace Global) en el menú desplegable.

  Autenticación

  Puede elegir entre introducir el token de la API directamente o recuperar el secreto para acceder a Dynatrace. AWS Secrets Manager

  - Token de la API

Genere el token de la API de Dynatrace que necesita para permitir la entrega de datos en este punto de conexión desde Firehose. Para obtener más información, consulte API de Dynatrace: tokens y autenticación.

- Secret

  Seleccione un secreto AWS Secrets Manager que contenga el token de API de Dynatrace. Si no ve su secreto en la lista desplegable, cree uno en AWS Secrets Manager. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

URL de la API

Proporcione la URL de la API de su entorno de Dynatrace.

Codificación de contenidos

Elija si desea habilitar la codificación de contenido para comprimir el cuerpo de la solicitud. Amazon Data Firehose utiliza la codificación de contenidos para comprimir el cuerpo de una solicitud antes de enviarla al destino. Cuando está habilitada, el contenido se comprime en formato GZIP.

Retry duration

Especifique durante cuánto tiempo Firehose reintenta el envío de datos al punto de conexión HTTP seleccionado.

Tras enviar los datos, Firehose espera primero una confirmación del punto de conexión HTTP. Si se produce un error o la confirmación no llega dentro del periodo de tiempo de espera de confirmación, Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento. Después de eso, Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Firehose envía datos al punto de conexión HTTP, ya sea en el intento inicial o en un reintento, reinicia el contador de tiempo de espera de confirmación y espera a que llegue una confirmación del punto de conexión HTTP.

Aunque se agote el tiempo de reintento, Firehose sigue esperando la confirmación hasta que la recibe o hasta que finaliza el tiempo de espera de confirmación. Si se agota el tiempo de espera de confirmación, Firehose determina si queda tiempo en el contador de reintento.

Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una confirmación o determina que el tiempo de reintento se ha agotado.

Si no desea que Firehose vuelva a intentar el envío de datos, establezca este valor en 0.

Parámetros: opcional

Amazon Data Firehose incluye estos pares de clave-valor en cada llamada HTTP. Estos parámetros pueden ayudarlo a identificar y organizar sus destinos.

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. Las sugerencias del búfer incluyen el tamaño y el intervalo del búfer para los flujos. El tamaño del búfer recomendado para el destino varía según el proveedor de servicios.

Configure los ajustes de destino para LogicMonitor

Esta sección describe las opciones de uso de LogicMonitor como destino. Para obtener más información, consulte https://www.logicmonitor.com.

- Proporcione valores para los siguientes campos:

  URL del punto de conexión HTTP

  Especifique la URL del punto de conexión HTTP en el siguiente formato.

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

  Autenticación

  Puedes elegir entre introducir la clave de API directamente o recuperar el secreto desde donde AWS Secrets Manager accedes LogicMonitor.

  - Clave de API

    Póngase en contacto LogicMonitor para obtener la clave de API que necesita para habilitar la entrega de datos a este punto final desde Firehose.

  - Secret

Seleccione un secreto AWS Secrets Manager que contenga la clave de API. LogicMonitor Si no ve su secreto en la lista desplegable, cree uno en AWS Secrets Manager. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

Codificación de contenidos

Amazon Data Firehose utiliza la codificación de contenidos para comprimir el cuerpo de una solicitud antes de enviarla al destino. Selecciona GZIP o Disabled para codificar enable/disable el contenido de tu solicitud.

Retry duration

Especifique durante cuánto tiempo Amazon Data Firehose reintenta el envío de datos al punto de conexión HTTP seleccionado.

Tras enviar los datos, Amazon Data Firehose espera primero una confirmación del punto de conexión HTTP. Si se produce un error o la confirmación no llega dentro del periodo de tiempo de espera de confirmación, Amazon Data Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento. Después de eso, Amazon Data Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Amazon Data Firehose envía datos al punto de conexión HTTP (ya sea en el intento inicial o en un reintento), reinicia el contador de tiempo de espera de confirmación y espera a que llegue una confirmación del punto de conexión HTTP.

Aunque se agote el tiempo de reintento, Amazon Data Firehose sigue esperando la confirmación hasta que la recibe o hasta que finaliza el periodo de tiempo de espera de confirmación. Si se agota el tiempo de espera de confirmación, Amazon Data Firehose determina si queda tiempo en el contador de reintento. Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una confirmación o determina que el tiempo de reintento se ha agotado.

Si no desea que Amazon Data Firehose vuelva a intentar el envío de datos, establezca este valor en 0.

Parámetros: opcional

Amazon Data Firehose incluye estos pares de clave-valor en cada llamada HTTP. Estos parámetros pueden ayudarlo a identificar y organizar sus destinos.

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía de un proveedor de servicios a otro.

Configuración de los ajustes de destino de Logz.io

En esta sección se describen las opciones de uso de Logz.io como destino. Para obtener más información, consulte https://logz.io/.

> **Note**
>
> En la región Europa (Milán), Logz.io no se admite como destino de Amazon Data Firehose.

- Proporcione valores para los siguientes campos:

  URL del punto de conexión HTTP

  Especifique la URL del punto de conexión HTTP en el siguiente formato. La URL debe ser una URL HTTPS.

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

  Por ejemplo

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

  Autenticación

  Puedes elegir entre introducir el token de envío directamente o recuperar el secreto AWS Secrets Manager para acceder a Logz.io.

  - Token de envío

    Póngase en contacto con Logz.io para obtener el token de envío necesario para permitir la entrega de datos en este punto de conexión desde Firehose.

- Secret

  Selecciona un secreto AWS Secrets Manager que contenga el token de envío de Logz.io. Si no ve su secreto en la lista desplegable, cree uno en AWS Secrets Manager. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

Retry duration

Especifique durante cuánto tiempo Amazon Data Firehose reintenta el envío de datos a Logz.io.

Tras enviar los datos, Amazon Data Firehose espera primero una confirmación del punto de conexión HTTP. Si se produce un error o la confirmación no llega dentro del periodo de tiempo de espera de confirmación, Amazon Data Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento. Después de eso, Amazon Data Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Amazon Data Firehose envía datos al punto de conexión HTTP (ya sea en el intento inicial o en un reintento), reinicia el contador de tiempo de espera de confirmación y espera a que llegue una confirmación del punto de conexión HTTP.

Aunque se agote el tiempo de reintento, Amazon Data Firehose sigue esperando la confirmación hasta que la recibe o hasta que finaliza el periodo de tiempo de espera de confirmación. Si se agota el tiempo de espera de confirmación, Amazon Data Firehose determina si queda tiempo en el contador de reintento. Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una confirmación o determina que el tiempo de reintento se ha agotado.

Si no desea que Amazon Data Firehose vuelva a intentar el envío de datos, establezca este valor en 0.

Parámetros: opcional

Amazon Data Firehose incluye estos pares de clave-valor en cada llamada HTTP. Estos parámetros pueden ayudarlo a identificar y organizar sus destinos.

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía de un proveedor de servicios a otro.

Configuración de los ajustes de destino de MongoDB Atlas

En esta sección se describen las opciones de uso de MongoDB Atlas como destino. Para obtener más información, consulte MongoDB Atlas en Amazon Web Services.

- Proporcione valores para los siguientes campos:

  URL de API Gateway

  Especifique la URL del punto de conexión HTTP en el siguiente formato.

  ```
  https://xxxxx.execute-api.region.amazonaws.com/stage
  ```

  La URL debe ser una URL `HTTPS`.

  Autenticación

  Puede elegir entre introducir la clave de API directamente o recuperar el secreto AWS Secrets Manager para acceder a MongoDB Atlas.

  - Clave de API

    Siga las instrucciones de MongoDB Atlas en Amazon Web Services para obtener el `APIKeyValue` necesario para la entrega de datos en este punto de conexión desde Firehose.

  - Secret

    Seleccione un secreto AWS Secrets Manager que contenga la clave de API para API Gateway respaldada por Lambda que interactúa con MongoDB Atlas. Si no ve su secreto en la lista desplegable, cree uno en AWS Secrets Manager. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

Codificación de contenidos

Amazon Data Firehose utiliza la codificación de contenidos para comprimir el cuerpo de una solicitud antes de enviarla al destino. Selecciona GZIP o Disabled para codificar el enable/disable contenido de tu solicitud.

Retry duration

Especifique durante cuánto tiempo Amazon Data Firehose reintenta el envío de datos al proveedor de terceros seleccionado.

Tras enviar los datos, Amazon Data Firehose espera primero una confirmación del punto de conexión HTTP. Si se produce un error o la confirmación no llega dentro del periodo de tiempo de espera de confirmación, Amazon Data Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento. Después de eso, Amazon Data Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Amazon Data Firehose envía datos al punto de conexión HTTP (ya sea en el intento inicial o en un reintento), reinicia el contador de tiempo de espera de confirmación y espera a que llegue una confirmación del punto de conexión HTTP.

Aunque se agote el tiempo de reintento, Amazon Data Firehose sigue esperando la confirmación hasta que la recibe o hasta que finaliza el periodo de tiempo de espera de confirmación. Si se agota el tiempo de espera de confirmación, Amazon Data Firehose determina si queda tiempo en el contador de reintento. Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una confirmación o determina que el tiempo de reintento se ha agotado.

Si no desea que Amazon Data Firehose vuelva a intentar el envío de datos, establezca este valor en 0.

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía de un proveedor de servicios a otro.

Parámetros: opcional

Amazon Data Firehose incluye estos pares de clave-valor en cada llamada HTTP. Estos parámetros pueden ayudarlo a identificar y organizar sus destinos.

Configuración de los ajustes de destino de New Relic

En esta sección se describen las opciones de uso de New Relic como destino. Para obtener más información, consulte https://newrelic.com.

- Proporcione valores para los siguientes campos:

  URL del punto de conexión HTTP

  Elija la URL del punto de conexión HTTP entre las siguientes opciones del menú desplegable.

  - Registros de New Relic: EE. UU.
  - Métricas de New Relic: EE. UU.
  - Métricas de New Relic: UE

  Autenticación

  Puedes elegir entre introducir la clave de API directamente o recuperar el secreto AWS Secrets Manager para acceder a New Relic.

  - Clave de API

    Ingrese la clave de licencia, que es una cadena hexadecimal de 40 caracteres, en la configuración de la cuenta de New Relic One. Esta clave de API es necesaria para permitir la entrega de datos en este punto de conexión desde Firehose.

  - Secret

    Selecciona un secreto AWS Secrets Manager que contenga la clave de API de New Relic. Si no ve su secreto en la lista desplegable, cree uno en AWS Secrets Manager. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

Codificación de contenidos

Amazon Data Firehose utiliza la codificación de contenidos para comprimir el cuerpo de una solicitud antes de enviarla al destino. Selecciona GZIP o Desactivado para codificar el enable/disable contenido de tu solicitud.

Retry duration

Especifique durante cuánto tiempo Amazon Data Firehose reintenta el envío de datos al punto de conexión HTTP de New Relic.

Tras enviar los datos, Amazon Data Firehose espera primero una confirmación del punto de conexión HTTP. Si se produce un error o la confirmación no llega dentro del periodo de tiempo de espera de confirmación, Amazon Data Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento. Después de eso, Amazon Data Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Amazon Data Firehose envía datos al punto de conexión HTTP (ya sea en el intento inicial o en un reintento), reinicia el contador de tiempo de espera de confirmación y espera a que llegue una confirmación del punto de conexión HTTP.

Aunque se agote el tiempo de reintento, Amazon Data Firehose sigue esperando la confirmación hasta que la recibe o hasta que finaliza el periodo de tiempo de espera de confirmación. Si se agota el tiempo de espera de confirmación, Amazon Data Firehose determina si queda tiempo en el contador de reintento. Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una confirmación o determina que el tiempo de reintento se ha agotado.

Si no desea que Amazon Data Firehose vuelva a intentar el envío de datos, establezca este valor en 0.

Parámetros: opcional

Amazon Data Firehose incluye estos pares de clave-valor en cada llamada HTTP. Estos parámetros pueden ayudarlo a identificar y organizar sus destinos.

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía de un proveedor de servicios a otro.

Configuración de los ajustes de destino de Snowflake

En esta sección, se describen las opciones de uso de Snowflake como destino.

> **Note**
>
> La integración de Firehose con Snowflake está disponible en EE. UU. Este (Virginia del Norte), EE. UU. Oeste (Oregón), Europa (Irlanda), EE. UU. Este (Ohio), Asia Pacífico (Tokio), Europa (Fráncfort), Asia Pacífico (Singapur), Asia Pacífico (Seúl) y Asia Pacífico (Sídney), Asia Pacífico (Bombay), Europa (Londres), Sudamérica (São Paulo), Canadá (), Europa (París), Asia Pacífico (Osaka), Europa (Estocolmo), Asia Pacífico (Yakarta). Regiones de AWS

Configuraciones de conexión

- Proporcione valores para los siguientes campos:

  URL de la cuenta de Snowflake

  Especifique la URL de una cuenta de Snowflake. Por ejemplo: `xy12345.us-east-1.aws.snowflakecomputing.com`. Consulte la documentación de Snowflake para saber cómo determinar la URL de su cuenta. Tenga en cuenta que no debe especificar el número de puerto, y el protocolo (https://) es opcional.

  Autenticación

  Puede elegir entre introducir el nombre de usuario, la clave privada y la contraseña manualmente o recuperar el secreto AWS Secrets Manager para acceder a Snowflake.

  - Inicio de sesión de usuario

    Especifique el usuario de Snowflake que se utilizará para cargar los datos. Asegúrese de que el usuario tenga acceso para insertar datos en la tabla de Snowflake.

- Clave privada

  Especifique la clave privada para la autenticación con Snowflake en formato `PKCS8`. Además, no incluya el encabezado y el pie de página PEM como parte de la clave privada. Si la clave está dividida en varias líneas, elimine los saltos de línea. A continuación, se muestra un ejemplo de cómo debe ser su clave privada.

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

  Elimine el espacio en `KEY_CONTENT` y colóquela en Firehose. Se requiere ningún carácter header/footer o caracteres de nueva línea.

- Contraseña

  Especifique la contraseña para descifrar la clave privada cifrada. Puede dejar este campo vacío si la clave privada no está cifrada. Para obtener más información, consulte Uso de la autenticación de pares de claves y la rotación de claves.

- Secret

  Seleccione un secreto AWS Secrets Manager que contenga las credenciales de Snowflake. Si no ve su secreto en la lista desplegable, cree uno en AWS Secrets Manager. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

Configuración de roles

Usar el rol de Snowflake predeterminado: si se selecciona esta opción, Firehose no pasará ningún rol a Snowflake. Se asume que el rol predeterminado carga los datos. Asegúrese de que el rol predeterminado tenga permiso para insertar datos en la tabla de Snowflake.

Usar un rol de Snowflake personalizado: introduzca un rol de Snowflake no predeterminado para que lo asuma Firehose al cargar datos en la tabla de Snowflake.

Conexión a Snowflake

Las opciones son Privada o Pública.

ID de VPCE privado (opcional)

El ID de VPCE para que Firehose se conecte de forma privada con Snowflake. El formato de ID es com.amazonaws.vpce. [región] .vpce-svc-. *[id]* Para obtener más información, consulte & Snowflake.AWS PrivateLink

> **Note**
>
> Si su clúster de Snowflake tiene habilitados los enlaces privados, utilice una política de red basada en `AwsVpceIds` para permitir los datos de Amazon Data Firehose. Firehose no requiere que configure una política de red basada en IP en su cuenta de Snowflake. Si tiene habilitada una política de red basada en IP, podría interferir con la conectividad de Firehose. En un caso extremo que requiera una política basada en IP, póngase en contacto con el equipo de Firehose enviando un ticket de soporte. Para obtener una lista de los VPCE IDs que puede utilizar, consulte la. Acceso a Snowflake en VPC

Configuración de las bases de datos

- Debe especificar la siguiente configuración para poder utilizar Snowflake como destino del flujo de Firehose.
  - Base de datos de Snowflake: todos los datos de Snowflake se mantienen en bases de datos.
  - Esquema de Snowflake: cada base de datos consta de uno o más esquemas, que son agrupaciones lógicas de objetos de base de datos, como tablas y vistas.
  - Tabla de Snowflake: todos los datos de Snowflake se almacenan en tablas de bases de datos, estructuradas de forma lógica como conjuntos de columnas y filas.

Opciones de carga de datos para la tabla de Snowflake

- Utilice claves JSON como nombres de columnas
- Utilice columnas VARIANT
  - Nombre de la columna de contenido: especifique un nombre de columna en la tabla, donde se deben cargar los datos sin procesar.

- Nombre de la columna de metadatos (opcional): especifique un nombre de columna en la tabla, donde se debe cargar la información de los metadatos. Al habilitar este campo, verá la siguiente columna en la tabla Snowflake según el tipo de origen.

  Para Direct PUT como origen

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

  Para Kinesis Data Stream como origen

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

Retry duration

Tiempo (entre 0 y 7200 segundos) para que Firehose vuelva a intentarlo si falla la apertura del canal o la entrega a Snowflake debido a problemas con el servicio de Snowflake. Firehose hace reintentos con un retroceso exponencial hasta que finaliza el tiempo de reintento. Si establece la duración del reintento en 0 (cero) segundos, Firehose no lo reintenta tras producirse errores en Snowflake y enruta los datos al bucket de errores de Amazon S3.

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía de un proveedor de servicios a otro. Para obtener más información, consulte Configuración de sugerencias de almacenamiento en búfer.

Configuración de los ajustes de destino de Splunk

Esta sección describe las opciones de uso de Splunk como destino.

> **Note**
>
> Firehose entrega los datos a los clústeres de Splunk configurados con equilibrador de carga clásico o un equilibrador de carga de aplicación.

- Proporcione valores para los siguientes campos:

  Splunk cluster endpoint

  Para determinar el punto de conexión, consulte Configuración de Amazon Data Firehose para enviar datos a la plataforma Spunk en la documentación de Splunk.

  Splunk endpoint type

  Elija `Raw endpoint` en la mayoría de los casos. Elija `Event endpoint` si ha preprocesado sus datos AWS Lambda para enviarlos a diferentes índices por tipo de evento. Para obtener información sobre el punto de conexión que debe utilizar, consulte Configuración de Amazon Data Firehose para enviar datos a la plataforma de Spunk en la documentación de Splunk.

  Autenticación

  Puedes elegir entre introducir el token de autenticación directamente o recuperar el secreto para acceder AWS Secrets Manager a Splunk.

  - Authentication token

    Para configurar un punto de conexión de Splunk que pueda recibir datos de Amazon Data Firehose, consulte Descripción general de la instalación y la configuración del complemento de Splunk en Amazon Kinesis Firehose en la documentación de Splunk. Guarde el token que obtiene de Splunk al configurar el punto de conexión de este flujo de Firehose y añádalo aquí.

  - Secret

    Seleccione un secreto AWS Secrets Manager que contenga el token de autenticación de Splunk. Si no ve su secreto en la lista desplegable, cree uno en AWS Secrets Manager. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

HEC acknowledgement timeout

Especifique durante cuánto tiempo debe esperar Amazon Data Firehose la confirmación de índices de Splunk. Si Splunk no envía la confirmación antes de que finalice el tiempo de espera, Amazon Data Firehose considera que ha habido un error en la entrega de datos. A continuación, Amazon Data Firehose hace un reintento o crea una copia de seguridad de los datos en el bucket de Amazon S3, según el valor que se haya establecido para el tiempo de reintento.

Retry duration

Especifique durante cuánto tiempo Amazon Data Firehose reintenta el envío de datos a Splunk.

Tras enviar los datos, Amazon Data Firehose espera primero una confirmación de Splunk. Si se produce un error o la confirmación no llega dentro del periodo de tiempo de espera de confirmación, Amazon Data Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento. Después de eso, Amazon Data Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Amazon Data Firehose envía datos a Splunk (ya sea en el intento inicial o en un reintento), reinicia el contador de tiempo de espera de confirmación y espera a que llegue una confirmación de Splunk.

Aunque se agote el tiempo de reintento, Amazon Data Firehose sigue esperando la confirmación hasta que la recibe o hasta que finaliza el periodo de tiempo de espera de confirmación. Si se agota el tiempo de espera de confirmación, Amazon Data Firehose determina si queda tiempo en el contador de reintento. Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una confirmación o determina que el tiempo de reintento se ha agotado.

Si no desea que Amazon Data Firehose vuelva a intentar el envío de datos, establezca este valor en 0.

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía según el proveedor de servicios.

Configuración de los ajustes de destino de Splunk Observability Cloud

En esta sección se describen las opciones de uso de Splunk Observability Cloud como destino. Para obtener más información, consulte https://docs.splunk.com/observability/en/gdi/get-data-in/connect/aws/aws-apiconfig.html# - -api connect-to-aws-using. the-splunk-observability-cloud

- Proporcione valores para los siguientes campos:

  URL del punto de conexión de ingesta de la nube

  Puede encontrar la URL de ingesta de datos en tiempo real de Splunk Observability Cloud en Profile > Organizations > Real-time Data Ingest Endpoint, en la consola de Splunk Observability.

  Autenticación

  Puede elegir entre introducir el token de acceso directamente o recuperar el secreto para acceder a Splunk Observability Cloud. AWS Secrets Manager

  - Token de acceso

    Copie su token de acceso a Splunk Observability con el ámbito de autorización de INGEST desde Tokens de acceso en Ajustes en la consola de Splunk Observability.

  - Secret

    Seleccione un secreto AWS Secrets Manager que contenga el token de acceso a Splunk Observability Cloud. Si no ve su secreto en la lista desplegable, cree uno en AWS Secrets Manager. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

  Codificación de contenidos

  Amazon Data Firehose utiliza la codificación de contenidos para comprimir el cuerpo de una solicitud antes de enviarla al destino. Seleccione GZIP o Desactivado para codificar el enable/disable contenido de su solicitud.

  Retry duration

  Especifique durante cuánto tiempo Amazon Data Firehose reintenta el envío de datos al punto de conexión HTTP seleccionado.

Tras enviar los datos, Amazon Data Firehose espera primero una confirmación del punto de conexión HTTP. Si se produce un error o la confirmación no llega dentro del periodo de tiempo de espera de confirmación, Amazon Data Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento. Después de eso, Amazon Data Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Amazon Data Firehose envía datos al punto de conexión HTTP (ya sea en el intento inicial o en un reintento), reinicia el contador de tiempo de espera de confirmación y espera a que llegue una confirmación del punto de conexión HTTP.

Aunque se agote el tiempo de reintento, Amazon Data Firehose sigue esperando la confirmación hasta que la recibe o hasta que finaliza el periodo de tiempo de espera de confirmación. Si se agota el tiempo de espera de confirmación, Amazon Data Firehose determina si queda tiempo en el contador de reintento. Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una confirmación o determina que el tiempo de reintento se ha agotado.

Si no desea que Amazon Data Firehose vuelva a intentar el envío de datos, establezca este valor en 0.

Parámetros: opcional

Amazon Data Firehose incluye estos pares de clave-valor en cada llamada HTTP. Estos parámetros pueden ayudarlo a identificar y organizar sus destinos.

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino varía de un proveedor de servicios a otro.

Configuración de los ajustes de destino de Sumo Logic

En esta sección se describen las opciones de uso de Sumo Logic como destino. Para obtener más información, consulte https://www.sumologic.com.

- Proporcione valores para los siguientes campos:

URL del punto de conexión HTTP

Especifique la URL del punto de conexión HTTP en el siguiente formato: `https://deployment name.sumologic.net/receiver/v1/kinesis/dataType/access token`. La URL debe ser una URL HTTPS.

Codificación de contenidos

Amazon Data Firehose utiliza la codificación de contenidos para comprimir el cuerpo de una solicitud antes de enviarla al destino. Selecciona GZIP o Desactivado para codificar el enable/disable contenido de tu solicitud.

Retry duration

Especifique durante cuánto tiempo Amazon Data Firehose reintenta el envío de datos a Sumo Logic.

Tras enviar los datos, Amazon Data Firehose espera primero una confirmación del punto de conexión HTTP. Si se produce un error o la confirmación no llega dentro del periodo de tiempo de espera de confirmación, Amazon Data Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento. Después de eso, Amazon Data Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Amazon Data Firehose envía datos al punto de conexión HTTP (ya sea en el intento inicial o en un reintento), reinicia el contador de tiempo de espera de confirmación y espera a que llegue una confirmación del punto de conexión HTTP.

Aunque se agote el tiempo de reintento, Amazon Data Firehose sigue esperando la confirmación hasta que la recibe o hasta que finaliza el periodo de tiempo de espera de confirmación. Si se agota el tiempo de espera de confirmación, Amazon Data Firehose determina si queda tiempo en el contador de reintento. Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una confirmación o determina que el tiempo de reintento se ha agotado.

Si no desea que Amazon Data Firehose vuelva a intentar el envío de datos, establezca este valor en 0.

Parámetros: opcional

Amazon Data Firehose incluye estos pares de clave-valor en cada llamada HTTP. Estos parámetros pueden ayudarlo a identificar y organizar sus destinos.

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño del búfer recomendado para el destino de Elastic varía de un proveedor de servicios a otro.

Configuración de los ajustes de destino de Elastic

En esta sección se describen las opciones de uso de Elastic como destino.

- Proporcione valores para los siguientes campos:

  URL del punto de conexión de Elastic

  Especifique la URL del punto de conexión HTTP en el siguiente formato: `https://<cluster-id>.es.<region>.aws.elastic-cloud.com`. La URL debe ser una URL HTTPS.

  Autenticación

  Puedes elegir entre introducir la clave de API directamente o recuperar el secreto AWS Secrets Manager para acceder a Elastic.

  - Clave de API

    Póngase en contacto con Elastic para obtener la clave de API necesaria para permitir la entrega de datos en su servicio desde Firehose.

  - Secret

    Selecciona un secreto AWS Secrets Manager que contenga la clave de API de Elastic. Si no ve su secreto en la lista desplegable, cree uno en AWS Secrets Manager. Para obtener más información, consulte Autenticate con AWS Secrets Manager Amazon Data Firehose.

Codificación de contenidos

Amazon Data Firehose utiliza la codificación de contenidos para comprimir el cuerpo de una solicitud antes de enviarla al destino. Selecciona GZIP (que es lo que está seleccionado de forma predeterminada) o Inhabilitado para codificar enable/disable el contenido de tu solicitud.

Retry duration

Especifique durante cuánto tiempo Amazon Data Firehose reintenta el envío de datos a Elastic.

Tras enviar los datos, Amazon Data Firehose espera primero una confirmación del punto de conexión HTTP. Si se produce un error o la confirmación no llega dentro del periodo de tiempo de espera de confirmación, Amazon Data Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento. Después de eso, Amazon Data Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Amazon Data Firehose envía datos al punto de conexión HTTP (ya sea en el intento inicial o en un reintento), reinicia el contador de tiempo de espera de confirmación y espera a que llegue una confirmación del punto de conexión HTTP.

Aunque se agote el tiempo de reintento, Amazon Data Firehose sigue esperando la confirmación hasta que la recibe o hasta que finaliza el periodo de tiempo de espera de confirmación. Si se agota el tiempo de espera de confirmación, Amazon Data Firehose determina si queda tiempo en el contador de reintento. Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una confirmación o determina que el tiempo de reintento se ha agotado.

Si no desea que Amazon Data Firehose vuelva a intentar el envío de datos, establezca este valor en 0.

Parámetros: opcional

Amazon Data Firehose incluye estos pares de clave-valor en cada llamada HTTP. Estos parámetros pueden ayudarlo a identificar y organizar sus destinos.

Sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos en el destino especificado. El tamaño de búfer recomendado para el destino de Elastic es de 1 MiB.

Configuración de copias de seguridad

Amazon Data Firehose utiliza Amazon S3 para hacer copias de seguridad de todos los datos, o solo aquellos que han fallado, que intenta entregar en el destino elegido.

> **Important**
>
> - La configuración de copias de seguridad solo se admite si el origen del flujo de Firehose es Direct PUT o Kinesis Data Streams.
> - La característica de almacenamiento en búfer cero solo está disponible para los destinos de la aplicación y no está disponible para el destino de copias de seguridad de Amazon S3.

Puede especificar la configuración de copias de seguridad de S3 para el flujo de Firehose si ha elegido una de las siguientes opciones.

- Si establece Amazon S3 como destino de la transmisión de Firehose y elige especificar una función de AWS Lambda para transformar los registros de datos o si decide convertir los formatos de registro de datos para la transmisión de Firehose.
- Si establece Amazon Redshift como destino de la transmisión de Firehose y decide especificar una función AWS Lambda para transformar los registros de datos.
- Si configuras alguno de los siguientes servicios como destino de tu transmisión de Firehose: Amazon OpenSearch Service, Datadog, Dynatrace, HTTP Endpoint, LogicMonitor MongoDB Cloud, New Relic, Splunk o Sumo Logic, Snowflake o Apache Iceberg Tables.

A continuación, se indica la configuración de copias de seguridad del flujo de Firehose:

- Copia de seguridad de registros de origen en Amazon S3: si el destino seleccionado es S3 o Amazon Redshift, esta configuración indica si desea habilitar la copia de seguridad de los datos de

origen o mantenerla deshabilitada. Si hay algún otro servicio admitido (que no sea S3 o Amazon Redshift) como destino seleccionado, esta configuración indica si desea hacer una copia de seguridad de todos los datos de origen o solo de los datos fallidos.

- Bucket de copias de seguridad de S3: es el bucket de S3 en el que Amazon Data Firehose hace una copia de seguridad de sus datos.
- Prefijo del bucket de copias de seguridad de S3: es el prefijo en el que Amazon Data Firehose hace una copia de seguridad de sus datos.
- Prefijo de salida de errores del bucket de copias de seguridad de S3: se hace una copia de seguridad de todos los datos fallidos en este prefijo de salida de errores del bucket de S3.
- Sugerencias de almacenamiento en búfer, compresión y cifrado para copia de seguridad: Amazon Data Firehose utiliza Amazon S3 para hacer copias de seguridad de todos los datos, o solo aquellos que han fallado, que intenta entregar en el destino elegido. Amazon Data Firehose almacena en búfer los datos de entrada antes de entregarlos (hacer una copia de seguridad de ellos) en Amazon S3. Puedes elegir un tamaño de búfer de 1 a 128 segundos y un intervalo de búfer de 60 a 900 segundos. MiBs La condición que primero se cumpla desencadenará la entrega de datos en Amazon S3. Si habilita la transformación de datos, el intervalo del búfer se aplica a partir del momento en que Amazon Data Firehose recibe los datos transformados hasta la entrega de los datos en Amazon S3. Si la entrega de los datos en el destino se realiza a una velocidad inferior a la de la escritura de datos en el flujo de Firehose, Amazon Data Firehose aumenta el tamaño del búfer de forma dinámica para alcanzar esa velocidad. Esta acción ayuda a garantizar que todos los datos se entregan en el destino.
- Compresión en S3: elija la compresión de datos GZIP, Snappy, Zip o Snappy compatible con Hadoop, o sin compresión de datos. Las compresiones Snappy, Zip y Snappy compatible con Hadoop no están disponibles para los flujos de Firehose con Amazon Redshift como destino.
- Formato de extensión de archivo S3 (opcional): especifique un formato de extensión de archivo para los objetos entregados al bucket de destino de Amazon S3. Si habilita esta característica, la extensión de archivo especificada anulará las extensiones de archivo predeterminadas incorporadas por las funciones de conversión de formato de datos o de compresión en S3, como .parquet o .gz. Asegúrese de haber configurado la extensión de archivo correcta cuando utilice esta característica con la conversión de formato de datos o la compresión en S3. La extensión del archivo debe empezar con un punto (.) y puede contener los caracteres permitidos: 0-9a-z!-_.*‘(). La extensión del archivo no puede superar los 128 caracteres.
- Firehose admite el cifrado del lado del servidor de Amazon S3 con AWS Key Management Service (SSE-KMS) para cifrar los datos entregados en Amazon S3. Puede optar por utilizar el tipo de cifrado predeterminado especificado en el depósito S3 de destino o cifrar con una clave de la lista

de claves de su propiedad. AWS KMS Si cifra los datos con AWS KMS claves, puede usar la clave AWS administrada predeterminada (aws/s3) o una clave administrada por el cliente. Para obtener más información, consulte Protección de datos mediante el cifrado del lado del servidor con claves administradas por AWS KMS (SSE-KMS).

Configuración de sugerencias de almacenamiento en búfer

Amazon Data Firehose almacena en búfer una cantidad determinada de datos de streaming de entrada (tamaño del almacenamiento en búfer) durante un periodo determinado (intervalo del almacenamiento en búfer) antes de entregarlos en los destinos especificados. Debería utilizar sugerencias de almacenamiento en búfer cuando desee entregar archivos de tamaño óptimo a Amazon S3 y obtener un mejor rendimiento de las aplicaciones de procesamiento de datos o para ajustar la tasa de entrega de Firehose para que coincida con la velocidad de destino.

Puede configurar el tamaño y el intervalo del búfer al crear nuevos flujos de Firehose o actualizar el tamaño y el intervalo del búfer en sus flujos de Firehose existentes. El tamaño del búfer se mide en segundos y el intervalo de almacenamiento en MBs búfer. Sin embargo, si especifica un valor para uno de ellos, también deberá proporcionar un valor para el otro. La primera condición del búfer que se cumpla ordenará a Firehose que entregue los datos. Si no configura los valores de almacenamiento en búfer, se utilizarán los valores predeterminados.

Puede configurar las sugerencias de almacenamiento en búfer de Firehose a través de,, o. Consola de administración de AWS AWS Command Line Interface AWS SDKs Para las transmisiones existentes, puedes reconfigurar las sugerencias de almacenamiento en búfer con un valor que se adapte a tus casos de uso mediante la opción Editar de la consola o mediante la API. UpdateDestination En el caso de las transmisiones nuevas, puedes configurar las sugerencias de almacenamiento en búfer como parte de la creación de nuevas transmisiones mediante la consola o la API. CreateDeliveryStream Para ajustar el tamaño del búfer, establece `SizeInMBs` y `IntervalInSeconds` en el `DestinationConfiguration` parámetro específico de destino de la API o. CreateDeliveryStreamUpdateDestination

> **Note**
>
> - Las sugerencias de búfer se aplican a nivel de fragmento o partición, mientras que las sugerencias de búfer de partición dinámica se aplican a nivel de flujo o tema.
> - Para reducir las latencias en los casos de uso en tiempo real, puede utilizar una sugerencia de intervalo de almacenamiento en búfer cero. Al configurar el intervalo

de almacenamiento en búfer como cero segundos, Firehose no almacenará los datos en búfer y los entregará en unos segundos. Antes de cambiar las sugerencias de almacenamiento en búfer por un valor inferior, consulte con el proveedor las sugerencias de almacenamiento en búfer recomendadas de Firehose para sus destinos.

- La característica de almacenamiento en búfer cero solo está disponible para los destinos de la aplicación y no está disponible para el destino de copias de seguridad de Amazon S3.
- La característica de almacenamiento en búfer cero no está disponible para el particionamiento dinámico.
- Firehose utiliza la carga en varias partes para el destino de S3 cuando configura un intervalo de tiempo de búfer inferior a 60 segundos para ofrecer latencias más bajas. Debido a que la carga se realiza en varias partes para el destino de S3, los costos de la API PUT de S3 aumentarán en cierta medida si elige un intervalo de tiempo de almacenamiento inferior a 60 segundos.

Para ver los rangos de sugerencias de almacenamiento en búfer específicos del destino y los valores predeterminados, consulte la siguiente tabla:

Destino	Tamaño del búfer en MB (valor predeterminado entre paréntesis)	Intervalo del búfer en segundos (valor predeterminado entre paréntesis)
Amazon S3	1-128 (5)	0-900 (300)
Tablas de Apache Iceberg	1-128 (5)	0-900 (300)
Amazon Redshift	1-128 (5)	0-900 (300)
OpenSearch Sin servidor	1-100 (5)	0-900 (300)
OpenSearch	1-100 (5)	0-900 (300)

Destino	Tamaño del búfer en MB (valor predeterminado entre paréntesis)	Intervalo del búfer en segundos (valor predeterminado entre paréntesis)
Splunk	1-5 (5)	0-60 (60)
Datadog	1-4 (4)	0-900 (60)
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)
Punto de conexión 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)

Configuración de opciones avanzadas

En la sección siguiente, se indica la configuración avanzada del flujo de Firehose.

- Cifrado del lado del servidor: Amazon Data Firehose admite el cifrado del lado del servidor de Amazon S3 con el Servicio de administración de AWS claves (AWS KMS) para cifrar los datos entregados en Amazon S3. Para obtener más información, consulte Protección de datos mediante el cifrado del lado del servidor con claves administradas por KMS (SSE-KMS). AWS
- Registro de errores: Amazon Data Firehose registra los errores relacionados con el procesamiento y la entrega. Además, cuando la transformación de datos está habilitada, puede registrar las invocaciones de Lambda y enviar los errores de entrega de datos a Logs. CloudWatch Para obtener más información, consulte Supervisión de Amazon Data Firehose mediante Registros de CloudWatch.

  > **Important**
  >
  > Si bien es opcional, se recomienda encarecidamente habilitar el registro de errores de Amazon Data Firehose durante la creación del flujo de Firehose. Esta práctica garantiza que pueda acceder a los detalles de los errores en caso de que se produzcan fallas en el procesamiento o la entrega de los registros.

- Permisos: Amazon Data Firehose utiliza roles de IAM para todos los permisos que necesita el flujo de Firehose. Puede crear un nuevo rol en el que los permisos necesarios se asignen automáticamente o elegir un rol existente creado para Amazon Data Firehose. La función se utiliza para conceder a Firehose acceso a varios servicios, como el bucket de S3, la clave de AWS KMS (si el cifrado de datos está activado) y la función Lambda (si la transformación de datos está habilitada). La consola podría crear un rol con marcadores de posición. Para obtener más información, consulte ¿Qué es IAM?.

  > **Note**
  >
  > El rol de IAM (incluidos los marcadores de posición) se crea en función de la configuración que se elija al crear un flujo de Firehose. Si realiza algún cambio en el origen o destino del flujo de Firehose, debe actualizar manualmente el rol de IAM.

- Etiquetas: puede añadir etiquetas para organizar sus AWS recursos, realizar un seguimiento de los costes y controlar el acceso.

  Si se especifican etiquetas en la acción `CreateDeliveryStream`, Amazon Data Firehose realiza una autorización adicional en la acción `firehose:TagDeliveryStream` para verificar que los usuarios tengan permisos para crear etiquetas. Si no concede este permiso, las solicitudes

para crear nuevos flujos de Firehose con etiquetas de recursos de IAM fallarán con un resultado `AccessDeniedException` como el siguiente.

```
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.
```

En el ejemplo siguiente, se muestra una política que permite que los usuarios creen un flujo de Firehose y apliquen etiquetas.

Una vez que haya elegido la configuración de copias de seguridad y la configuración avanzada, revise sus opciones y, a continuación, seleccione Create Firehose stream (Crear flujo de Firehose).

El nuevo flujo de Firehose se encontrará en el estado Creating (En creación) durante algún tiempo antes de estar disponible. Podrá comenzar a enviar los datos de su productor cuando el flujo de Firehose pase al estado Active (Activo).

Prueba de flujos de Firehose con datos de ejemplo

Puede utilizar la Consola de administración de AWS para incorporar datos de tableros de cotizaciones simulados. La consola ejecuta un script en su navegador para colocar registros de ejemplo en el flujo de Firehose. Esto le permite probar la configuración del flujo de Firehose sin tener que generar sus propios datos de prueba.

A continuación, mostramos un ejemplo de datos simulados:

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

Tenga en cuenta que se aplicará la tarifa estándar de Amazon Data Firehose cuando el flujo de Firehose transmita los datos, pero la generación de datos no implica cargos. Para detener este cargo, puede detener el flujo de muestra desde la consola en cualquier momento.

Requisitos previos

Antes de comenzar, cree un flujo de Firehose. Para obtener más información, consulte Tutorial: Crear un flujo de Firehose desde la consola.

Prueba con Amazon S3

Utilice el siguiente procedimiento para probar el flujo de Firehose con Amazon Simple Storage Service (Amazon S3) como destino.

Prueba de un flujo de Firehose con Amazon S3

1. Abra la consola de Firehose en https://console.aws.amazon.com/firehose/.
2. Elija un flujo de Firehose activo. El flujo de Firehose debe estar en estado Activo antes de que pueda empezar a enviar datos.
3. En Test with demo data, seleccione Start sending demo data para generar datos de tableros de cotizaciones simulados.
4. Siga las instrucciones que aparecen en pantalla para verificar que los datos se están entregando al bucket de S3. Tenga en cuenta que posiblemente pasen unos minutos hasta que los objetos aparezcan en el bucket. Esto dependerá de la configuración de búfer del bucket.

5. Una vez terminada la prueba, seleccione Stop sending demo data para detener los cargos por uso.

Comprobación con Amazon Redshift

Utilice el siguiente procedimiento para probar el flujo de Firehose con Amazon Redshift como destino.

Prueba de un flujo de Firehose con Amazon Redshift

1. El flujo de Firehose espera que haya una tabla en el clúster de Amazon Redshift. Conéctese a Amazon Redshift a través de una interfaz de SQL y ejecute la siguiente declaración para crear una tabla que acepte los datos de muestra.

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

2. Abra la consola de Firehose en https://console.aws.amazon.com/firehose/.
3. Elija un flujo de Firehose activo. El flujo de Firehose debe estar en estado Activo antes de que pueda empezar a enviar datos.
4. Edite los detalles de destino del flujo de Firehose para que apunte a la nueva tabla `firehose_test_table`.
5. En Test with demo data, seleccione Start sending demo data para generar datos de tableros de cotizaciones simulados.
6. Siga las instrucciones que aparecen en pantalla para verificar que los datos se están entregando a la tabla. Tenga en cuenta que posiblemente pasen unos minutos hasta que los objetos aparezcan en la tabla. Esto dependerá de la configuración de búfer.
7. Una vez terminada la prueba, seleccione Stop sending demo data para detener los cargos por uso.
8. Edite los detalles de destino del flujo de Firehose para que apunte a otra tabla.
9. Elimine la `firehose_test_table` tabla (opcional).

Prueba con OpenSearch Service

Utilice el siguiente procedimiento para probar el flujo de Firehose con Amazon OpenSearch Service como destino.

Prueba de un flujo de Firehose con OpenSearch Service

1. Abra la consola de Firehose en https://console.aws.amazon.com/firehose/.
2. Elija un flujo de Firehose activo. El flujo de Firehose debe estar en estado Activo antes de que pueda empezar a enviar datos.
3. En Test with demo data, seleccione Start sending demo data para generar datos de tableros de cotizaciones simulados.
4. Siga las instrucciones que aparecen en pantalla para verificar que los datos se están entregando en su dominio de OpenSearch Service. Para obtener más información, consulte Searching Documents in an OpenSearch Service Domain en la Guía para desarrolladores de Amazon OpenSearch Service.
5. Una vez terminada la prueba, seleccione Stop sending demo data para detener los cargos por uso.

Prueba con Splunk

Utilice el siguiente procedimiento para probar el flujo de Firehose con Splunk como destino.

Prueba de un flujo de Firehose con Splunk

1. Abra la consola de Firehose en https://console.aws.amazon.com/firehose/.
2. Elija un flujo de Firehose activo. El flujo de Firehose debe estar en estado Activo antes de que pueda empezar a enviar datos.
3. En Test with demo data, seleccione Start sending demo data para generar datos de tableros de cotizaciones simulados.
4. Compruebe si los datos se están entregando al índice de Splunk. Ejemplos de términos de búsqueda de Splunk: `sourcetype="aws:firehose:json"` y `index="name-of-your-splunk-index"`. Para obtener más información acerca de cómo buscar eventos en Splunk, consulte Search Manual en la documentación de Splunk.

Si los datos de prueba no aparecen en el índice de Splunk, compruebe si hay eventos de error en el bucket de Amazon S3. Consulte también Datos no entregados a Splunk.

5. Una vez terminada la prueba, seleccione Stop sending demo data para evitar incurrir en cargos por uso.

Pruebas con las tablas de Apache Iceberg

Utilice el siguiente procedimiento para probar el flujo de Firehose con las tablas de Apache Iceberg como destino.

Pruebas de un flujo de Firehose con tablas de Apache Iceberg

1. Abra la consola de Firehose en https://console.aws.amazon.com/firehose/.
2. Elija un flujo de Firehose activo. El flujo de Firehose debe estar en estado Activo antes de que pueda empezar a enviar datos.
3. En Test with demo data, seleccione Start sending demo data para generar datos de tableros de cotizaciones simulados.
4. Siga las instrucciones que aparecen en pantalla para verificar que los datos se están entregando a las tablas de Apache Iceberg. Tenga en cuenta que posiblemente pasen unos minutos hasta que los objetos aparezcan en el bucket. Esto dependerá de la configuración de búfer.
5. Si los datos de prueba no aparecen en las tablas de Apache Iceberg, compruebe si hay eventos de error en el bucket de Amazon S3.
6. Una vez terminada la prueba, seleccione Stop sending demo data para evitar incurrir en cargos por uso.

Enviar datos a un flujo de Firehose

En esta sección, se describe cómo puede usar diferentes orígenes de datos para enviar datos al flujo de Firehose. Si es la primera vez que utiliza Amazon Data Firehose, le recomendamos familiarizarse antes con los conceptos y los términos que encontrará en ¿Qué es Amazon Data Firehose?.

> **Note**
>
> Algunos servicios de AWS solo pueden enviar mensajes y eventos a un flujo de Firehose que esté en la misma región. Si el flujo de Firehose no aparece como opción al configurar un destino para Registros de Amazon CloudWatch, Eventos de CloudWatch o AWS IoT, compruebe que el flujo de Firehose esté en la misma región que los demás servicios. Para obtener información sobre los puntos de conexión de servicio de cada región, consulte los puntos de conexión de Amazon Data Firehose.

Puede enviar datos al flujo de Firehose desde los siguientes orígenes de datos.

Temas

Configurar el agente de Kinesis para enviar datos

El agente de Amazon Kinesis es una aplicación de software de Java independiente que sirve de implementación de referencia para mostrar cómo se pueden recopilar y enviar datos a Firehose. El agente monitoriza constantemente un conjunto de archivos y envía nuevos datos a su flujo de Firehose. El agente le muestra cómo encargarse de la rotación de archivos, cómo crear puntos de control y cómo realizar reintentos cuando se producen errores. También le muestra cómo entregar todos los datos de manera confiable, puntual y sencilla. Además, le muestra cómo emitir métricas de CloudWatch para que supervise y solucione mejor los problemas que surjan en el proceso de streaming. Para obtener más información, awslabs/amazon-kinesis-agent.

De forma predeterminada, los registros de cada archivo se analizan en función del carácter de nueva línea (`'\n'`). Sin embargo, el agente también se puede configurar para analizar registros multilínea (consulte Especificar las opciones de configuración del agente).

Puede instalar el agente en entornos de servidor basados en Linux, como servidores web, de registro o de base de datos. Después de instalar el agente, configúrelo especificando los archivos que desee monitorizar y el flujo de Firehose de los datos. Una vez configurado, el agente recopila datos de los archivos de forma duradera y los envía de forma confiable al flujo de Firehose.

Requisitos previos

Antes de usar el agente de Kinesis, asegúrese de cumplir los siguientes requisitos previos.

- Su sistema operativo debe ser Amazon Linux o Red Hat Enterprise Linux, versión 7 o posterior.
- La versión 2.0.0 o posterior del agente se ejecuta con la versión 1.8 o posterior de JRE. La versión 1.1.x del agente se ejecuta con la versión 1.7 o posterior de JRE.
- Si utiliza Amazon EC2 para ejecutar el agente, lance la instancia de EC2.
- El rol de IAM o las credenciales de AWS que especifique deben tener permiso para llevar a cabo la operación PutRecordBatch de Amazon Data Firehose para que el agente envíe datos a su flujo de Firehose. Si habilita el seguimiento de CloudWatch para el agente, también se necesita permiso para llevar a cabo la operación PutMetricData de CloudWatch. Para obtener más información, consulte Control del acceso con Amazon Data Firehose, Supervisión del estado del agente de Kinesis y Autenticación y control de acceso de Amazon CloudWatch.

Administrar las credenciales de AWS

Administre las credenciales de AWS con uno de los siguientes métodos:

- Cree un proveedor de credenciales personalizado. Para obtener más información, consulte the section called “Crear proveedores de credenciales personalizados”.
- Especifique un rol de IAM al lanzar la instancia EC2.
- Especifique las credenciales de AWS al configurar el agente (consulte las entradas de `awsAccessKeyId` y `awsSecretAccessKey` en la tabla de configuración en the section called “Especificar las opciones de configuración del agente”).
- Edite `/etc/sysconfig/aws-kinesis-agent` para especificar la región de AWS y las claves de acceso de AWS.

- Si la instancia de EC2 está en otra cuenta de AWS, cree un rol de IAM para proporcionar acceso al servicio Amazon Data Firehose. Especifique ese rol al configurar el agente (assumeRoleARN y assumeRoleExternalId). Utilice uno de los métodos anteriores para especificar las credenciales de AWS de un usuario en la otra cuenta que tenga permiso para asumir este rol.

Crear proveedores de credenciales personalizados

Puede crear un proveedor de credenciales personalizado e indicar su nombre de clase y ruta de archivo jar al agente de Kinesis en las siguientes opciones de configuración: `userDefinedCredentialsProvider.classname` y `userDefinedCredentialsProvider.location`. Para obtener las descripciones de estas dos opciones de configuración, consulte the section called “Especificar las opciones de configuración del agente”.

Para crear un proveedor de credenciales personalizado, defina una clase que implemente la interfaz `AWSCredentialsProvider`, como la del ejemplo siguiente.

```
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() {
    }
}
```

Su clase debe tener un constructor sin argumentos.

AWS invoca periódicamente el método de actualización para obtener credenciales actualizadas. Si desea que el proveedor de credenciales proporcione credenciales diferentes a lo largo de su vida útil, incluya el código para actualizar las credenciales en este método. También puede dejar este método vacío si desea un proveedor de credenciales que ofrezca credenciales estáticas (no cambiantes).

Descargar e instalar el agente

Primero, conéctese a la instancia. Para obtener más información, consulte Conexión a una instancia en la Guía del usuario de Amazon EC2. Si tiene problemas para conectarse, consulte Solución de problemas con la conexión a la instancia en la Guía del usuario de Amazon EC2.

A continuación, instale el agente siguiendo uno de los siguientes métodos.

- Configuración del agente desde los repositorios de Amazon Linux

  Este método solo funciona para instancias de Amazon Linux. Utilice el siguiente comando:

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

  La versión 2.0.0 o posterior del agente se instala en equipos con el sistema operativo Amazon Linux 2 (AL2). Esta versión del agente requiere la versión 1.8 o posterior de Java. Si la versión de Java requerida aún no está presente, el proceso de instalación del agente la instala. Para obtener más información sobre Amazon Linux 2, consulte https://aws.amazon.com/amazon-linux-2/.
- Configuración del agente desde el repositorio de Amazon S

  Este método funciona para Red Hat Enterprise Linux, así como para las instancias de Amazon Linux 2, ya que instala el agente desde el repositorio disponible públicamente. Utilice el siguiente comando para descargar e instalar la versión más reciente de la versión 2.x.x del agente:

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

  Para instalar una versión concreta del agente, especifique el número de versión en el comando. Por ejemplo, el siguiente comando instala la versión 2.0.1 del agente.

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

Si tiene Java 1.7 y no quiere actualizar la versión, puede descargar la versión 1.x.x del agente, que es compatible con Java 1.7. Por ejemplo, para descargar la versión 1.1.6 del agente, puede utilizar el comando siguiente:

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

Puede descargar el agente más reciente con el siguiente comando

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

- Configuración del agente desde el repositorio de GitHub

  1. En primer lugar, asegúrese de que tiene instalada la versión de Java requerida, en función de la versión del agente.
  2. Descargue el agente del repositorio de GitHub awslabs/amazon-kinesis-agent.
  3. Instale el agente. Para ello, diríjase al directorio de descargas y ejecute el siguiente comando:

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

- Configuración del agente en un contenedor de Docker

  El agente de Kinesis también puede ejecutarse en un contenedor además de a través de la base de contenedores amazonlinux. Utilice el siguiente Dockerfile y ejecute `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"]
```

Configurar e iniciar el agente

Configuración e inicio del agente

1. Abra y edite el archivo de configuración (como superusuario si utiliza permisos de acceso de archivo predeterminado): `/etc/aws-kinesis/agent.json`

   En este archivo de configuración, especifique los archivos (`"filePattern"`) desde los que el agente deberá recopilar datos y el nombre del flujo de Firehose (`"deliveryStream"`) al que deberá enviarlos. El nombre de archivo es un patrón y el agente reconoce las rotaciones de archivos. No puede rotar más de un archivo ni crear más de uno nuevo por segundo. El agente utiliza la marca temporal de la creación de archivos para determinar los archivos de los que se debe hacer un seguimiento y que se deben poner en cola en el flujo de Firehose. Crear nuevos archivos o rotar los archivos más de una vez por segundo impide al agente diferenciarlos correctamente.

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

   La región de AWS predeterminada es `us-east-1`. Si utiliza una región diferente, añada el ajuste `firehose.endpoint` al archivo de configuración y especifique el punto de enlace de la región. Para obtener más información, consulte Especificar las opciones de configuración del agente.

2. Comience el agente de forma manual:

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

3. Configure el agente para iniciarse al arrancar el sistema (opcional):

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

El agente ya está se ejecutando como un servicio de sistema en segundo plano. Monitoriza constantemente los archivos especificados y envía datos al flujo de Firehose especificado. La auditoría de actividad se registra en `/var/log/aws-kinesis-agent/aws-kinesis-agent.log`.

Especificar las opciones de configuración del agente

El agente admite dos opciones de configuración obligatorias, `filePattern` y `deliveryStream`, además de configuraciones opcionales para activar características adicionales. Las opciones de configuración obligatorias y opcionales se especifican en `/etc/aws-kinesis/agent.json`.

Cada vez que cambie el archivo de configuración, debe detener y comenzar el agente con los siguientes comandos:

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

También puede hacerlo con el siguiente comando:

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

Las opciones de configuración generales son las siguientes.

Opción de configuración	Descripción
`assumeRoleARN`	El Nombre de recurso de Amazon (ARN) de la función que debe asumir el usuario. Para obtener más información, consulte Delegación del acceso entre cuentas de AWS mediante roles de IAM en la Guía del usuario de IAM.
`assumeRoleExternalId`	Un identificador opcional que determina quién puede asumir el rol. Para obtener más información, consulte Cómo utilizar un ID externo en la Guía del usuario de IAM.
`awsAccessKeyId`	ID de clave de acceso de AWS que anula las credenciales predeterminadas. Este ajuste tiene prioridad sobre los demás proveedores de credenciales.

Opción de configuración	Descripción
`awsSecretAccessKey`	Clave de secreta de AWS que anula las credenciales predeterminadas. Este ajuste tiene prioridad sobre los demás proveedores de credenciales.
`cloudwatch.emitMetrics`	Permite al agente emitir métricas a CloudWatch si se establece (true). Predeterminado: true
`cloudwatch.endpoint`	Punto de conexión regional de CloudWatch. Valor predeterminado: `monitoring.us-east-1.amazonaws.com`
`firehose.endpoint`	Punto de conexión regional de Amazon Data Firehose. Valor predeterminado: `firehose.us-east-1.amazonaws.com`
`sts.endpoint`	Punto de conexión regional de AWS Security Token Service. Valor predeterminado: `https://sts.amazonaws.com`
`userDefinedCredentialsProvider.classname`	Si define un proveedor de credenciales personalizado, proporcione su nombre de clase completo mediante esta configuración. No incluya `.class` al final del nombre de la clase.
`userDefinedCredentialsProvider.location`	Si define un proveedor de credenciales personalizado, utilice esta configuración para especificar la ruta absoluta del archivo jar que contiene el proveedor de credenciales personalizado. El agente también busca el archivo jar en la siguiente ubicación: `/usr/share/aws-kinesis-agent/lib/` .

Las opciones de configuración de flujo son las siguientes.

Opción de configuración	Descripción
`aggregatedRecordSizeBytes`	Para generar los registros de agregación del agente y ponerlos en el flujo de Firehose en una operación, especifique esta opción. Establezca el tamaño que desea que tenga el registro agregado antes de que el agente lo añada al flujo de Firehose. Valor predeterminado: 0 (sin agregación)
`dataProcessingOptions`	La lista de opciones de procesamiento aplicadas a cada registro analizado antes de enviarlo al flujo de Firehose. Las opciones de procesamiento se realizan en el orden especificado. Para obtener más información, consulte Preprocesar los datos con los agentes.
`deliveryStream`	[Obligatorio] El nombre del flujo de Firehose.
`filePattern`	[Obligatorio] Un glob para los archivos que deben ser monitorizados por el agente. Cualquier archivo que coincida con este patrón es seleccionado y monitorizado automáticamente por el agente. En todos los archivos que coincidan con este patrón, conceda permisos de lectura a `aws-kinesis-agent-user` . En el directorio que contiene los archivos, conceda permisos de lectura y ejecución a `aws-kinesis-agent-user` . **Important** El agente recoge cualquier archivo que coincida con este patrón. Para asegurarse de que el agente no recoge registros no deseados, seleccione este patrón con precaución.
`initialPosition`	La posición inicial desde la que el archivo comenzó a ser analizado. Los valores válidos son `START_OF_FILE` y `END_OF_FILE` . Valor predeterminado: `END_OF_FILE`
`maxBufferAgeMillis`	El tiempo máximo, en milisegundos, durante el cual el agente almacena los datos en búfer antes de enviarlos al flujo de Firehose.

Opción de configuración	Descripción
	Rango de valores: 1000 - 900 000 (de 1 segundo a 15 minutos) Valor predeterminado: 60 000 (1 minuto)
`maxBufferSizeBytes`	El tamaño máximo, en bytes, que el agente almacena en búfer antes de enviarlos al flujo de Firehose. Rango de valores: 1 - 4 194 304 (4 MB) Valor predeterminado: 4 194 304 (4 MB)
`maxBufferSizeRecords`	La cantidad máxima de registros en datos que el agente almacena en búfer antes de enviarlos al flujo de Firehose. Rango de valores: 1 - 500 Predeterminado: 500
`minTimeBetweenFilePollsMillis`	El intervalo de tiempo, en milisegundos, en el que el agente sondea y analiza los archivos monitorizados para identificar datos nuevos. Intervalo de valores: 1 o más Predeterminado: 100
`multiLineStartPattern`	El patrón para identificar el comienzo de un registro. Un registro consta de una línea que coincide con el patrón y de líneas siguientes que no coinciden con el patrón. Los valores válidos son expresiones regulares . De forma predeterminada, cada línea en los archivos de registro se analiza como un registro.
`skipHeaderLines`	La cantidad de líneas de los archivos monitorizados, a partir de la primera, que el agente debe omitir en el momento de analizarlos. Intervalo de valores: 0 o más Cantidad predeterminada: 0 (cero)

Opción de configuración	Descripción
`truncatedRecordTerminator`	Cadena que utiliza el agente para truncar un registro analizado cuando su tamaño supera el límite de tamaño de registros de Amazon Data Firehose. (1000 KB) Valor predeterminado: `'\n'` (línea nueva)

Configurar múltiples flujos y directorios de archivos

Puede configurar el agente para que monitorice varios directorios de archivos y envíe datos a varias secuencias especificando varias opciones de configuración de secuencia. En el siguiente ejemplo de configuración, el agente supervisa dos directorios de archivos y envía datos a un flujo de datos de Kinesis y a un flujo de Firehose, respectivamente. Puede especificar diferentes puntos de conexión para flujos de datos de Kinesis y Amazon Data Firehose para que el flujo de datos y el flujo de Firehose no tengan que estar en la misma región.

```
{
    "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 obtener más información detallada sobre el uso del agente con Amazon Kinesis Data Streams, consulte Writing to Amazon Kinesis Data Streams with Kinesis Agent.

Preprocesar los datos con los agentes

El agente puede preprocesar los registros analizados de los archivos monitorizados antes de enviarlos al flujo de Firehose. Para habilitar esta característica, añada la opción de configuración `dataProcessingOptions` al flujo de archivos. Puede añadir una o más opciones de procesamiento que se ejecutarán en el orden especificado.

El agente es compatible con las siguientes opciones de procesamiento. Dado que el agente es de código abierto, el usuario puede desarrollar y ampliar sus opciones de procesamiento. Puede descargar el agente desde Kinesis Agent.

Opciones de procesamiento

`SINGLELINE`

Elimina los caracteres de nueva línea y los espacios situados al principio y al final de las líneas para convertir un registro multilínea en un registro de una sola línea.

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

`CSVTOJSON`

Convierte un registro con un formato separado mediante delimitadores al formato JSON.

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

`customFieldNames`

[Obligatorio] Los nombres de campos utilizados como claves en cada par de valores de clave JSON. Por ejemplo, si especifica `["f1", "f2"]`, el registro "v1, v2" se convierte en `{"f1":"v1","f2":"v2"}`.

`delimiter`

La cadena utilizada como delimitador en el registro. El valor predeterminado es una coma (,).

LOGTOJSON

Convierte un registro con un formato de registro en un registro con formato JSON. Los formatos de registro admitidos son Apache Common Log, Apache Combined Log, Apache Error Log y RFC3164 Syslog.

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

logFormat

[Obligatorio] El formato de entrada del registro. Los valores posibles son los siguientes:

- COMMONAPACHELOG: formato común de registro de Apache. Cada entrada de registro sigue el siguiente patrón de forma predeterminada: "%{host} %{ident} %{authuser} [%{datetime}] \"%{request}\" %{response} %{bytes}".
- COMBINEDAPACHELOG: formato combinado de registro de Apache. Cada entrada de registro sigue el siguiente patrón de forma predeterminada: "%{host} %{ident} %{authuser} [%{datetime}] \"%{request}\" %{response} %{bytes} %{referrer} %{agent}".
- APACHEERRORLOG: formato de registro de errores de Apache. Cada entrada de registro sigue el siguiente patrón de forma predeterminada: "[%{timestamp}] [%{module}: %{severity}] [pid %{processid}:tid %{threadid}] [client: %{client}] %{message}".
- SYSLOG: formato RFC3164 de Syslog. Cada entrada de registro sigue el siguiente patrón de forma predeterminada: "%{timestamp} %{hostname} %{program}[%{processid}]: %{message}".

matchPattern

Sobrescribe el patrón predeterminado del formato de log especificado. Utilice esta configuración para extraer valores de entradas de log si utilizan un formato personalizado. Si especifica matchPattern, también debe especificar customFieldNames.

`customFieldNames`

Los nombres de campos utilizados como claves en cada par de valores de clave JSON. Utilice esta opción para definir nombres de campos para valores extraídos de `matchPattern`, o sobrescriba los nombres de campos de los formatos de logs predefinidos.

Example : Configuración LOGTOJSON

Este es un ejemplo de configuración `LOGTOJSON` de una entrada de registro en Formato común de registro de Apache convertida a formato JSON:

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

Antes de la conversión:

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

Después de la conversión:

```
{"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 : Configuración LOGTOJSON con campos personalizados

Este es otro ejemplo de configuración `LOGTOJSON`:

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

Con esta configuración, la misma entrada de registro con Formato común de registro de Apache del ejemplo anterior se convierte a formato JSON de la siguiente manera:

```
{"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 : Convertir una entrada de registro con Formato común de registro de Apache

La siguiente configuración de secuencia convierte la entrada de registro común de Apache en un registro de una línea con formato JSON:

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

Example : Convertir registros multilínea

La siguiente configuración de flujo analiza aquellos registros multilínea cuya primera línea comience por "`[SEQUENCE=`". Primero, cada registro se convierte en un registro de una línea. Después, se extraen los valores del registro basándose en tabulaciones delimitadoras. Finalmente, los valores extraídos se asignan a valores `customFieldNames` específicos para formar un registro de una línea en 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 : Configuración LOGTOJSON con patrón de coincidencia

Este es un ejemplo de una configuración de entrada de registro con Formato común de registro de Apache LOGTOJSON convertida a formato JSON con el 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 de la conversión:

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

Después de la conversión:

```
{"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"}
```

Utilizar comandos CLI de agente comunes

En la siguiente tabla, se proporciona un conjunto de casos de uso comunes y los comandos correspondientes para trabajar con el agente de AWS Kinesis.

Caso de uso	Comando
Iniciar automáticamente al agente al arrancar el sistema	`sudo chkconfig aws-kinesis-agent on`
Comprobar el estado del agente	`sudo service aws-kinesis-agent status`
Detener el agente	`sudo service aws-kinesis-agent stop`
Leer el archivo de registro del agente desde esta ubicación	`/var/log/aws-kinesis-agent/aws-kinesis-agent.log`
Desinstalar el agente	`sudo yum remove aws-kinesis-agent`

Solucionar problemas al enviar desde el agente de Kinesis

En esta tabla, se proporciona información y soluciones para los problemas más comunes que se presentan al utilizar el agente de Amazon Kinesis.

Problema	Solución
¿Por qué el agente de Kinesis no funciona en Windows?	El agente de Kinesis para Windows es un software diferente del agente de Kinesis para plataformas Linux.
¿Por qué se ralentiza el agente de Kinesis o aumenta el valor de `RecordSendErrors` ?	Normalmente esto se debe a la limitación de Kinesis. Compruebe la métrica `WriteProvisionedThroughputExceeded` para flujos de datos de Kinesis o la métrica `ThrottledRecords` para flujos de Firehose. Cualquier aumento desde 0 en estas métricas indica que es necesario aumentar los límites de flujos. Para obtener más información, consulte Límites de flujos de datos de Kinesis y flujos de Firehose.

Problema	Solución
	Una vez que descarte la limitación, compruebe si el agente de Kinesis está configurado para seguir una gran cantidad de archivos pequeños. Se produce un retraso cuando el agente de Kinesis sigue un archivo nuevo, por lo que el agente de Kinesis debería seguir una pequeña cantidad de archivos de mayor tamaño. Intente consolidar los archivos de registro en archivos más grandes.
¿Cómo resolver las excepciones `java.lang.OutOfMemoryError` ?	Esto sucede cuando el agente de Kinesis no tiene memoria suficiente para gestionar la carga de trabajo actual. Intente aumentar `JAVA_START_HEAP` y `JAVA_MAX_HEAP` en `/usr/bin/start-aws-kinesis-agent` y reinicie el agente.
¿Cómo resolver las excepciones `IllegalStateException : connection pool shut down` ?	El agente de Kinesis no tiene suficientes conexiones para gestionar la carga de trabajo actual. Intente aumentar `maxConnections` y `maxSendingThreads` en los ajustes generales de configuración del agente en `/etc/aws-kinesis/agent.json` . El valor predeterminado para estos campos es 12 veces los procesadores de tiempo de ejecución disponibles. Consulte AgentConfiguration.java para obtener más información sobre los ajustes de configuración avanzada del agente.
¿Cómo puedo depurar otro problema con el agente de Kinesis?	Los registros de nivel `DEBUG` pueden habilitarse en `/etc/aws-kinesis/log4j.xml` .
¿Cómo debo configurar el agente de Kinesis?	Cuanto menor sea el valor de `maxBufferSizeBytes` , más frecuentemente enviará datos el agente de Kinesis. Esto puede ser bueno ya que disminuye el tiempo de entrega de los registros, pero también aumenta las solicitudes por segundo a Kinesis.

Problema	Solución
¿Por qué el agente de Kinesis envía registros duplicados?	Esto ocurre debido a una mala configuración en el seguimiento de archivos. Asegúrese de que cada `fileFlow's filePattern` solo coincida con un archivo. Esto también puede ocurrir si el modo `logrotate` que se está utilizando está en modo `copytruncate` . Intente cambiar al modo predeterminado o al de creación para evitar la duplicación. Para obtener más información sobre la gestión de registros duplicados, consulte Handling Duplicate Records.

Enviar datos con el SDK de AWS

Puede utilizar la API de Amazon Data Firehose para enviar datos a un flujo de Firehose con AWS SDK para Java, .NET, Node.js, Python o Ruby. Si es la primera vez que utiliza Amazon Data Firehose, le recomendamos familiarizarse antes con los conceptos y los términos que encontrará en ¿Qué es Amazon Data Firehose?. Para obtener más información, consulte Comience a crear con Amazon Web Services.

Estos ejemplos no representan códigos listos para producción, ya que no comprueban todas las excepciones posibles ni toman en cuenta todas las consideraciones de seguridad y desempeño posibles.

La API de Amazon Data Firehose ofrece dos operaciones para enviar datos al flujo de Firehose: PutRecord y PutRecordBatch. `PutRecord()` envía un registro de datos en una llamada y `PutRecordBatch()` puede enviar varios registros de datos en una llamada.

Operaciones individuales de escritura con PutRecord

Para incluir datos, solo se necesita el nombre del flujo de Firehose y un búfer de bytes (<= 1000 KB). Como Amazon Data Firehose agrupa en lotes varios registros antes de cargar el archivo en Amazon S3, es posible que desee agregar un separador de registros. Utilice el siguiente código para incluir los registros de datos de uno en uno en un flujo de Firehose:

```
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 obtener más contexto de código, consulte el código de muestra que se incluye en AWS SDK. Para obtener información sobre la sintaxis de las solicitudes y respuestas, consulte el tema correspondiente en Firehose API Operations.

Operaciones de escritura por lotes con PutRecordBatch

Para incluir datos, solo se necesita el nombre del flujo de Firehose y una lista de registros. Como Amazon Data Firehose agrupa en lotes varios registros antes de cargar el archivo en Amazon S3, es posible que desee agregar un separador de registros. Utilice el siguiente código para incluir los registros de datos por lotes en un flujo de Firehose:

```
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 obtener más contexto de código, consulte el código de muestra que se incluye en AWS SDK. Para obtener información sobre la sintaxis de las solicitudes y respuestas, consulte el tema correspondiente en Firehose API Operations.

Enviar los registros de CloudWatch a Firehose

Los eventos de registro de CloudWatch se pueden enviar a Firehose mediante los filtros de suscripción de CloudWatch. Para obtener más información, consulte Filtros de suscripción con Amazon Data Firehose.

Los eventos de registro de CloudWatch se envían a Firehose en formato gzip comprimido. Si quiere enviar eventos de registro descomprimidos a los destinos de Firehose, puede utilizar la característica de descompresión de Firehose para descomprimir registros de CloudWatch automáticamente.

> **Important**
>
> Actualmente, Firehose no admite la entrega de registros de CloudWatch en el destino de Amazon OpenSearch Service porque Amazon CloudWatch combina varios eventos de registro en un registro de Firehose, y Amazon OpenSearch Service no puede aceptar varios eventos de registro en un registro. Como alternativa, puede considerar la posibilidad de utilizar un filtro de suscripción para Amazon OpenSearch Service en Registros de CloudWatch.

Descomprimir los registros de CloudWatch

Si utiliza Firehose para entregar registros de CloudWatch y quiere entregar datos descomprimidos a su destino de flujo de Firehose, utilice la conversión de formato de datos de Firehose (Parquet, ORC) o la partición dinámica. Debe activar la descompresión para el flujo de Firehose.

Puede activar la descompresión mediante Consola de administración de AWS, AWS Command Line Interface o los SDK de AWS.

> **Note**
>
> Si habilita la característica de descompresión en un flujo, utilícelo exclusivamente para los filtros de suscripciones a registros de CloudWatch y no para registros ofrecidos. Si habilita la característica de descompresión en un flujo que se utiliza para ingerir tanto registros de CloudWatch como registros ofrecidos, se produce un error en la ingestión de registros ofrecidos a Firehose. Esta característica de descompresión solo está disponible para registros de CloudWatch.

Extraer el mensaje tras la descompresión de registros de CloudWatch

Cuando habilita la descompresión, tiene la opción de habilitar también la extracción de mensajes. Al utilizar la extracción de mensajes, Firehose filtra todos los metadatos (como el propietario, el grupo de registro, el flujo de registro y otros) de los registros descomprimidos de CloudWatch y

entrega solo el contenido de los campos de mensajes. Si envía datos a un destino de Splunk, debe activar la extracción de mensajes para que Splunk analice los datos. Los siguientes son ejemplos de resultados después de la descompresión con y sin extracción de mensajes.

Figura 1: Ejemplo de resultado después de la descompresión sin extracción del mensaje:

```
{
 "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: Ejemplo de resultado después de la descompresión con extracción del mensaje:

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

Habilitar la descompresión en un nuevo flujo de Firehose desde la consola

Para habilitar la descompresión en un nuevo flujo de Firehose mediante la Consola de administración de AWS

1. Inicie sesión en la Consola de administración de AWS y abra la consola de Kinesis en https://console.aws.amazon.com/kinesis.
2. Elija Amazon Data Firehose en el panel de navegación.
3. Seleccione Create Firehose stream (Crear flujo de Firehose).
4. En Choose source and destination (Elegir origen y destino)

   Origen

   La fuente del flujo de Firehose. Elija una de las siguientes fuentes:

   - Direct PUT: elija esta opción para crear un flujo de Firehose en el que las aplicaciones de los productores escriban directamente. Para obtener una lista de servicios de AWS, agentes y servicios de código abierto que se integran con Direct PUT en Firehose, consulte esta sección.
   - Flujo de Kinesis: seleccione esta opción para configurar un flujo de Firehose que utilice un flujo de datos de Kinesis como origen de datos. A continuación, puede usar Firehose para leer fácilmente los datos de un flujo de datos de Kinesis existente y cargarlos en los destinos. Para obtener más información, consulte Escritura en Firehose mediante Kinesis Data Streams.

   Destino

   Destino del flujo de Firehose. Elija una de las opciones siguientes:

   - Amazon S3
   - Splunk
5. En Firehose stream name (Nombre del flujo de Firehose), introduzca un nombre para el flujo.
6. (Opcional) En Transform records (Transformar registros):

   - En la sección Decompress source records from Amazon CloudWatch Logs (Descomprimir registros fuente de Registros de Amazon CloudWatch), seleccione Turn on decompression (Activar la descompresión).
   - Si desea utilizar la extracción de mensajes después de la descompresión, seleccione Turn on message extraction (Activar la extracción de mensajes).

Habilitar la descompresión en un flujo de Firehose existente

Esta sección presenta las instrucciones para habilitar la descompresión de los flujos de Firehose existentes. Abarca dos escenarios: los flujos con el procesamiento de Lambda deshabilitado y los flujos con el procesamiento de Lambda ya habilitado. Las siguientes secciones describen los procedimientos paso a paso para cada caso, como la creación o modificación de las funciones de Lambda, la actualización de la configuración de Firehose y la supervisión de las métricas de CloudWatch para garantizar la implementación correcta de la característica de descompresión Firehose integrada.

Habilitar la descompresión cuando el procesamiento de Lambda está deshabilitado

Para habilitar la descompresión en un flujo de Firehose existente con el procesamiento de Lambda deshabilitado, primero debe habilitar el procesamiento de Lambda. Esta condición solo es válida para los flujos existentes. Los siguientes pasos muestran la forma de habilitar la descompresión en los flujos existentes que no tienen habilitado el procesamiento de Lambda.

1. Creación de una función de Lambda. Puede crear una transferencia de registros ficticia o utilizar este esquema para crear una nueva función de Lambda.
2. Actualice su flujo de Firehose actual para habilitar el procesamiento de Lambda, y utilice la función de Lambda que creó para el procesamiento.
3. Una vez se actualice el flujo con la nueva función de Lambda, regrese a la consola de Firehose y habilite la descompresión.
4. Deshabilite el procesamiento de Lambda que habilitó en el paso 1. Ahora puede eliminar la clave que creó en el paso 1.

Habilitar la descompresión cuando el procesamiento de Lambda está habilitado

Si ya tiene un flujo de Firehose con una función de Lambda, puede sustituirlo por la característica de descompresión de Firehose para realizar la descompresión. Antes de continuar, revise el código de la función de Lambda para confirmar que solo realiza la descompresión o la extracción de mensajes. La salida de la función de Lambda debería tener un aspecto similar a los ejemplos que se muestran en la figura 1 o la figura 2. Si el resultado tiene un aspecto similar, puede reemplazar la función de Lambda siguiendo estos pasos.

1. Sustituya la función de Lambda actual por este esquema. La nueva función de Lambda del esquema detecta automáticamente si los datos entrantes están comprimidos o descomprimidos. Solo realiza la descompresión si los datos de entrada están comprimidos.
2. Active la descompresión con la opción de Firehose integrada para la descompresión.
3. Active las métricas de CloudWatch para el flujo de Firehose si aún no están habilitadas. Supervise la métrica `CloudWatchProcessorLambda_IncomingCompressedData` y espere a que esta cambie a cero. Esto confirma que todos los datos de entrada enviados a la función de Lambda están descomprimidos y que la función de Lambda ya no es necesaria.
4. Elimine la transformación de datos de Lambda porque ya no la necesita para descomprimir el flujo.

Desactivar la descompresión en el flujo de Firehose

Desactivar la descompresión en un flujo de datos con Consola de administración de AWS

1. Inicie sesión en la Consola de administración de AWS y abra la consola de Kinesis en https://console.aws.amazon.com/kinesis.
2. Elija Amazon Data Firehose en el panel de navegación.
3. Elija el flujo de Firehose que desea editar.
4. En la página Firehose stream details (Detalles del flujo de Firehose), seleccione la pestaña Configuration (Configuración).
5. En la sección Transform and convert records (Transformar y convertir registros), seleccione Edit (Editar).
6. En Decompress source records from Amazon CloudWatch Logs (Descomprimir registros fuente de Registros de Amazon CloudWatch), desactive la opción Turn on decompression (Activar la descompresión) y, a continuación, seleccione Save changes (Guardar cambios).

Solución de problemas de descompresión en Firehose

En la siguiente tabla, se muestra cómo Firehose gestiona los errores durante la descompresión y el procesamiento de los datos, incluida la entrega de registros a un bucket de S3 con errores, el registro de errores y la emisión de métricas. También se explica el mensaje de error devuelto en el caso de operaciones de colocación de datos no autorizadas.

Problema	Solución
¿Qué ocurre con los datos de origen en caso de que se produzca un error durante la descompresión?	Si Amazon Data Firehose no puede descomprimir el registro, este se entrega tal como está (en formato comprimido) al bucket de S3 con errores que especific ó durante la creación del flujo de Firehose. Junto con el registro, el objeto entregado también incluye el código de error y el mensaje de error, y estos objetos se entregará n en un prefijo de bucket de S3 llamado `decompres sion-failed` . Firehose seguirá procesando otros registros después de la descompresión fallida de un registro.
¿Qué ocurre con los datos de origen en caso de que se produzca un error en el proceso de procesamiento tras una descompresión satisfactoria?	Si Amazon Data Firehose produce un error en los pasos de procesamiento posteriores a la descompresión, como el particionamiento dinámico y la conversión de formatos de datos, el registro se entrega en formato comprimid o al bucket de S3 con errores que especificó durante la creación del flujo de Firehose. Junto con el registro, el objeto entregado también incluye el código de error y el mensaje de error.
¿Cómo se le informa en caso de que se produzca un error o una excepción?	En caso de que se produzca un error o una excepción durante la descompresión, si configura los registros de CloudWatch, Firehose registrará los mensajes de error en registros de CloudWatch. Además, Firehose envía métricas a CloudWatch que puede supervisar. Opcionalm ente, también puede crear alarmas basadas en métricas emitidas por Firehose.
¿Qué ocurre cuando las operacion es `put` no provienen de los registros de CloudWatch?	Cuando los `puts` de clientes no provienen de registros de CloudWatch, aparece el siguiente mensaje de error: `Put to Firehose failed for AccountId: <accountI D>, FirehoseName:  <firehosename> because the  request is not originating from allowed source  types.`

Problema	Solución
¿Qué métricas emite Firehose para la característica de descompresión?	Firehose emite métricas para la descompresión de todos los registros. Debe seleccionar el periodo (1 minuto), la estadística (suma) y el intervalo de fechas para obtener el número de `DecompressedRecords` con errores o satisfactorios, o de `DecompressedBytes` con errores o satisfactorios. Para obtener más información, consulte Métricas de descompresión de Registros de CloudWatch.

Enviar eventos de CloudWatch a Firehose

Puede configurar Amazon CloudWatch para enviar eventos a un flujo de Firehose; para ello, agregue un destino a una regla de eventos de CloudWatch.

Creación de un destino para una regla de eventos de CloudWatch que envíe eventos a un flujo de Firehose existente

1. Inicie sesión en la Consola de administración de AWS y abra la consola de CloudWatch en https://console.aws.amazon.com/cloudwatch/.
2. Seleccione Creación de regla.
3. En la página Step 1: Create rule (Paso 1: crear regla), en Targets (Destinos), elija Add target (Añadir destino) y, a continuación, elija Firehose stream (Flujo de Firehose).
4. Elija un flujo de Firehose existente.

Para obtener más información acerca de cómo crear reglas de Eventos de CloudWatch, consulte Getting Started with Amazon CloudWatch Events.

Configurar AWS IoT para enviar datos a Firehose

Es posible añadir una acción para configurar AWS IoT de modo que le envíe información a un flujo de Firehose.

Crear una acción que envíe eventos a un flujo de Firehose existente

1. Al crear una regla en la consola de AWS IoT, en la página Create a rule (Crear una regla), en Set one or more actions (Establecer una o varias acciones), elija Add action (Añadir acción).
2. Elija Enviar mensajes a un flujo de Amazon Kinesis Firehose.
3. Elija Configurar acción.
4. En Stream name (Nombre del flujo), elija un flujo de Firehose existente.
5. En Separator, seleccione un carácter de separación a insertar entre registros.
6. En Nombre del rol de IAM, elija un rol de IAM existente o elija Crear un nuevo rol.
7. Seleccione Agregar acción.

Para obtener más información acerca de la creación de reglas de AWS IoT, consulte AWS IoT Rule Tutorials.

Transformación de los datos de origen en Amazon Data Firehose

Amazon Data Firehose puede invocar su función de Lambda para transformar los datos de entrada de origen y entregarlos transformados en sus destinos. Puede habilitar la transformación de datos de Amazon Data Firehose al crear el flujo de Firehose.

Comprenda el flujo de transformación de datos

Al habilitar la transformación de datos de Firehose, Firehose almacena en búfer los datos de entrada. La sugerencia de tamaño del búfer oscila entre 0,2 MB y 3 MB. La sugerencia de tamaño del búfer de Lambda predeterminado es de 1 MB para todos los destinos, excepto Splunk y Snowflake. Para Splunk y Snowflake, la sugerencia de almacenamiento en búfer predeterminado es de 256 KB. La sugerencia del intervalo de almacenamiento en búfer de Lambda oscila entre 0 y 900 segundos. La sugerencia de intervalo de almacenamiento en búfer de Lambda predeterminado es de sesenta segundos para todos los destinos, excepto Snowflake. Para Snowflake, la sugerencia del intervalo de almacenamiento en búfer predeterminado es de 30 segundos. Para ajustar el tamaño del búfer, defina el parámetro ProcessingConfiguration de la API CreateDeliveryStream o UpdateDestination con el ProcessorParameter denominado `BufferSizeInMBs` y `IntervalInSeconds`. A continuación, Firehose invoca de forma sincrónica la función de Lambda especificada con cada lote almacenado en búfer con el modo de invocación sincrónica de AWS Lambda. Los datos transformados se envían de Lambda a Firehose. A continuación, Firehose envía esos datos al destino cuando se alcanza en el destino el tamaño de almacenamiento en búfer o el intervalo de almacenamiento en búfer especificado (lo que ocurra primero).

> **Important**
>
> El modo de invocación sincrónica de Lambda tiene un límite de tamaño de carga de 6 MB para la solicitud y la respuesta. Asegúrese de que el tamaño de almacenamiento en búfer para enviar la solicitud a la función es inferior o igual a 6 MB. Asegúrese también de que la respuesta devuelta por la función no sea superior a 6 MB.

Duración de la invocación de Lambda

Amazon Data Firehose admite un tiempo de invocación a Lambda de hasta 5 minutos. Si la función de Lambda tarda más de 5 minutos en completarse, aparece el siguiente error: Firehose encountered timeout errors when calling AWS Lambda. The maximum supported function timeout is 5 minutes.

Para obtener información sobre lo que hace Amazon Data Firehose si se produce un error como este, consulte the section called “Gestión de los errores en la transformación de datos ”.

Parámetros necesarios para la transformación de datos

Todos los registros transformados de Lambda deben contener los siguientes parámetros. De lo contrario, Amazon Data Firehose los rechaza y los trata como errores de transformación de datos.

For Kinesis Data Streams and Direct PUT

Se requieren los siguientes parámetros para todos los registros transformados de Lambda.

- `recordId` – El ID de registro se transfiere de Amazon Data Firehose a Lambda durante la invocación. El registro transformado debe contener el mismo ID de registro. Cualquier discrepancia entre el ID del registro original y el del transformado se trata como un error de transformación de datos.
- `result` – Es el estado de la transformación de los datos del registro. Los valores posibles son `Ok` si el registro se ha transformado correctamente, `Dropped` si la lógica de procesamiento ha omitido el registro intencionadamente y `ProcessingFailed` si el registro no se ha podido transformar. Si un registro tiene el estado `Ok` o `Dropped`, Amazon Data Firehose considera que se ha procesado correctamente. De lo contrario, Amazon Data Firehose considerará que no se ha procesado correctamente.
- `data` – Es la carga útil de datos transformados después codificarlos en base64.

  A continuación se presenta un ejemplo de salida de Lambda:

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

For Amazon MSK

Se requieren los siguientes parámetros para todos los registros transformados de Lambda.

- `recordId` – El ID de registro se transfiere desde Firehose hacia Lambda durante la invocación. El registro transformado debe contener el mismo ID de registro. Cualquier discrepancia entre el ID del registro original y el del transformado se trata como un error de transformación de datos.
- `result` – Es el estado de la transformación de los datos del registro. Los valores posibles son `Ok` si el registro se ha transformado correctamente, `Dropped` si la lógica de procesamiento ha omitido el registro intencionadamente y `ProcessingFailed` si el registro no se ha podido transformar. Si un registro tiene el estado `Ok` o `Dropped`, Firehose considera que se ha procesado correctamente. De lo contrario, Firehose considerará que no se ha procesado correctamente.
- `KafkaRecordValue` – Es la carga útil de datos transformados después codificarlos en base64.

  A continuación se presenta un ejemplo de salida de Lambda:

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

Esquemas de Lambda compatibles

Estos esquemas muestran cómo puede crear y utilizar las funciones de AWS Lambda para transformar los datos de los flujos de datos de Amazon Data Firehose.

Visualización de los esquemas disponibles en la consola de AWS Lambda

1. Inicie sesión en la Consola de administración de AWS y abra la consola AWS Lambda en https://console.aws.amazon.com/lambda/.
2. Elija Create function (Crear función) y, a continuación, elija Use a blueprint (Utilizar un proyecto).
3. En el campo Blueprints (Esquemas), busque la palabra clave `firehose` para encontrar los esquemas de Lambda de Amazon Data Firehose.

Lista de esquemas:

- Procesar registros enviados al flujo de Amazon Data Firehose (Node.js, Python)

  Este esquema muestra un ejemplo básico de cómo procesar los datos del flujo de datos de Firehose mediante AWS Lambda.

  Fecha de lanzamiento más reciente: noviembre de 2016.

  Notas de la versión: ninguna.
- Procesar CloudWatch Logs enviados a Firehose

  Este esquema está obsoleto. No utilice este esquema. Si los datos descomprimidos de CloudWatch Logs superan los 6 MB (límite de Lambda), es posible que se apliquen tarifas elevadas. Para obtener información sobre el procesamiento de los registros de CloudWatch enviados a Firehose, consulte Cómo escribir en Firehose mediante Registros de CloudWatch.
- Convertir los registros de flujos de Amazon Data Firehose en formato syslog en JSON (Node.js)

  En este esquema se muestra cómo convertir los registros de entrada en formato RFC3164 de Syslog en JSON.

  Fecha de lanzamiento más reciente: noviembre de 2016.

  Notas de la versión: ninguna.

Visualización de los esquemas que están disponibles en AWS Serverless Application Repository

1. Vaya a AWS Serverless Application Repository.
2. Elija Examinar todas las aplicaciones.
3. En el campo Applications (Aplicaciones) busque la palabra clave `firehose`.

También puede crear una función de Lambda sin utilizar un esquema. Consulte Introducción a AWS Lambda.

Gestión de los errores en la transformación de datos

Si se produce un error al invocar la función de Lambda por un tiempo de espera de red o porque se ha alcanzado el límite de invocaciones de Lambda, Amazon Data Firehose intenta realizar dicha

invocación tres veces más de forma predeterminada. Si la invocación no se realiza correctamente, Amazon Data Firehose omite ese lote de registros. los trata como registros que no se han podido procesar. Puede especificar o anular las opciones de reintento con las API CreateDeliveryStream o `UpdateDestination`. Este tipo de errores de invocación puede registrarse en Registros de Amazon CloudWatch. Para obtener más información, consulte Supervisión de Amazon Data Firehose mediante Registros de CloudWatch.

Si el estado de la transformación de datos de un registro es `ProcessingFailed`, Amazon Data Firehose lo trata como un registro que no ha podido procesarse correctamente. Los registros de este tipo de errores pueden emitirse desde la función de Lambda a Registros de Amazon CloudWatch. Para obtener más información, consulte Acceso a los registros de Amazon CloudWatch para AWS Lambda en la Guía para desarrolladores de AWS Lambda.

Si ocurre un error durante la transformación de datos, los registros que no se hayan podido procesar se entregan en el bucket de S3, en la carpeta `processing-failed`. Los registros tienen el siguiente formato:

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

`attemptsMade`

La cantidad de intentos de solicitud de invocación.

`arrivalTimestamp`

Hora a la que Amazon Data Firehose recibió el registro.

`errorCode`

Código de error HTTP devuelto por Lambda.

`errorMessage`

Mensaje de error HTTP devuelto Lambda.

`attemptEndingTimestamp`

Hora a la que Amazon Data Firehose dejó de intentar las invocaciones de Lambda.

`rawData`

Los datos de registros codificados en base64.

`lambdaArn`

El nombre de recurso de Amazon (ARN) de la función de Lambda.

Copia de seguridad de registros de origen

Amazon Data Firehose puede hacer una copia de seguridad de todos los registros no transformados en su bucket de S3 a la vez que entrega los registros transformados en su destino. Puede habilitar la copia de seguridad de registros de origen al crear o actualizar el flujo de Firehose. El backup de los registros de origen no se puede deshabilitar después de haberlo habilitado.

Partición de datos de streaming en Amazon Data Firehose

El particionamiento dinámico le permite particionar continuamente los datos de streaming en Firehose mediante claves dentro de los datos (por ejemplo, `customer_id` o `transaction_id`) y, a continuación, entregar los datos agrupados mediante estas claves en los prefijos correspondientes de Amazon Simple Storage Service (Amazon S3). Esto facilita la ejecución de análisis rentables y de alto rendimiento en datos de streaming en Amazon S3 mediante diversos servicios, como Amazon Athena, Amazon EMR, Amazon Redshift Spectrum y Amazon QuickSight. Además, AWS Glue puede llevar a cabo trabajos de extracción, transformación y carga (ETL) más sofisticados una vez que los datos de streaming particionados de forma dinámica se entreguen en Amazon S3, en casos de uso en los que se requiera un procesamiento adicional.

El particionamiento de los datos minimiza la cantidad de datos analizados, optimiza el rendimiento y reduce los costos de las consultas de análisis en Amazon S3. También aumenta el acceso granular a los datos. Los flujos de Firehose se utilizan tradicionalmente para capturar y cargar datos en Amazon S3. Para particionar un conjunto de datos de streaming con el objetivo de llevar a cabo análisis basados en Amazon S3, tendría que ejecutar aplicaciones de particionamiento entre buckets de Amazon S3 antes de hacer que los datos estén disponibles para su análisis, lo que podría resultar complicado o costoso.

Con el particionamiento dinámico, Firehose agrupa continuamente los datos en tránsito mediante claves de datos definidas de forma dinámica o estática y entrega los datos a prefijos individuales de Amazon S3 por clave. Esto reduce el tiempo de obtención de información en minutos u horas. También reduce los costos y simplifica las arquitecturas.

Temas

Habilitación del particionamiento dinámico en Amazon Data Firehose

Puede configurar el particionamiento dinámico para sus flujos de Firehose mediante la consola de administración, la CLI o las API de Amazon Data Firehose.

> **Important**
>
> Solo puede habilitar el particionamiento dinámico cuando crea un flujo de Firehose nuevo. No puede habilitar el particionamiento dinámico para un flujo de Firehose existente que aún no tenga habilitado el particionamiento dinámico.

Para consultar los pasos detallados sobre cómo habilitar y configurar el particionamiento dinámico mediante la consola de administración de Firehose mientras crea un flujo de Firehose nuevo, consulte Creación de un flujo de Amazon Firehose. Cuando lleve a cabo la tarea de especificar el destino de su flujo de Firehose, asegúrese de seguir los pasos de la sección Configuración de los ajustes de destino, ya que, actualmente, el particionamiento dinámico solo se admite en los flujos de Firehose que utilizan Amazon S3 como destino.

Una vez que se habilita el particionamiento dinámico en un flujo de Firehose activo, puede agregar nuevas claves de particionamiento dinámico o eliminar o actualizar las existentes y las expresiones de prefijos de S3 para actualizar la configuración. Tras la actualización, Firehose comienza a utilizar las nuevas claves y las nuevas expresiones de prefijos de S3.

> **Important**
>
> Una vez que habilita el particionamiento dinámico en un flujo de Firehose, no se puede deshabilitar en este flujo de Firehose.

Comprensión de las claves de particionamiento

Con el particionamiento dinámico, crea conjuntos de datos específicos a partir de los datos de S3 de streaming mediante el particionamiento de los datos basado en claves de particionamiento. Las claves de particionamiento le permiten filtrar los datos de streaming en función de valores específicos. Por ejemplo, si necesita filtrar los datos en función del ID del cliente y el país, puede

especificar el campo de datos `customer_id` como clave de particionamiento y el campo de datos `country` como otra clave de particionamiento. A continuación, especifique las expresiones (con los formatos admitidos) para definir los prefijos de los buckets de S3 en los que se entregarán los registros de datos particionados de forma dinámica.

Puede crear claves de particionamiento con los métodos siguientes.

- Análisis en línea: este método utiliza el mecanismo de soporte integrado de Firehose, un analizador jq, para extraer las claves para el particionamiento de los registros de datos que están en formato JSON. Actualmente, solo admitimos la versión `jq 1.6`.
- Función de AWS Lambda: este método utiliza una función de AWS Lambda específica para extraer y devolver los campos de datos necesarios para el particionamiento.

> **Important**
>
> Al habilitar el particionamiento dinámico, debe configurar al menos uno de estos métodos para particionar los datos. Puede configurar uno de estos métodos para especificar las claves de particionamiento o ambos al mismo tiempo.

Creación de claves de particionamiento con análisis en línea

Para configurar el análisis en línea como método de particionamiento dinámico para sus datos de streaming, debe elegir los parámetros de registro de datos que se utilizarán como claves de particionamiento y proporcionar un valor para cada clave de particionamiento especificada.

El siguiente ejemplo de registro de datos muestra cómo se pueden definir las claves de particionamiento para este mediante el análisis en línea. Tenga en cuenta que los datos deben codificarse en formato Base64. También puede consultar el ejemplo de la CLI.

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

Por ejemplo, puede elegir particionar los datos en función del parámetro `customer_id` o del parámetro `event_timestamp`. Esto significa que desea que el valor del parámetro `customer_id` o del parámetro `event_timestamp` de cada registro se utilice para determinar el prefijo de S3 en el que se entregará el registro. También puede elegir un parámetro anidado, como `device` con una expresión `.type.device`. La lógica de particionamiento dinámico puede depender de varios parámetros.

Tras seleccionar los parámetros de datos para las claves de particionamiento, asigne cada parámetro a una expresión jq válida. En la siguiente tabla se muestra este tipo de asignación de parámetros a expresiones jq:

Parámetro	Expresión 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")

En tiempo de ejecución, Firehose utiliza la columna de la derecha de la tabla anterior para evaluar los parámetros en función de los datos de cada registro.

Creación de claves de particionamiento con una función de AWS Lambda

En el caso de los registros de datos comprimidos o cifrados, o los datos que estén en cualquier formato de archivo que no sea JSON, puede utilizar la función de AWS Lambda integrada con su propio código personalizado para descomprimir, descifrar o transformar los registros con el fin de extraer y devolver los campos de datos necesarios para el particionamiento. Se trata de una expansión de la función de Lambda de transformación existente que está disponible en la actualidad con Firehose. Puede transformar, analizar y devolver los campos de datos que luego puede usar para el particionamiento dinámico con la misma función de Lambda.

A continuación, se muestra un ejemplo de una función de Lambda de procesamiento de flujos de Firehose en Python que reproduce todos los registros leídos de la entrada a la salida y extrae las claves de particionamiento de los 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
```

A continuación, se muestra un ejemplo de una función de Lambda de procesamiento de flujos de Firehose en Go que reproduce todos los registros leídos de la entrada a la salida y extrae las claves de particionamiento de los 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)
}
```

Usar el prefijo del bucket de Amazon S3 para entregar datos

Al crear un flujo de Firehose que utiliza Amazon S3 como destino, debe especificar un bucket de Amazon S3 en el que Firehose entregará los datos. Los prefijos de buckets de Amazon S3 se utilizan para organizar los datos que almacena en los buckets de Amazon S3. Un prefijo de bucket de Amazon S3 es similar a un directorio que permite agrupar objetos similares.

Con el particionamiento dinámico, los datos particionados se entregan en los prefijos de Amazon S3 especificados. Si no habilita el particionamiento dinámico, es opcional especificar un prefijo de bucket de S3 para su flujo de Firehose. Sin embargo, si decide habilitar el particionamiento dinámico, debe especificar los prefijos de bucket de S3 en los que Firehose entregará los datos particionados.

En todos los flujos de Firehose en los que se habilita el particionamiento dinámico, el valor del prefijo de bucket de S3 se compone de expresiones que se basan en las claves de particionamiento especificadas para ese flujo de Firehose. Si vuelve a utilizar el ejemplo de registro de datos anterior, puede crear el siguiente valor de prefijo de S3, que consta de expresiones que se basan en las claves de particionamiento definidas anteriormente:

```
"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}/"
}
```

Firehose evalúa la expresión anterior en tiempo de ejecución. Agrupa los registros que coinciden con la misma expresión de prefijo de S3 evaluada en un único conjunto de datos. A continuación, Firehose entrega cada conjunto de datos en el prefijo de S3 evaluado. La frecuencia de entrega del conjunto de datos en S3 se determina según la configuración del búfer del flujo de Firehose. Como resultado, el registro de este ejemplo se entrega en la siguiente clave de objeto de 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
```

En el caso del particionamiento dinámico, debe usar el siguiente formato de expresión en el prefijo de bucket de S3: `!{namespace:value}`, donde el espacio de nombres puede ser `partitionKeyFromQuery`, `partitionKeyFromLambda` o ambos. Si utiliza el análisis en línea para crear las claves de particionamiento para sus datos de origen, debe especificar un valor de prefijo de bucket de S3 que conste de expresiones especificadas en el siguiente

formato: `"partitionKeyFromQuery:keyID"`. Si utiliza una función de AWS Lambda para crear claves de particionamiento para sus datos de origen, debe especificar un valor de prefijo de bucket de S3 que conste de expresiones especificadas en el siguiente formato: `"partitionKeyFromLambda:keyID"`.

> **Note**
>
> También puede especificar el valor del prefijo de bucket de S3 con el formato de estilo de subárbol, por ejemplo, customer_id=!{partitionKeyFromQuery:customer_id}.

Para obtener más información, consulte “Elección de Amazon S3 como destino” en Creación de un flujo de Amazon Firehose y Prefijos personalizados para los objetos de Amazon S3.

Adición de un delimitador de nueva línea al entregar datos en Amazon S3

Puede habilitar el delimitador de nueva línea para agregar un delimitador de nueva línea entre los registros de los objetos que se entregan en Amazon S3. Esto puede resultar útil para analizar objetos en Amazon S3. Esto también resulta especialmente útil cuando se aplica el particionamiento dinámico a los datos agregados, ya que la desagregación de varios registros (que debe aplicarse a los datos agregados antes de poder particionarlos dinámicamente) elimina nuevas líneas de los registros como parte del proceso de análisis.

Aplicación del particionamiento dinámico a datos agregados

Puede aplicar el particionamiento dinámico a los datos agregados (por ejemplo, varios eventos, registros o registros agregados en una sola llamada a la API `PutRecord` y `PutRecordBatch`), pero primero se deben desagregar estos datos. Para desagregar los datos, puede habilitar la anulación de la agregación de varios registros, es decir, el proceso de analizar los registros del flujo de Firehose y separarlos.

La desagregación de varios registros puede ser de tipo `JSON`, lo que significa que la separación de registros se lleva a cabo en función de objetos JSON consecutivos. La desagregación también puede ser de tipo `Delimited`, lo que significa que la separación de registros se lleva a cabo en función de un delimitador personalizado específico. Este delimitador personalizado debe ser una cadena codificada en base64. Por ejemplo, si desea utilizar la siguiente cadena como delimitador personalizado `####`, debe especificarla en el formato codificado en base64, que se traduce a

`IyMjIw==`. La desagregación de registros por JSON o por delimitador tiene un límite de 500 por registro.

> **Note**
>
> Al desagregar registros JSON, asegúrese de que la entrada siga presentándose en el formato JSON compatible. Los objetos JSON deben estar en una sola línea sin delimitador o estar únicamente delimitados por líneas nuevas (JSONL). Una matriz de objetos JSON no es una entrada válida.
> Estos son ejemplos de entradas correctas: `{"a":1}{"a":2}` and `{"a":1}\n{"a":2}`
> Este es un ejemplo de una entrada incorrecta: `[{"a":1}, {"a":2}]`

Con los datos agregados, al habilitar el particionamiento dinámico, Firehose analiza los registros y busca objetos JSON válidos o registros delimitados en cada llamada a la API en función del tipo de desagregación de varios registros especificado.

> **Important**
>
> Si sus datos están agregados, el particionamiento dinámico solo se puede aplicar si primero se desagregan los datos.

> **Important**
>
> Cuando utilice la característica de transformación de datos en Firehose, la desagregación se aplicará antes de la transformación de datos. Los datos que lleguen a Firehose se procesarán en el siguiente orden: desagregación → transformación de datos mediante Lambda → claves de particionamiento.

Solución de errores de particionamiento dinámico

Si Amazon Data Firehose no puede analizar los registros de datos de su flujo de Firehose o no puede extraer las claves de particionamiento especificadas ni evaluar las expresiones incluidas en el valor del prefijo de S3, estos registros de datos se entregan en el prefijo del bucket de errores de S3 que debe especificar al crear el flujo de entrega en el que se habilita el particionamiento dinámico. El prefijo del bucket de errores de S3 contiene todos los registros que Firehose no puede entregar en

el destino de S3 especificado. Estos registros se organizan en función del tipo de error. Junto con el registro, el objeto entregado también incluye información sobre el error para ayudar a comprenderlo y resolverlo.

Debe especificar un prefijo de bucket de errores de S3 para un flujo de Firehose si desea habilitar el particionamiento dinámico para este flujo de Firehose. Si no desea habilitar el particionamiento dinámico para un flujo de Firehose, es opcional especificar un prefijo de bucket de errores de S3.

Datos de búfer para particionamiento dinámico

Amazon Data Firehose almacena en búfer una cantidad determinada de datos de streaming de entrada durante un periodo determinado antes de entregarlos en los destinos especificados. Puede configurar el tamaño y el intervalo del búfer al crear nuevos flujos de Firehose o actualizar el tamaño y el intervalo del búfer en sus flujos de Firehose existentes. El tamaño de un búfer se mide en MB y el intervalo de un búfer, en segundos.

> **Note**
>
> La característica de almacenamiento en búfer cero no está disponible para el particionamiento dinámico.

Cuando el particionamiento dinámico está habilitado, Firehose almacena internamente en búfer los registros que pertenecen a una partición determinada en función de la sugerencia de almacenamiento en búfer configurada (tamaño y tiempo) antes de enviarlos a su bucket de Amazon S3. Para entregar objetos con el tamaño máximo, Firehose utiliza internamente el almacenamiento en búfer de varias etapas. Por lo tanto, el retraso de principio a fin de un lote de registros puede ser 1,5 veces mayor que el tiempo de sugerencia de almacenamiento en búfer configurado. Esto afecta a la actualización de los datos de un flujo de Firehose.

El recuento de particiones activas es el número total de particiones activas en el búfer de entrega. Por ejemplo, si la consulta de particionamiento dinámico crea 3 particiones por segundo y tiene una configuración de sugerencias de búfer que activa la entrega cada 60 segundos, tendrá un promedio de 180 particiones activas. Si Firehose no puede entregar los datos de una partición en un destino, esta partición se cuenta como activa en el búfer de entrega hasta que se pueda entregar.

Se crea una nueva partición cuando se evalúa un prefijo de S3 para obtener un nuevo valor en función de los campos de datos de registro y las expresiones de prefijos de S3. Se crea un búfer

nuevo para cada partición activa. Todos los registros posteriores con el mismo prefijo de S3 evaluado se envían a ese búfer.

Una vez que el búfer alcanza el límite de tamaño del búfer o el intervalo de tiempo del búfer, Firehose crea un objeto con los datos del búfer y lo entrega en el prefijo de Amazon S3 especificado. Una vez entregado el objeto, el búfer de esa partición y la propia partición se eliminan y se quitan del recuento de particiones activas.

Firehose entrega los datos de cada búfer como un único objeto una vez que se cumple el tamaño o el intervalo del búfer para cada partición por separado. Cuando el número de particiones activas alcanza el límite de 500 por flujo de Firehose, el resto de los registros del flujo de Firehose se entregan en el prefijo del bucket de errores de S3 especificado (activePartitionExceeded). Puede utilizar el formulario de límites de Amazon Data Firehose para solicitar un aumento de esta cuota hasta un máximo de 5000 particiones activas por flujo de Firehose. Si necesita más particiones, puede crear más flujos de Firehose y distribuir las particiones activas entre ellos.

Conversión del formato de los datos de entrada en Amazon Data Firehose

Amazon Data Firehose puede convertir el formato de los datos de entrada de JSON a Apache Parquet o Apache ORC antes de almacenarlos en Amazon S3. Parquet y ORC son formatos de datos en columnas que ahorran espacio y permiten unas búsquedas más rápidas en comparación con los formatos orientados a filas como JSON. Si desea convertir un formato de entrada distinto de JSON, como valores separados por comas (CSV) o texto estructurado, puede utilizar AWS Lambda para transformarlo primero a JSON. Para obtener más información, consulte Transformación de los datos de origen.

Puede convertir el formato de los datos incluso si agrega los registros antes de enviarlos a Amazon Data Firehose.

Amazon Data Firehose requiere los tres elementos siguientes para convertir el formato de los datos de los registros:

Deserializer

Amazon Data Firehose requiere un deserializador para leer el JSON de los datos de entrada. Puede elegir uno de los siguientes dos tipos de deserializador:

Al combinar varios documentos JSON en el mismo registro, asegúrese de que la entrada siga presentándose en el formato JSON compatible. Una matriz de documentos JSON no es una entrada válida.

Por ejemplo, esta es la entrada correcta: `{"a": 1}{"b": 1}`, y esta es la incorrecta: `[{"a":1}, {"a":2}]`.

- SerDe JSON de Apache Hive
- SerDe JSON de OpenX

Elección del deserializador JSON

Elija el SerDe JSON de OpenX si el JSON de entrada contiene marcas temporales en los formatos siguientes:

- aaaa-MM-dd'T'HH:mm:ss[.S]'Z', donde la fracción puede tener hasta 9 dígitos: por ejemplo, `2017-02-07T15:13:01.39256Z`.
- aaaa-[M]M-[d]d HH:mm:ss[.S], donde la fracción puede tener hasta 9 dígitos: por ejemplo, `2017-02-07 15:13:01.14`.
- Segundos epoch: por ejemplo, `1518033528`.
- Milisegundos epoch: por ejemplo, `1518033528123`.
- Segundos epoch con número de punto flotante: por ejemplo, `1518033528.123`.

El SerDe JSON de OpenX puede convertir puntos (`.`) en guiones bajos (_). También puede convertir claves JSON a minúsculas antes de deserializarlas. Para obtener más información sobre las opciones disponibles con este deserializador a través de Amazon Data Firehose, consulte OpenXJsonSerDe.

Si no está seguro de qué deserializador elegir, utilice el SerDe JSON de OpenX, a menos que tenga marcas temporales que este no admita.

Si tiene marcas temporales en un formato distinto de los que se han mostrado anteriormente, utilice el SerDe JSON de Apache Hive. Si elige este deserializador, puede especificar los formatos de marca temporal que va a utilizar. Para ello, siga la sintaxis de los patrones de las cadenas de formato `DateTimeFormat` de Joda-Time. Para obtener más información, consulte Class DateTimeFormat.

También puede utilizar el valor especial `millis` para analizar las marcas temporales en milisegundos con formato de tiempo Unix. Si no especifica un formato, Amazon Data Firehose utiliza `java.sql.Timestamp::valueOf` de forma predeterminada.

El SerDe JSON de Hive no permite lo siguiente:

- Puntos (`.`) en los nombres de las columnas.
- Campos cuyo tipo sea `uniontype`.
- Campos que tienen tipos numéricos en el esquema, pero que son cadenas en el JSON. Por ejemplo, si el esquema es (un entero) y el JSON es `{"a":"123"}`, el SerDe de Hive genera un error.

El SerDe de Hive no convierte JSON anidado en cadenas. Por ejemplo, si se tiene `{"a": {"inner":1}}`, no trata `{"inner":1}` como una cadena.

Esquema

Amazon Data Firehose requiere un esquema para determinar cómo interpretar esos datos. Utilice AWS Glue para crear un esquema en AWS Glue Data Catalog. Amazon Data Firehose hará referencia a ese esquema y lo usará para interpretar los datos de entrada. Puede utilizar el mismo esquema para configurar tanto Amazon Data Firehose como el software de análisis. Para obtener más información, consulte Rellenar el Catálogo de datos de AWS Glue en la Guía para desarrolladores de AWS Glue.

> **Note**
>
> El esquema creado en el Catálogo de datos de AWS Glue debe coincidir con la estructura de los datos de entrada. De lo contrario, los datos convertidos no contendrán atributos que no estén especificados en el esquema. Si utiliza un JSON anidado, utilice un tipo STRUCT en el esquema que refleje la estructura de los datos JSON. Consulte este ejemplo para ver cómo gestionar un JSON anidado con un tipo STRUCT.

> **Important**
>
> En el caso de los tipos de datos que no especifican un límite de tamaño, existe un límite práctico de 32 MB para todos los datos de una sola fila.
> Si especifica la longitud para `CHAR` o `VARCHAR`, Firehose trunca las cadenas a la longitud especificada cuando lee los datos de entrada. Si la cadena de datos subyacente es más larga, permanece sin cambios.

Serializer

Firehose requiere un serializador para convertir los datos al formato de almacenamiento en columnas de destino (Parquet u ORC): puede elegir entre uno de los dos tipos de serializadores.

- SerDe de ORC
- SerDe de Parquet

Elección del serializador

El serializador que elija depende de sus necesidades empresariales. Para obtener más información sobre las dos opciones de serializador, consulte ORC SerDe y Parquet SerDe.

Habilitar la conversión del formato de registros

Si habilita la conversión de formatos de registros, no podrá definir el destino de Amazon Data Firehose en Amazon OpenSearch Service, Amazon Redshift ni Splunk. Una vez habilitada la conversión de formatos, Amazon S3 es el único destino que se puede utilizar para el flujo de Firehose. En la siguiente sección, se muestra cómo habilitar la conversión del formato de registros desde la consola y desde las operaciones de la API de Firehose. Para ver un ejemplo de cómo configurar la conversión de formatos de registros con CloudFormation, consulte AWS::DataFirehose::DeliveryStream.

Habilitar la conversión de formatos de registros desde la consola

Puede habilitar la conversión del formato de datos en la consola al crear o actualizar un flujo de Firehose. Una vez habilitada la conversión de formatos de datos, Amazon S3 es el único destino que se puede configurar para el flujo de Firehose. Además, la compresión de Amazon S3 se deshabilita al habilitar la conversión de formatos. Sin embargo, la compresión Snappy se realiza automáticamente como parte del proceso de conversión. El formato de trama de Snappy que Amazon Data Firehose utiliza en este caso es compatible con Hadoop. Esto significa que puede utilizar los resultados de la compresión de Snappy y ejecutar consultas con estos datos en Athena. Para ver el formato de trama de Snappy que Hadoop utiliza, consulte BlockCompressorStream.java.

Habilitación de la conversión del formato de datos de un flujo de datos de Firehose

1. Inicie sesión en la Consola de administración de AWS y abra la consola de Amazon Data Firehose en https://console.aws.amazon.com/firehose/.
2. Elija un flujo de Firehose que desee actualizar o cree uno nuevo siguiendo los pasos descritos en Tutorial: Crear un flujo de Firehose desde la consola.
3. En Convert record format (Convertir formato de registro), establezca Record format conversion (Conversión del formato de registro) en Enabled (Habilitado).
4. Elija el formato de salida que desea utilizar. Para obtener más información acerca de las dos opciones, consulte Apache Parquet y Apache ORC.

5. Elija una tabla de AWS Glue para especificar un esquema para los registros de origen. Establezca la región, la base de datos, la tabla y la versión de la tabla.

Gestión de la conversión de formatos de registro desde la API de Firehose

Si desea que Amazon Data Firehose convierta el formato de los datos de entrada de JSON a Parquet u ORC, especifique el elemento DataFormatConversionConfiguration opcional en ExtendedS3DestinationConfiguration o en ExtendedS3DestinationUpdate. Si especifica DataFormatConversionConfiguration, se aplican las siguientes restricciones.

- En BufferingHints, no puede establecer `SizeInMBs` en un valor inferior a 64 si habilita la conversión del formato de registros. Además, si la conversión de formato no está habilitada, el valor predeterminado es 5. El valor pasa a ser 128 cuando se habilita.
- Debe establecer `CompressionFormat` del tipo de datos ExtendedS3DestinationConfiguration o ExtendedS3DestinationUpdate en `UNCOMPRESSED`. El valor predeterminado de `CompressionFormat` es `UNCOMPRESSED`. Por lo tanto, también puede dejarlo sin especificar en ExtendedS3DestinationConfiguration. Los datos se siguen comprimiendo como parte del proceso de serialización utilizando la compresión Snappy de forma predeterminada. El formato de trama de Snappy que Amazon Data Firehose utiliza en este caso es compatible con Hadoop. Esto significa que puede utilizar los resultados de la compresión de Snappy y ejecutar consultas con estos datos en Athena. Para ver el formato de trama de Snappy que Hadoop utiliza, consulte BlockCompressorStream.java. Al configurar el serializador, puede elegir otros tipos de compresión.

Gestión de errores en la conversión del formato de datos

Cuando Amazon Data Firehose no puede analizar ni deserializar un registro (por ejemplo, cuando los datos no coinciden con el esquema), los escribe en Amazon S3 con un prefijo de error. Si esta operación de escritura falla, Amazon Data Firehose la reintenta para siempre y bloquea cualquier posible entrega posterior. Para cada registro con errores, Amazon Data Firehose escribe un documento JSON con el siguiente esquema:

```
{
  "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
  }
}
```

Comprensión de la entrega de datos en Amazon Data Firehose

Cuando envía datos al flujo de Firehose, se envían automáticamente al destino que elija. En la siguiente tabla, se explica la entrega de datos a diferentes destinos.

Destino	Detalles
Amazon S3	Para la entrega de datos a Amazon S3, Firehose concatena varios registros entrantes en función de la configuración de almacenam iento en búfer del flujo de Firehose. A continuación, entrega los registros en Amazon S3 como un objeto de Amazon S3. De forma predeterminada, Firehose concatena los datos sin ningún delimitad or. Si desea tener nuevos delimitadores de línea entre los registros, puede añadirlos activando la característica en la configuración de la consola de Firehose o en el parámetro de la API. La entrega de datos entre Firehose y el destino de Amazon S3 se cifra con TLS (HTTPS).
Amazon Redshift	Para entregar datos en Amazon Redshift, Firehose envía primero los datos de entrada al bucket de S3 en el formato descrito anteriorm ente. A continuación, Firehose emite un comando COPY de Amazon Redshift para cargar los datos del bucket de S3 en el clúster aprovisionado de Amazon Redshift o el grupo de trabajo de Amazon Redshift sin servidor. Asegúrese de que, después de que Amazon Data Firehose haya concatenado varios registros de entrada a un objeto de Amazon S3, este objeto se pueda copiar en el clúster aprovisionado de Amazon Redshift o en el grupo de trabajo de Amazon Redshift sin servidor. Para obtener más información, consulte Amazon Redshift COPY Command Data Format Parameter s.
OpenSearch Service y OpenSearch sin servidor	Para la entrega de datos en OpenSearch Service y OpenSearc h sin servidor, Amazon Data Firehose almacena los registros de entrada en el búfer en función de la configuración de almacenam iento en búfer del flujo Firehose. A continuación, genera una solicitud masiva de OpenSearch Service u OpenSearch sin servidor para

Destino	Detalles
	indexar varios registros en su clúster de OpenSearch Service o su colección de OpenSearch sin servidor. Asegúrese de que el registro esté codificado en UTF-8 y aplanado en un objeto JSON de una sola línea antes de enviarlo a Amazon Data Firehose. Además, la opción `rest.action.multi.allow_explicit_index` del clúster de OpenSearch Service se debe establecer en true (valor predeterminado) para poder aceptar solicitudes masivas con un índice explícito que se establece para cada registro. Para obtener más información, consulte OpenSearch Service Configure Advanced Options en la Guía para desarrolladores de Amazon OpenSearch Service.
Splunk	Para la entrega de datos en Splunk, Amazon Data Firehose concatena los bytes que se envían. Si desea delimitadores en los datos, como, por ejemplo, un carácter de nueva línea, debe insertarlos usted mismo. Asegúrese de que Splunk esté configurado para analizar dichos delimitadores. Para volver a enviar a Splunk los datos que se enviaron al bucket de errores de S3 (copia de seguridad de S3), siga los pasos que se mencionan en la documentación de Splunk.
Punto de conexión HTTP	Para entregar datos en un punto de conexión HTTP que pertenece a un proveedor de servicios de terceros admitido, puede usar el servicio Amazon Lambda integrado para crear una función que transforme los registros de entrada en el formato que coincida con el formato que espera la integración del proveedor de servicios. Póngase en contacto con el proveedor de servicios de terceros cuyo punto de conexión HTTP haya elegido como destino para obtener más información sobre el formato de registro aceptado.

Destino	Detalles
Snowflake	Para la entrega de datos a Snowflake, Amazon Data Firehose almacena internamente los datos en búfer durante un segundo y utiliza las operaciones de la API de streaming de Snowflake para insertar datos en Snowflake. De forma predeterminada, los registros que se insertan se vacían y se archivan en la tabla de Snowflake cada segundo. Tras realizar la llamada de inserción, Firehose emite una métrica de CloudWatch que mide el tiempo que tardaron los datos en enviarse a Snowflake. Firehose actualmente solo admite un elemento JSON como carga útil de registro y no admite matrices JSON. Asegúrese de que la carga útil de entrada sea un objeto JSON válido y esté bien formada sin comillas dobles, comillas ni caracteres de escape adicionales.

Cada destino de Firehose tiene su propia frecuencia de entrega de datos. Para obtener más información, consulte Configuración de sugerencias de almacenamiento en búfer.

Registros duplicados

Amazon Data Firehose utiliza una semántica de procesamiento de tipo “al menos una vez” para la entrega de datos. En algunas circunstancias, como cuando se agota el tiempo de espera de la entrega de datos, los reintentos de entrega que Amazon Data Firehose lleva a cabo pueden generar duplicados si la solicitud de entrega de datos finalmente se procesa. Esto se aplica a todos los tipos de destino que admite Amazon Data Firehose, excepto los de destinos de Amazon S3, tablas de Apache Iceberg y destinos de Snowflake.

Temas

- Comprensión de las entregas en todas las cuentas y regiones de AWS
- Comprender las especificaciones de solicitudes y respuestas de entrega de puntos de conexión HTTP
- Gestión de errores en la entrega de datos
- Configuración del formato de nombres de objetos de Amazon S3
- Configuración de la rotación de indexación para OpenSearch Service
- Pausa y reanudación de la entrega de datos

Comprensión de las entregas en todas las cuentas y regiones de AWS

Amazon Data Firehose admite la entrega de datos en destinos de punto de conexión HTTP en las cuentas de AWS. El flujo de Firehose y el punto de conexión HTTP que haya elegido como destino pueden estar en cuentas de AWS diferentes.

Amazon Data Firehose también admite la entrega de datos en destinos de punto de conexión HTTP en las regiones de AWS. Puede entregar datos desde un flujo de Firehose de una región de AWS en un punto de conexión HTTP de otra región de AWS. También puede entregar datos desde un flujo de Firehose en un destino de punto de conexión HTTP fuera de una región de AWS, por ejemplo, en su propio servidor en las instalaciones; para ello, debe establecer la URL del punto de conexión HTTP en el destino deseado. En estos casos, se agregan cargos de transferencia de datos adicionales a los gastos de entrega. Para obtener más información, consulte la sección Transferencia de datos de la página “Precios de las instancias bajo demanda”.

Comprender las especificaciones de solicitudes y respuestas de entrega de puntos de conexión HTTP

Para que Amazon Data Firehose entregue los datos correctamente en los puntos de conexión HTTP personalizados, estos puntos de conexión deben aceptar solicitudes y enviar respuestas con determinados formatos de solicitud y respuesta de Amazon Data Firehose. En esta sección, se describen las especificaciones de formato de las solicitudes HTTP que el servicio Amazon Data Firehose envía a los puntos de conexión HTTP personalizados, así como las especificaciones de formato de las respuestas HTTP que espera el servicio Amazon Data Firehose. Los puntos de conexión HTTP tienen 3 minutos para responder a una solicitud antes de que Amazon Data Firehose agote el tiempo de espera de esa solicitud. Amazon Data Firehose trata las respuestas que no cumplen con el formato adecuado como errores de entrega.

Formato de solicitudes

Parámetros de ruta y URL

Los configura directamente como parte de un único campo de URL. Amazon Data Firehose los envía tal y como están configurados, sin modificarlos. Solo se admiten los destinos HTTPS. Las restricciones de URL se aplican durante la configuración del flujo de entrega.

> **Note**
>
> Actualmente, solo se admite el puerto 443 para la entrega de datos de puntos de conexión HTTP.

Encabezados HTTP: X-Amz-Firehose-Protocol-Version

Este encabezado se usa para indicar la versión de los formatos de solicitudes o respuestas. Actualmente, la única versión es la 1.0.

Encabezados HTTP: X-Amz-Firehose-Request-Id

El valor de este encabezado es un GUID opaco que se puede utilizar con fines de depuración y eliminación de duplicados. Si es posible, las implementaciones de puntos de conexión deben registrar el valor de este encabezado, tanto para las solicitudes que son correctas como para las que no lo son. El ID de solicitud se mantiene igual durante varios intentos de la misma solicitud.

Encabezados HTTP: Content-Type

El valor del encabezado Content-Type es siempre `application/json`.

Encabezados HTTP: Content-Encoding

Se puede configurar un flujo de Firehose para que utilice GZIP a fin de comprimir el cuerpo al enviar solicitudes. Cuando esta compresión está habilitada, el valor del encabezado Content-Encoding se establece en gzip, según la práctica habitual. Si la compresión no está habilitada, el encabezado Content-Encoding no aparece.

Encabezados HTTP: Content-Length

Se usa de forma estándar.

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

El ARN del flujo de Firehose se representa en formato de cadena ASCII. El ARN codifica la región, el ID de la cuenta de AWS y el nombre del flujo. Por ejemplo, `arn:aws:firehose:us-east-1:123456789:deliverystream/testStream`.

Encabezados HTTP: X-Amz-Firehose-Access-Key

Este encabezado contiene una clave de API u otras credenciales. Puede crear o actualizar la clave de API (también conocida como token de autorización) al crear o actualizar el flujo de

entrega. Amazon Data Firehose restringe el tamaño de la clave de acceso a 4096 bytes. Amazon Data Firehose no intenta interpretar esta clave de ninguna manera. La clave configurada se copia palabra por palabra en el valor de este encabezado. Sin embargo, si utiliza Secrets Manager para configurar la clave, el secreto debe seguir un formato de objeto JSON específico: `{"api_key": "..."}`.

El contenido puede ser arbitrario y, potencialmente, representar un token JWT o una ACCESS_KEY. Si un punto de conexión requiere credenciales de varios campos (por ejemplo, nombre de usuario y contraseña), los valores de todos los campos deben almacenarse juntos en una única clave de acceso en un formato que el punto de conexión comprenda (JSON o CSV). Este campo puede codificarse en base64 si el contenido original es binario. Amazon Data Firehose no modifica ni codifica el valor configurado y utiliza el contenido tal cual.

Encabezados HTTP: X-Amz-Firehose-Common-Attributes

Este encabezado contiene los atributos comunes (metadatos) que pertenecen a toda la solicitud o a todos los registros de la solicitud. Los configura directamente al crear un flujo de Firehose. El valor de este atributo está codificado como un objeto JSON con el siguiente 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
```

A continuación se muestra un ejemplo:

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

Cuerpo: tamaño máximo

Configura el tamaño máximo del cuerpo, que puede ser de hasta 64 MiB antes de la compresión.

Cuerpo: esquema

El cuerpo contiene un único documento JSON con el siguiente esquema JSON (escrito en 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
```

A continuación se muestra un ejemplo:

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

Formato de respuesta

Comportamiento predeterminado en caso de error

Si una respuesta no cumple con los requisitos que se indican a continuación, el servidor de Firehose la tratará como si tuviera un código de estado 500 sin cuerpo.

Código de estado

El código de estado HTTP DEBE estar en el rango 2XX, 4XX o 5XX.

El servidor de Amazon Data Firehose NO sigue los redireccionamientos (códigos de estado 3XX). Solo el código de respuesta 200 se considera una entrega correcta de los registros a HTTP/EP. El código de respuesta 413 (tamaño excedido) se considera un error permanente y, si está configurado, el lote de registros no se envía al bucket de errores. Todos los demás códigos de

respuesta se consideran errores recuperables y están sujetos a un algoritmo de reintentos de retroceso que se explica más adelante.

Encabezados: tipo de contenido

El único tipo de contenido aceptable es application/json.

Encabezados HTTP: Content-Encoding

NO SE DEBE utilizar Content-Encoding. El cuerpo DEBE estar descomprimido.

Encabezados HTTP: Content-Length

El encabezado Content-Length DEBE estar presente si la respuesta tiene un cuerpo.

Cuerpo: tamaño máximo

El cuerpo de la respuesta debe tener un tamaño de 1 MiB o menos.

```
"$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
```

A continuación se muestra un ejemplo:

```
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
}
```

Gestión de las respuestas de errores

En todos los casos de error, el servidor de Amazon Data Firehose vuelve a intentar entregar el mismo lote de registros mediante un algoritmo de retroceso exponencial. Los reintentos se retroceden con un tiempo de retroceso inicial (1 segundo) con un factor de fluctuación (15 %) y cada reintento posterior se retrocede con la fórmula (initial-backoff-time * (multiplier(2) ^ retry_count)) con la fluctuación agregada. El tiempo de retroceso está limitado a un intervalo máximo de 2 minutos. Por ejemplo, en el enésimo reintento, el tiempo de retroceso es = MAX(120, 2^n) * random(0.85, 1.15).

Los parámetros especificados en la ecuación anterior están sujetos a cambios. Consulte la documentación de AWS Firehose para conocer el tiempo de retroceso inicial exacto, el tiempo de retroceso máximo y los porcentajes de multiplicador y fluctuación utilizados en el algoritmo de retroceso exponencial.

En cada intento posterior, la clave de acceso o el destino en el que se entregan los registros pueden cambiar en función de la configuración actualizada del flujo de Firehose. El servicio

Amazon Data Firehose utiliza el mismo ID de solicitud en todos los reintentos de la mejor manera posible. El servidor de puntos de conexión HTTP puede utilizar esta última característica con fines de eliminación de duplicados. Si la solicitud sigue sin entregarse después del tiempo máximo permitido (según la configuración del flujo de Firehose), el lote de registros se puede entregar opcionalmente en un bucket de errores según la configuración del flujo.

Ejemplos

Ejemplo de una solicitud originada en CWLog.

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

Gestión de errores en la entrega de datos

Cada destino de Amazon Data Firehose tiene su propia gestión de errores en la entrega de datos.

Cuando configura un flujo de Firehose para muchos destinos (como OpenSearch, Splunk y puntos de conexión HTTP), también configura un bucket de S3 en el que se pueden hacer copias de seguridad de los datos que no se puedan entregar. Para obtener más información sobre cómo Firehose hace copias de seguridad de los datos en caso de errores de entrega, consulte las secciones de destino correspondientes de esta página. Para obtener más información sobre cómo conceder acceso a los buckets de S3 donde se pueden hacer copias de seguridad de los datos que no se pueden entregar, consulte Concesión a Firehose de acceso a un destino de Amazon S3. Cuando Firehose (a) no puede entregar los datos en el destino del flujo y (b) no puede escribir los datos en el bucket de S3 de copias de seguridad en caso de entregas con errores, pausa de forma efectiva la entrega del flujo hasta que los datos puedan entregarse en el destino o escribirse en la ubicación de S3 de copias de seguridad.

Amazon S3

La entrega de datos al bucket de S3 podría generar errores por varias razones. Por ejemplo, es posible que el bucket ya no exista, que el rol de IAM adoptado por Amazon Data Firehose no tenga acceso al bucket, que se haya producido un error de red o cualquier otro evento parecido. En esos casos, Amazon Data Firehose intenta llevar a cabo de nuevo la entrega durante un máximo de 24 horas hasta que se completa. El tiempo máximo de almacenamiento de datos de Amazon Data Firehose es de 24 horas. Si, una vez transcurridas esas 24 horas, no se pueden entregar los datos, estos se pierden.

La entrega de datos al bucket de S3 puede generar errores por varias razones, como por ejemplo:

- El bucket ya no existe.
- El rol de IAM que asume Amazon Data Firehose carece de acceso al bucket.
- Problemas de red.
- Errores de S3, como HTTP 500 u otros errores de API.

En estos casos, Amazon Data Firehose volverá a intentar la entrega:

- Fuentes DirectPut: los reintentos continúan durante un máximo de 24 horas.
- Fuentes de Kinesis Data Streams o Amazon MSK: los reintentos continúan de forma indefinida, hasta cumplir con la política de retención del flujo.

Amazon Data Firehose envía los registros fallidos a un bucket de errores de S3 solo cuando se produce un error en el procesamiento de Lambda o en la conversión de parquet. Otros escenarios de error provocarán reintentos continuos con S3 hasta que se alcance el periodo de retención. Cuando Firehose entrega correctamente los registros a S3, crea un archivo de objetos de S3 y, en caso de errores parciales en los registros, vuelve a intentar la entrega automáticamente y actualiza el mismo archivo de objetos de S3 con los registros procesados correctamente.

Amazon Redshift

Si el destino es Amazon Redshift, puede especificar durante cuánto tiempo reintentar la entrega (entre 0 y 7200 segundos) al crear un flujo de Firehose.

La entrega de datos en su clúster aprovisionado de Amazon Redshift o grupo de trabajo de Amazon Redshift sin servidor puede fallar por varios motivos. Por ejemplo, puede que haya una configuración de clúster incorrecta en el flujo de Firehose, que un clúster o grupo de trabajo esté en mantenimiento o que se haya producido un error de red. En estos casos, Amazon Data Firehose reintenta la entrega durante el tiempo especificado e ignora ese lote concreto de objetos de Amazon S3. La información de los objetos ignorados se entrega al bucket de S3 en forma de archivo de manifiesto, en la carpeta `errors/`, que puede utilizar para reposiciones manuales. Para obtener más información acerca de cómo usar el comando COPY para copiar datos manualmente con archivos de manifiesto, consulte Uso de un manifiesto para especificar archivos de datos.

Amazon OpenSearch Service y OpenSearch sin servidor

Si el destino es OpenSearch Service u OpenSearch sin servidor, puede especificar una duración de reintento (entre 0 y 7200 segundos) al crear el flujo de Firehose.

La entrega de datos en el clúster de OpenSearch Service o la colección de OpenSearch sin servidor puede fallar por varios motivos. Por ejemplo, es posible que haya una configuración de clúster de OpenSearch Service o una colección de OpenSearch sin servidor incorrecta en el flujo de Firehose, que un clúster de OpenSearch Service o una colección de OpenSearch sin servidor esté en mantenimiento, que se haya producido un error de red o cualquier otro evento parecido. En estos casos, Amazon Data Firehose reintenta la entrega durante el tiempo especificado e ignora esa solicitud de indexación concreta. Los documentos ignorados se entregan al bucket de S3 en forma de archivo de manifiesto, en la carpeta `AmazonOpenSearchService_failed/`, que puede utilizar para reposiciones manuales.

En el caso de OpenSearch Service, cada documento tiene el siguiente 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)"
}
```

En el caso de OpenSearch sin servidor, cada documento tiene el siguiente 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

Cuando Amazon Data Firehose envía datos a Splunk, espera recibir una confirmación de Splunk. Si se produce un error o la confirmación no llega dentro del periodo de tiempo de espera de confirmación, Amazon Data Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento. Después de eso, Amazon Data Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Amazon Data Firehose envía datos a Splunk, ya sea en el intento inicial o en un reintento, reinicia el contador de tiempo de espera de confirmación. A continuación, espera a que llegue una confirmación desde Splunk. Aunque se agote el tiempo de reintento, Amazon Data

Firehose sigue esperando la confirmación hasta que la recibe o hasta que finaliza el tiempo de espera de confirmación. Si se agota el tiempo de espera de confirmación, Amazon Data Firehose comprueba para determinar si queda tiempo en el contador de reintento. Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una confirmación o determina que el tiempo de reintento se ha agotado.

Un error de recepción de una confirmación no es el único tipo de error de entrega de datos que puede producirse. Para obtener información sobre los demás tipos de errores de entrega de datos, consulte Errores de entrega de datos relacionados con Splunk. Cualquier error de entrega de datos desencadenará la lógica de reintentos si el tiempo de reintento es mayor que 0.

A continuación se muestra un ejemplo de registro de error.

```
{
  "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 de punto de conexión HTTP

Cuando Amazon Data Firehose envía datos a un destino de punto de conexión HTTP, espera una respuesta de este destino. Si se produce un error o la respuesta no llega dentro del periodo de tiempo de espera de respuesta, Amazon Data Firehose pone en marcha el contador de tiempo de reintento. Continúa intentándolo hasta que se agota el tiempo de reintento. Después de eso, Amazon Data Firehose considera que se trata de un error de entrega de datos y crea una copia de seguridad de los datos en el bucket de Amazon S3.

Cada vez que Amazon Data Firehose envía datos a un destino de punto de conexión HTTP, ya sea en el intento inicial o en un reintento, reinicia el contador de tiempo de espera de respuesta. A continuación, espera a que llegue una respuesta del destino de punto de conexión HTTP. Aunque se agote el tiempo de reintento, Amazon Data Firehose sigue esperando la respuesta hasta que la

recibe o hasta que finaliza el tiempo de espera de respuesta. Si se agota el tiempo de espera de respuesta, Amazon Data Firehose determina si queda tiempo en el contador de reintento. Si queda tiempo, vuelve a intentarlo y repite la lógica hasta que recibe una respuesta o determina que el tiempo de reintento se ha agotado.

Un error de recepción de una respuesta no es el único tipo de error de entrega de datos que puede producirse. Para obtener información sobre los demás tipos de errores de entrega de datos, consulte HTTP Endpoint Data Delivery Errors.

A continuación se muestra un ejemplo de registro de error.

```
{
 "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

Si el destino es Snowflake, puede especificar una duración de reintento opcional (de 0 a 7200 segundos) cuando crea un flujo de Firehose. El valor de predeterminado para la duración de reintento es de 60 segundos.

La entrega de datos a la tabla de Snowflake puede fallar por varios motivos, como una configuración de destino de Snowflake incorrecta, una interrupción de Snowflake, un fallo de red, etc. La política de reintentos no se aplica a los errores no recuperables. Por ejemplo, si Snowflake rechaza su carga útil de JSON porque hay una columna adicional que falta en la tabla, Firehose no intentará volver a entregarla. En su lugar, crea una copia de seguridad de todos los errores de inserción debidos a problemas de carga útil de JSON en el bucket de errores de S3.

Del mismo modo, si se produce un error en la entrega debido a un rol, tabla o base de datos incorrectos, Firehose no lo vuelve a intentar y escribe los datos en el bucket de S3. La duración del reintento solo se aplica a los errores debidos a un problema con el servicio de Snowflake, a fallos

transitorios de la red, etc. En estos casos, Firehose aplica el reintento durante el tiempo especificado antes de entregarlo en S3. Los registros fallidos se entregan en la carpeta snowflake-failed/, que se puede utilizar para rellenarlos manualmente.

El siguiente es un ejemplo de JSON para cada registro que entregue a 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"
}
```

Configuración del formato de nombres de objetos de Amazon S3

Cuando Firehose entrega datos a Amazon S3, el nombre de la clave del objeto S3 sigue el formato<evaluated prefix><suffix>, donde el sufijo tiene el formato <Firehose stream name>-<Firehose stream version>-<year>-<month>-<day>-<hour>-<minute>-<second>-<uuid><file extension> <Firehose stream version>, que comienza por 1 y aumenta en 1 por cada cambio de configuración del flujo de Firehose. Puede cambiar las configuraciones de flujos de Firehose (por ejemplo, el nombre del bucket de S3, las sugerencias de almacenamiento en búfer, la compresión y el cifrado). Puede hacerlo mediante la consola de Firehose o la operación de la API UpdateDestination.

Para <evaluated prefix>, Firehose añade un prefijo de hora predeterminado en el formato YYYY/MM/dd/HH. Este prefijo crea una jerarquía lógica en el bucket, en la que cada barra inclinada (/) crea un nivel jerárquico. Puede modificar esta estructura especificando un prefijo personalizado que incluya expresiones que se evalúan en tiempo de ejecución. Para obtener información acerca de cómo especificar este prefijo, consulte Prefijos personalizados para los objetos de Amazon Simple Storage Service.

De forma predeterminada, la zona horaria utilizada como prefijo y sufijo es UTC, pero puede cambiarla por la zona horaria que prefiera. Por ejemplo, para usar la hora estándar de Japón en lugar de UTC, puede configurar la zona horaria en Asia/Tokio en la Consola de administración de AWS o en configuración de parámetros de la API (CustomTimeZone). La siguiente lista contiene las zonas horarias que Firehose admite para la configuración del prefijo de S3.

Zonas horarias admitidas

La siguiente es una lista de las zonas horarias que Firehose admite para la configuración del prefijo de 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
```

No puede cambiar el campo de sufijo excepto <file extension>. Al habilitar la conversión o compresión de formatos de datos, Firehose añadirá una extensión de archivo en función de la configuración. En la siguiente tabla, se explica la extensión de archivo predeterminada que añade Firehose:

Configuración	Extensión de archivo
Conversión de formato de datos: Parquet	.parquet
Conversión de formato de datos: ORC	.orc
Compresión: Gzip	.gz
Compresión: Zip	.zip

Configuración	Extensión de archivo
Compresión: Snappy	.snappy
Compresión: Hadoop-Snappy	.hsnappy

También puede especificar la extensión de archivo que prefiera en la consola o en la API de Firehose. La extensión del archivo debe empezar con un punto (.) y puede contener los caracteres permitidos: 0-9a-z!-_.*‘(). La extensión del archivo no puede superar los 128 caracteres.

> **Note**
>
> Al especificar una extensión de archivo, esta anulará la extensión de archivo predeterminada que Firehose añada cuando esté habilitada la conversión o compresión de formato de datos.

Comprensión de los prefijos personalizados para los objetos de Amazon S3

Los objetos entregados a Amazon S3 siguen el formato de nombre de <prefijo evaluado><sufijo>. Puede especificar un prefijo personalizado que incluya expresiones que se evalúan en tiempo de ejecución. El prefijo personalizado que especifique anulará el prefijo predeterminado de `yyyy/MM/dd/HH`.

Puede utilizar expresiones de las siguientes formas en el prefijo personalizado: `!{namespace:value}`, donde `namespace` puede ser una de estas opciones, tal como se explica en las secciones siguientes.

- `firehose`
- `timestamp`
- `partitionKeyFromQuery`
- `partitionKeyFromLambda`

Si un prefijo termina por una barra inclinada, aparece como una carpeta en el bucket de Amazon S3. Para obtener más información, consulte Formato de nombre de objetos de Amazon S3 en la Guía para desarrolladores de Amazon Data Firehose.

`timestamp`Espacio de nombres de

Los valores válidos para este espacio de nombres son cadenas DateTimeFormatter de Java válidas. Por ejemplo, en el año 2018, la expresión `!{timestamp:yyyy}` se evalúa como `2018`.

Cuando se evalúan marcas de tiempo, Firehose utiliza la marca de tiempo de llegada aproximada del registro más antiguo incluido en el objeto de Amazon S3 que se va a escribir.

De forma predeterminada, la marca de tiempo está en UTC. Sin embargo, puede especificar la zona horaria que prefiera. Por ejemplo, puede configurar la zona horaria en Asia/Tokio en la Consola de administración de AWS o en la configuración de parámetros de la API (CustomTimeZone) si desea utilizar la hora estándar de Japón en lugar de UTC. Para ver la lista de zonas horarias compatibles, consulte Formato de nombre de objeto de Amazon S3.

Si se utiliza el espacio de nombres `timestamp` más de una vez en la misma expresión de prefijo, cada instancia se evalúa como el mismo instante en el tiempo.

`firehose`Espacio de nombres de

Hay dos valores que se pueden utilizar con este espacio de nombres: `error-output-type` y `random-string`. En la tabla siguiente, se explica cómo hacerlo.

Los valores del espacio de nombres **`firehose`**

Conversión	Descripción	Ejemplo de entrada	Ejemplo de resultado	Notas
`error-output-type`	Se evalúa como una de las siguientes cadenas, en función de la configuración del flujo de Firehose y el motivo del error: {processing-failed, AmazonOpenSearchService-failed, splunk-fa	`myPrefix/result=!{firehose:error-output-type}/!{timestamp:yyyy/MM/dd}`	`myPrefix/result=processing-failed/2018/08/03`	El valor error-output-type solo se puede utilizar en el campo ErrorOutputPrefix.

Conversión	Descripción	Ejemplo de entrada	Ejemplo de resultado	Notas
	iled, format-conversion-failed, http-endpoint-failed}. Si lo utiliza más de una vez en la misma expresión, cada instancia se evalúa como la misma cadena de error.			
`random-string`	Se evalúa como una cadena aleatoria de 11 caracteres. Si lo utiliza más de una vez en la misma expresión, cada instancia se evalúa como una cadena aleatoria nueva.	`myPrefix/!{firehose:random-string}/`	`myPrefix/046b6c7f-0b/`	Puede utilizarlo con ambos tipos de prefijos. Puede colocarlo al principio de la cadena de formato para obtener un prefijo aleatorio, que a veces es necesario para alcanzar un rendimiento extremadamente alto con Amazon S3.

Espacios de nombres **partitionKeyFromLambda** y **partitionKeyFromQuery**

En el caso del particionamiento dinámico, debe usar el siguiente formato de expresión en el prefijo de bucket de S3: `!{namespace:value}`, donde el espacio de nombres puede ser `partitionKeyFromQuery`, `partitionKeyFromLambda` o ambos. Si utiliza el análisis en línea para crear las claves de particionamiento para sus datos de origen, debe especificar un valor de prefijo de bucket de S3 que conste de expresiones especificadas en el siguiente formato: `"partitionKeyFromQuery:keyID"`. Si utiliza una función de AWS Lambda para crear claves de particionamiento para sus datos de origen, debe especificar un valor de prefijo de bucket de S3 que conste de expresiones especificadas en el siguiente formato: `"partitionKeyFromLambda:keyID"`. Para obtener más información, consulte “Elección de Amazon S3 como destino” en Creación de un flujo de Firehose.

Reglas semánticas

Las siguientes reglas se aplican a las expresiones `Prefix` y `ErrorOutputPrefix`.

- En el espacio de nombres `timestamp`, se evalúa cualquier carácter que no esté entre comillas simples. En otras palabras, cualquier cadena encerrada entre comillas simples en el campo de valor se interpreta literalmente.
- Si se especifica un prefijo que no contiene una expresión de espacio de nombres de marca de tiempo, Firehose agrega la expresión `!{timestamp:yyyy/MM/dd/HH/}` al valor del campo `Prefix`.
- La secuencia `!{` solo puede aparecer en expresiones `!{namespace:value}`.
- `ErrorOutputPrefix` únicamente puede ser null si `Prefix` no contiene ninguna expresión. En este caso, `Prefix` evalúa a `<specified-prefix>yyyy/MM/DDD/HH/` y `ErrorOutputPrefix` evalúa a `<specified-prefix><error-output-type>yyyy/MM/DDD/HH/`. `DDD` representa el día del año.
- Si especifica una expresión para `ErrorOutputPrefix`, debe incluir al menos una instancia de `!{firehose:error-output-type}`.
- `Prefix` no puede contener `!{firehose:error-output-type}`.
- Una vez evaluados, ni `Prefix` ni `ErrorOutputPrefix` pueden tener una longitud superior a 512 caracteres.
- Si el destino es Amazon Redshift, `Prefix` no debe contener expresiones y `ErrorOutputPrefix` debe ser null.

- Si el destino es Amazon OpenSearch Service o Splunk y no se especifica `ErrorOutputPrefix`, Firehose utiliza el campo `Prefix` para los registros con errores.
- Cuando el destino es Amazon S3, `Prefix` y `ErrorOutputPrefix` en la configuración de destino de Amazon S3 se utilizan para los registros correctos y los registros con errores, respectivamente. Si utiliza AWS CLI o la API, puede utilizar `ExtendedS3DestinationConfiguration` para especificar una configuración de copia de seguridad de Amazon S3 con sus propios valores de `Prefix` y `ErrorOutputPrefix`.
- Cuando utiliza la Consola de administración de AWS y establece el destino en Amazon S3, Firehose utiliza `Prefix` y `ErrorOutputPrefix` en la configuración de destino para los registros correctos y los registros con errores, respectivamente. Si especifica un prefijo mediante expresiones, debe especificar el prefijo de error que incluye `!{firehose:error-output-type}`.
- Cuando utiliza `ExtendedS3DestinationConfiguration` con AWS CLI, la API o CloudFormation, si especifica un valor de `S3BackupConfiguration`, Firehose no proporciona un valor predeterminado de `ErrorOutputPrefix`.
- No puede usar los espacios de nombres `partitionKeyFromLambda` y `partitionKeyFromQuery` al crear expresiones ErrorOutputPrefix.

Ejemplos de prefijos

Ejemplos de **`Prefix`** y **`ErrorOutputPrefix`**

Input	Prefijo evaluado (a las 10:30 h UTC del 27 de agosto de 2018)

`Prefix`: sin especificar
 	
`ErrorOutputPrefix` : `myFirehoseFailures/!{firehose:error-output-type}/`
	`Prefix`: `2018/08/27/10`
	
	`ErrorOutputPrefix` : `myFirehoseFailures/processing-failed/`

`Prefix`: `!{timestamp:yyyy/MM/dd}` `ErrorOutputPrefix` : sin especificar	Entrada no válida: `ErrorOutputPrefix` no puede ser null si Prefix contiene expresiones

Input	Prefijo evaluado (a las 10:30 h UTC del 27 de agosto de 2018)

`Prefix`: `myFirehose/DeliveredYear=!{timestamp:yyyy}/anyMonth/rand=!{firehose:random-string}`
 	
`ErrorOutputPrefix` : `myFirehoseFailures/!{firehose:error-output-type}/!{timestamp:yyyy}/anyMonth/!{timestamp:dd}`
	`Prefix`: `myFirehose/DeliveredYear=2018/anyMonth/rand=5abf82daaa5`
	
	`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` : sin especificar
	`Prefix`: `myFirehosePrefix/2018/08/27/`
	
	`ErrorOutputPrefix` : `myFirehosePrefix/processing-failed/2018/08/27/`

Configuración de la rotación de indexación para OpenSearch Service

Si el destino es OpenSearch Service, puede especificar una opción de rotación de índice basada en tiempo. Las cinco opciones disponibles son: NoRotation, OneHour, OneDay, OneWeek o OneMonth.

En función de la opción de rotación seleccionada, Amazon Data Firehose agregará una parte de la marca de tiempo de llegada en UTC al nombre de índice especificado. También rota la marca de tiempo añadida en consecuencia. En el siguiente ejemplo se muestra el nombre de índice resultante de cada opción de rotación de índice en OpenSearch Service, donde el nombre de índice especificado es myindex y la marca de tiempo de llegada es `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**
>
> Con la opción `OneWeek`, Data Firehose crea índices automáticamente con el formato <AÑO>-w<NÚMERO DE LA SEMANA> (por ejemplo, `2020-w33`), donde el número de la semana se calcula según la hora UTC y las siguientes convenciones de EE. UU.:
>
> - Una semana comienza el domingo
> - La primera semana del año es la primera semana que contiene un sábado de este año

Pausa y reanudación de la entrega de datos

Tras configurar un flujo de Firehose, los datos disponibles en el origen del flujo se entregan de forma continua en el destino. Si se produce alguna situación en la que el destino del flujo no esté disponible temporalmente (por ejemplo, durante las operaciones de mantenimiento planificadas), puede que desee pausar temporalmente la entrega de datos y reanudarla cuando el destino vuelva a estar disponible.

> **Important**
>
> Si utiliza el enfoque que se describe a continuación para pausar y reanudar un flujo, después de reanudarlo, verá que se envían pocos registros al bucket de errores de Amazon S3, mientras que el resto del flujo continúa enviándose al destino. Esta es una limitación conocida de este enfoque y se debe a que se registra como fallido un número reducido de registros que no se podían entregar previamente al destino tras varios reintentos.

Pausa de un flujo de Firehose

Para pausar la entrega de un flujo en Firehose, primero elimine los permisos para que Firehose escriba en la ubicación de copias de seguridad de S3 en caso de entregas con errores. Por ejemplo, si quiere pausar el flujo de Firehose con un destino de OpenSearch, puede actualizar los permisos. Para obtener más información, consulte Concesión a Firehose acceso a un destino público de OpenSearch Service.

Elimine el permiso `"Effect": "Allow"` de la acción `s3:PutObject` y agregue de forma explícita una declaración que aplique el permiso `Effect": "Deny"` en la acción `s3:PutObject` para el bucket de S3 que se utiliza para hacer copias de seguridad de las entregas con errores. A continuación, desactive el destino del flujo (por ejemplo, desactive el dominio de OpenSearch de destino) o elimine los permisos para que Firehose escriba en el destino. Para actualizar los permisos de otros destinos, consulte la sección correspondiente al destino en Control del acceso con Amazon Data Firehose. Tras completar estas dos acciones, Firehose dejará de entregar flujos y podrá supervisar esto mediante las métricas de CloudWatch para Firehose.

> **Important**
>
> Al pausar la entrega del flujo en Firehose, debe asegurarse de que el origen del flujo (por ejemplo, Kinesis Data Streams o Managed Service para Kafka) esté configurado para retener los datos hasta que se reanude la entrega del flujo y los datos se entreguen en el destino. Si el origen es DirectPUT, Firehose retendrá los datos durante 24 horas. Se pueden producir pérdidas de datos si no se reanuda el flujo y no se entregan los datos antes de que venza el periodo de retención de datos.

Reanudación de un flujo de Firehose

Para reanudar la entrega, primero revierta el cambio llevado a cabo anteriormente en el destino del flujo; para ello, active el destino y asegúrese de que Firehose tenga permisos para entregar el flujo en el destino. A continuación, revierta los cambios llevados a cabo anteriormente en los permisos aplicados al bucket de S3 para hacer copias de seguridad de las entregas con errores. Es decir, aplique el permiso `"Effect": "Allow"` a la acción `s3:PutObject` y elimine el permiso `"Effect": "Deny"` de la acción `s3:PutObject` para el bucket de S3 que se utiliza para hacer copias de seguridad de las entregas con errores. Por último, supervise con las métricas de CloudWatch para Firehose para confirmar que el flujo se entrega en el destino. Para ver y solucionar los errores, consulte Supervisión de Firehose con Registros de Amazon CloudWatch.

Entrega de datos a tablas de Apache Iceberg con Amazon Data Firehose

Apache Iceberg es un formato de tabla de código abierto de alto rendimiento para realizar análisis de macrodatos. Apache Iceberg aporta la fiabilidad y la simplicidad de las tablas de SQL a los lagos de datos de Amazon S3 y hace posible que motores de análisis de código abierto, como Spark, Flink, Trino, Hive e Impala trabajen simultáneamente con los mismos datos. Para obtener más información, consulte Apache Iceberg y Consideraciones y limitaciones.

Puede usar Firehose para entregar datos de flujo a las tablas de Apache Iceberg en Amazon S3. Las tablas de Apache Iceberg pueden autogestionarse en Amazon S3 o alojarse en las tablas de Amazon S3. En las tablas de Iceberg autogestionadas, usted administra todas las optimizaciones de las tablas, como la compactación y caducidad de las instantáneas. Las tablas de Amazon S3 brindan almacenamiento optimizado para cargas de trabajo de análisis a gran escala, con características que mejoran continuamente el rendimiento de las consultas y reducen los costos de almacenamiento de los datos tabulares. Para obtener más información sobre las tablas de Amazon S3, consulte las tablas de Amazon S3.

Esta característica permite enrutar los registros de un único flujo a diferentes tablas de Apache Iceberg. Puede aplicar automáticamente operaciones de inserción, actualización y eliminación a los registros de esas tablas. También admite un control de acceso a los datos detallado en las tablas de Apache Iceberg en Amazon S3 con. AWS Lake Formation Puede especificar los controles de acceso de forma centralizada AWS Lake Formation y proporcionar permisos más detallados a nivel de tabla y columna para Firehose.

Consideraciones y limitaciones

> **Note**
>
> Firehose admite las tablas Apache Iceberg como destino en todas las regiones, excepto en Regiones de AWSChina AWS GovCloud (US) Regions, y en Asia Pacífico (Malasia).

La compatibilidad de Firehose con las tablas de Apache Iceberg tiene las siguientes consideraciones y limitaciones.

- Rendimiento: si utiliza Direct PUT como fuente para entregar datos a las tablas de Apache Iceberg, el rendimiento máximo por transmisión es de 5 MiB/second en las regiones EE.UU. Este (Norte de Virginia), EE.UU. Oeste (Oregón) y Europa (Irlanda) y 1 MiB/second en todas las demás. Regiones de AWS Si desea insertar datos en las tablas de Iceberg sin actualizaciones ni eliminaciones, y desea aumentar el rendimiento del flujo, puede utilizar el Formulario de límites de Firehose para solicitar un aumento del límite de rendimiento.

  También puede establecer el indicador `AppendOnly` en `True` para insertar datos únicamente y no realizar actualizaciones ni eliminaciones. Si establece el indicador `AppendOnly` en `True`, Firehose se amplía automáticamente para adaptarse al rendimiento. Actualmente, este indicador solo se puede configurar con la operación de la CreateDeliveryStreamAPI.

  Si un flujo Direct PUT sufre una limitación debido a volúmenes de ingesta de datos más altos que superan la capacidad de rendimiento de un flujo de Firehose, este aumenta automáticamente el límite de rendimiento del flujo hasta que se contenga la limitación. Según el aumento del rendimiento y la limitación, Firehose podrá tardar más en aumentar el rendimiento de un flujo hasta los niveles deseados. Por este motivo, continúe reintentando los registros de ingesta de datos fallidos. Si espera que el volumen de datos aumente en cantidades grandes y repentinas, o si su nuevo flujo necesita un rendimiento superior al límite del predeterminado, solicite aumentar el límite de rendimiento.
- Transacciones por segundo (TPS) de S3: si desea optimizar el rendimiento de S3 con Kinesis Data Streams o Amazon MSK como origen, recomendamos segmentar el registro de origen con una clave de partición adecuada. De este modo, los registros de datos que se enrutan a la misma tabla de Iceberg se asignan a uno o pocos segmentos de origen conocidos como particiones. Si es posible, divida los registros de datos que pertenezcan a diferentes tablas de Iceberg de destino en diferentes partitions/shards, so that you can use all the aggregate throughput available across all the partitions/shards of the source topic/stream tablas.
- Columnas: para los nombres y valores de las columnas, Firehose solo toma el primer nivel de nodos de un JSON anidado de varios niveles. Por ejemplo, Firehose selecciona los nodos que están disponibles en el primer nivel, incluido el campo de posición. Los nombres de las columnas y los tipos de los datos de origen deben coincidir con los de las tablas de destino para que Firehose los entregue correctamente. En este caso, Firehose espera que, en las tablas de Iceberg, haya una columna de tipo de datos de estructura o de mapa que coincida con el campo de posición. Firehose admite 16 niveles de anidación. A continuación, se muestra un ejemplo de un JSON anidado.

```
{
```

```
   "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
   }
}
```

Si los nombres de las columnas o los tipos de datos no coinciden, Firehose genera un error y envía los datos al bucket de errores de S3. Si todos los nombres de las columnas y los tipos de datos coinciden en las tablas de Apache Iceberg, pero hay un campo adicional en el registro de origen, Firehose omite el nuevo campo.

- Un objeto JSON por registro: solo puede enviar un objeto JSON en un registro de Firehose. Si agrega y envía varios objetos JSON dentro de un registro, Firehose genera un error y envía los datos al bucket de errores de S3. Si agrega registros con KPL e ingiere datos en Firehose con Amazon Kinesis Data Streams como origen, Firehose se desagrega automáticamente y usa un objeto JSON por registro.
- Optimización de la compactación y el almacenamiento: cada vez que escribe a las tablas Iceberg con Firehose, este archiva y genera instantáneas, archivos de datos y elimina archivos. Tener varios archivos de datos aumenta la sobrecarga de metadatos y afecta al rendimiento de lectura. Para obtener un rendimiento eficaz de las consultas, puede considerar una solución que recoja periódicamente archivos de datos pequeños y los reescriba en un menor número de archivos de datos más grandes. Este proceso se denomina compactación. AWS Glue Data Catalog admite la compactación automática de las tablas Apache Iceberg. Para obtener más información, consulte Administración de compactación en la Guía del usuario de AWS Glue. Para obtener más información, consulte Compactación automática de tablas de Apache Iceberg. Por otro lado, puede ejecutar el comando de Athena Optimize para realizar la compactación de forma manual. Para obtener más información sobre el comando Optimize, consulte Athena Optimize.

  Además de compactar los archivos de datos, también puede optimizar el consumo de almacenamiento con la declaración VACUUM, que realiza el mantenimiento de las tablas de Apache Iceberg, como la caducidad de las instantáneas y la eliminación de un archivo huérfano. Como alternativa, puede utilizar esta herramienta AWS Glue Data Catalog que también permite optimizar las tablas de Apache Iceberg mediante la eliminación automática de los archivos de

datos, los archivos huérfanos y las instantáneas caducadas que ya no sean necesarias. Para obtener más información, consulte esta entrada del blog sobre Optimización del almacenamiento de las tablas de Apache Iceberg.

- No se permite la compatibilidad con el origen Amazon MSK Serverless para las tablas de Apache Iceberg como destino.
- En una operación de actualización, Firehose genera un archivo de eliminación seguido de una operación de inserción. La generación de archivos de eliminación conlleva cargos por parte de Amazon S3.
- Firehose no recomienda utilizar varios flujos de Firehose para escribir datos en la misma tabla de Apache Iceberg. Esto se debe a que Apache Iceberg se basa en el control de concurrencia optimista (OCC). Si varios flujos de Firehose intentan escribir simultáneamente en una sola tabla de Iceberg, solo un flujo conseguirá confirmar los datos en un momento dado. Los otros flujos que no se confirmen se deshabilitan y vuelven a intentar la operación de confirmación hasta que el tiempo de reintento configurado caduce. Una vez agotado el tiempo de reintento, los datos y las claves del archivo de eliminación (rutas de Amazon S3) se envían al prefijo de error de Amazon S3 configurado.
- La versión compatible de la biblioteca de Iceberg con Firehose es la 1.5.2.
- Para entregar datos cifrados a Amazon S3 Tables, debe configurar AWS Key Management Service los parámetros en Amazon S3 Tables y no en la configuración de Firehose. Si configura AWS Key Management Service parámetros en Firehose para entregar datos cifrados a Amazon S3 Tables, Firehose no podrá usar esos parámetros para cifrar. Para obtener más información, consulte Uso del cifrado del lado del servidor con claves. AWS KMS
- Firehose Streams solo admite la entrega a bases de datos y tablas que se crean a través de la API de Iceberg. GlueCatalog No se permite la entrega a bases de datos y tablas que se crean mediante Glue SDK. Recuerde que un guión (-) no es un carácter compatible con la base de datos y el nombre de la tabla en la biblioteca de Iceberg. Para obtener más información, consulte las expresiones regulares de la base de datos de Glue y expresiones regulares de la tabla de Glue que son compatibles con la biblioteca de Iceberg.
- Todos los archivos escritos por Firehose se calculan con los segmentos que se encuentran en el registro. Esto también es válido para los archivos eliminados. No se permiten las eliminaciones globales, como la escritura de archivos de eliminación sin segmentos para una tabla segmentada.

Requisitos previos para utilizar las tablas de Apache Iceberg como destino

Elija entre las siguientes opciones para completar los requisitos previos necesarios.

Temas

Requisitos previos para la entrega de tablas en Amazon S3

Antes de comenzar, complete los siguientes requisitos previos.

- Crear un bucket de Amazon S3: debe crear un bucket de Amazon S3 para añadir la ruta del archivo de metadatos durante la creación de las tablas. Para obtener más información, consulte Creación de un bucket de S3.
- Crear un rol de IAM con los permisos necesarios: Firehose necesita un rol de IAM con permisos específicos para acceder a las tablas de AWS Glue y escribir datos en Amazon S3. La misma función se utiliza para conceder AWS Glue acceso a los buckets de Amazon S3. Necesita este rol de IAM al crear una tabla de Iceberg y flujos de Firehose. Para obtener más información, consulte Conceder a Firehose acceso a las tablas de Amazon S3.
- Crear tablas de Apache Iceberg: si está configurando claves únicas en el flujo de Firehose para actualizaciones y eliminaciones, Firehose valida si la tabla y las claves únicas existen como parte de la creación del flujo. Para esta situación, debe crear tablas antes de crear el flujo de Firehose. Puede utilizarla AWS Glue para crear tablas Iceberg de Apache. Para obtener más información, consulte Creación de tablas de Apache Iceberg. Si no configurará claves únicas en el flujo de Firehose, no necesitará crear tablas de Iceberg antes de crear un flujo de Firehose.

> **Note**
>
> Firehose admite la siguiente versión y formato de tabla para las tablas de Apache Iceberg.
>
> - Versión de formato de tabla: Firehose solo admite el formato de tabla V2. No cree tablas en formato V1; de lo contrario, se producirá un error y los datos se enviarán al bucket de errores de S3.
> - Formato de almacenamiento de datos: Firehose escribe los datos en las tablas de Apache Iceberg en formato Parquet.

- Funcionamiento a nivel de fila: Firehose admite el modo Merge-on-Read (MOR) de escribir datos en las tablas Iceberg de Apache.

Requisitos previos para la entrega de tablas en Amazon S3

Cumpla los siguientes requisitos previos para enviar datos a los buckets de tablas de Amazon S3.

- Cree un bucket de tablas de S3, un espacio de nombres, tablas en el bucket de tablas y otros pasos de integración descritos en la Introducción a las tablas de Amazon S3. Los nombres de las columnas deben estar en minúscula debido a las limitaciones que impone la integración del catálogo de tablas de S3, tal como se especifica en las limitaciones de integración del catálogo de tablas de S3.
- Cree un rol de IAM con los permisos necesarios: Firehose necesita un rol de IAM con permisos específicos para AWS Glue acceder a las tablas y escribir datos en las tablas de un bucket de tablas de Amazon S3. Para escribir en las tablas de un bucket de tablas de S3, también debe proporcionar el rol de IAM con los permisos necesarios en AWS Lake Formation. Debe configurar este rol de IAM al crear un flujo de Firehose. Para obtener más información, consulte la sección Concesión de acceso de Firehose a las tablas de Amazon S3.
- Configura AWS Lake Formation los permisos: AWS Lake Formation administra el acceso a los recursos de tu tabla. Lake Formation utiliza su propio modelo de permisos que permite un control de acceso detallado a los recursos del catálogo de datos.

Para obtener información sobre la step-by-step integración, consulte el blog Cree un lago de datos para transmitir datos con Amazon S3 Tables y Amazon Data Firehose. Para obtener información adicional, consulte también Uso de tablas de Amazon S3 con servicios de AWS análisis.

Configuración del flujo de Firehose

Para crear un flujo de Firehose con las tablas de Apache Iceberg como destino, debe configurar lo siguiente.

> **Note**
>
> La configuración de un flujo de Firehose para la entrega a las tablas de los buckets de tablas de S3 es la misma que la de las tablas de Apache Iceberg de Amazon S3.

Configuración del origen y el destino

Para enviar datos a las tablas de Apache Iceberg, elija el origen del flujo.

Para configurar el origen del flujo, consulte Configurar los ajustes del origen.

A continuación, elija Tablas de Apache Iceberg como destino y proporcione un nombre de flujo de Firehose.

Configuración de transformación de datos

Para realizar transformaciones personalizadas en los datos, como añadir o modificar registros en el flujo entrante, puede añadir una función de Lambda al flujo de Firehose. Para obtener más información sobre la transformación de datos mediante Lambda en un flujo de Firehose, consulte Transformación de los datos de origen en Amazon Data Firehose.

En el caso de las tablas de Apache Iceberg, debe especificar cómo desea enrutar los registros entrantes a las distintas tablas de destino y las operaciones que desea realizar. Una de las formas de proporcionar la información de enrutamiento necesaria a Firehose es mediante una función de Lambda.

Para obtener más información, consulte Enrutar los registros a diferentes tablas de Iceberg.

Conectar catálogo de datos

Apache Iceberg requiere un catálogo de datos para escribir en las tablas de Apache Iceberg. Firehose se integra con AWS Glue Data Catalog las tablas Apache Iceberg.

Puedes usarlo AWS Glue Data Catalog en la misma cuenta que tu transmisión de Firehose o en una cuenta cruzada y en la misma región que tu transmisión de Firehose (predeterminado), o en una región diferente.

Si realiza entregas a una tabla de Amazon S3 y utiliza la consola para configurar el flujo de Firehose, seleccione el catálogo que corresponda a su catálogo de la tabla de Amazon S3. Si está utilizando la CLI para configurar su flujo de Firehose, entonces en la entrada `CatalogConfiguration`, utilice `CatalogARN` con el formato: `arn:aws:glue:<region>:<account-id>:catalog/s3tablescatalog/<s3 table bucket name>`. Para obtener más información, consulte Configuración de un flujo de Firehose en las tablas de Amazon S3.

> **Note**
>
> Firehose admite tres operaciones para las tablas de Iceberg: insertar, actualizar y eliminar. Si no se especifica una operación, Firehose utilizará por defecto la operación insertar, donde agregará cada registro entrante como una nueva fila y conservará los duplicados. Para modificar los registros existentes, especifique la operación de "actualización", que utiliza las claves principales para localizar y cambiar las filas existentes.
> Ejemplo:
>
> - Por defecto (insertar): varios registros de clientes idénticos crean filas duplicadas.
> - Actualización especificada: la nueva dirección del cliente actualiza el registro existente.

Configuración de expresiones JQ

En el caso de las tablas de Apache Iceberg, debe especificar cómo desea enrutar los registros entrantes a las diferentes tablas de destino y las operaciones que desea realizar, como insertar, actualizar y eliminar. Puede hacerlo configurando expresiones JQ para que Firehose las analice y obtenga la información requerida. Para obtener más información, consulte ???.

Configure claves únicas

Actualizaciones y eliminaciones con más de una tabla: las claves únicas son uno o más campos del registro de origen que identifican de forma exclusiva una fila en las tablas de Apache Iceberg. Si tiene un escenario de inserción único con más de una tabla, no tiene que configurar claves únicas. Si desea realizar actualizaciones y eliminaciones en determinadas tablas, debe configurar claves únicas para las tablas requeridas. Tenga en cuenta que la actualización insertará automáticamente la fila si falta en las tablas. Si solo tiene una tabla, puede configurar claves únicas. En una operación de actualización, Firehose genera un archivo de eliminación seguido de una inserción.

Puede configurar claves únicas por tabla como parte de la creación de transmisiones de Firehose o puede configurarlas de identifier-field-idsforma nativa en Iceberg durante la operación de creación o modificación de la tabla. La configuración de claves únicas por tabla durante la creación del flujo es opcional. Si no configura claves únicas por tabla durante la creación del flujo, Firehose comprueba mediante `identifier-field-ids` si hay tablas obligatorias y las usará como claves únicas. Si ninguna está configurada, se produce un error en la entrega de los datos con las operaciones de actualización y eliminación.

Para configurar esta sección, proporcione el nombre de la base de datos, el nombre de la tabla y las claves únicas de las tablas en las que desee actualizar o eliminar datos. Solo puede tener una entrada para cada tabla de la configuración. No es necesario configurar esta sección para los escenarios de solo anexado. Si lo desea, también puede optar por proporcionar un prefijo de bucket de errores si los datos de la tabla no se entregan, como se muestra en el siguiente ejemplo.

```
[
  {
    "DestinationDatabaseName": "MySampleDatabase",
    "DestinationTableName": "MySampleTable",
    "UniqueKeys": [
      "COLUMN_PLACEHOLDER"
    ],
    "S3ErrorOutputPrefix": "OPTIONAL_PREFIX_PLACEHOLDER"
  }
]
```

Firehose permite configurar claves únicas si el nombre de columna especificado es único en toda la tabla. Sin embargo, no permite los nombres de columnas totalmente cualificados como claves únicas. Por ejemplo, una clave denominada `top._id` no se considera una clave única si el nombre de la columna `_id` también se encuentra en el nivel superior. Si `_id` es exclusivo en toda la tabla, se utiliza independiente de su ubicación dentro de la estructura de la tabla. Esto determina si es una columna de nivel superior o una columna anidada. En el siguiente ejemplo, `_id` es una clave única válida para el esquema, ya que el nombre de la columna es exclusivo en todo el esquema.

```
[
 "schema": {
  "type": "struct",
  "fields": [
    {
      "name": "top",
      "type": {
        "type": "struct",
        "fields": [
          { "name": "_id", "type": "string" },
          { "name": "name", "type": "string" }
        ]
      }
    },
    { "name": "user", "type": "string" }
  ]
```

```
}
]
```

En el siguiente ejemplo, `_id` no es una clave única válida para el esquema porque se utiliza en la columna de nivel superior y en la estructura anidada.

```
[
"schema": {
  "type": "struct",
  "fields": [
    {
      "name": "top",
      "type": {
        "type": "struct",
        "fields": [
          { "name": "_id", "type": "string" },
          { "name": "name", "type": "string" }
        ]
      }
    },
    { "name": "_id", "type": "string" }
  ]
}
]
```

Especificación de duración de reintento

Puede usar esta configuración para especificar el tiempo en segundos durante el que Firehose debe volver a intentarlo si encuentra errores al escribir en las tablas de Apache Iceberg en Amazon S3. Puede establecer cualquier valor entre 0 y 7200 segundos para realizar los reintentos. De forma predeterminada, Firehose vuelve a intentarlo durante 300 segundos.

Gestión de la entrega o el procesamiento fallidos

Debe configurar Firehose para que entregue los registros a un bucket de copias de seguridad de S3 en caso de que se produzcan errores en el procesamiento o la entrega de un flujo una vez transcurrido el periodo de reintento. Para ello, configure el bucket de copias de seguridad de S3 y el prefijo de salida de error del bucket de copias de seguridad de S3 desde la Configuración de copias de seguridad de la consola.

Gestionar errores

Firehose envía todos los errores de entrega a CloudWatch Logs y a los depósitos de errores de Amazon S3.

Lista de errores:

Mensaje de error	Descripción
`Iceberg.NoSuchTable`	Firehose le escribe a una tabla que no existe, o la tabla no está en formato V2. Firehose no admite tablas en formato V1.
`Iceberg.InvalidTableName`	Se pasó un nombre vacío o nulo en la tabla, o la tabla no está en formato V2. Firehose no admite tablas en formato V1.
`S3.AccessDenied`	Asegúrese de que el rol de IAM creado en el paso de requisitos previos tenga los permisos y la política de confianza necesarios.
`Glue.AccessDenied`	Asegúrese de que el rol de IAM creado en el paso de requisitos previos tenga los permisos y la política de confianza necesarios.

Configuración de sugerencias de búfer

Firehose almacena en búfer una cantidad determinada de datos de streaming de entrada (Tamaño del almacenamiento en búfer) y durante un periodo determinado (Intervalo de almacenamiento en búfer) antes de entregarlos en las tablas de Apache Iceberg. Puede elegir un tamaño de búfer de 1 a 128 segundos MiBs y un intervalo de búfer de 0 a 900 segundos. Un mayor número de sugerencias de búfer se traduce en un menor número de escrituras en S3, un menor costo de compactación debido a que los archivos de datos son más grandes y un tiempo de ejecución de consultas más rápido, pero con una latencia más alta. Los valores de sugerencia de búfer más bajos proporcionan los datos con una latencia más baja.

Configuración de opciones avanzadas

Puede configurar el cifrado del lado del servidor, el registro de errores, los permisos y las etiquetas para sus tablas de Apache Iceberg. Para obtener más información, consulte Configuración de opciones avanzadas. Debe agregar el rol de IAM que creó como parte del ???. Firehose lo asumirá para acceder a las tablas de AWS Glue y escribir en los buckets de Amazon S3.

La creación del flujo de Firehose puede tardar varios minutos en completarse. Después de crear correctamente el flujo de Firehose, puede empezar a ingerir datos en este y verlos en las tablas de Apache Iceberg.

Enrutamiento de los registros entrantes a una sola tabla de Iceberg

Si quiere que Firehose inserte datos en una sola tabla de Iceberg, simplemente configure una única base de datos y tabla en la configuración del flujo, como se muestra en el siguiente ejemplo de JSON. En el caso de una sola tabla, no se necesita la expresión JQ ni la función de Lambda para proporcionar la información de enrutamiento a Firehose. Si proporciona estos campos junto con JQ o Lambda, Firehose tomará la entrada de JQ o Lambda.

```
[
  {
    "DestinationDatabaseName": "UserEvents",
    "DestinationTableName": "customer_id",
    "UniqueKeys": [
      "COLUMN_PLACEHOLDER"
    ],
    "S3ErrorOutputPrefix": "OPTIONAL_PREFIX_PLACEHOLDER"
  }
]
```

En este ejemplo, Firehose enruta todos los registros de entrada a la tabla de `customer_id` en la base de datos `UserEvents`. Si desea realizar operaciones de actualización o eliminación en una sola tabla, debe proporcionar la operación para cada registro entrante a Firehose mediante el método o el JSONQuerymétodo Lambda.

Enrutamiento de los registros entrantes a diferentes tablas de Iceberg

Amazon Data Firehose puede enrutar los registros entrantes de un flujo a diferentes tablas de Iceberg en función del contenido del registro. Los registros no se mantienen en orden cuando se entregan desde Amazon Data Firehose. Observe el siguiente registro de entrada de ejemplo.

```
{
  "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"
}
```

En este ejemplo, el campo **`deviceId`** tiene dos valores posibles: `Device1234` y `Device4567`. Cuando un registro entrante tiene un campo **`deviceId`** como `Device1234`, vamos a escribir el registro en una tabla de Iceberg denominada `Device1234`, y cuando un registro entrante tiene un campo **`deviceId`** como `Device4567`, vamos a escribir el registro en una tabla denominada `Device4567`.

Tenga en cuenta que los registros con `Device1234` y `Device4567` pueden tener un conjunto diferente de campos que se asignan a diferentes columnas de la tabla de Iceberg correspondiente. Los registros entrantes pueden tener una estructura JSON anidada que permita que el **`deviceId`** esté anidado dentro del registro JSON. En las próximas secciones, analizaremos cómo se pueden enrutar los registros a diferentes tablas proporcionando la información de enrutamiento adecuada a Firehose en estas situaciones.

Proporcione información de enrutamiento a Firehose con expresión JSONQuery

La forma más sencilla y rentable de proporcionar información de enrutamiento de registros a Firehose es proporcionar una JSONQuery expresión. Con este enfoque, se proporcionan JSONQuery expresiones para tres parámetros: `Database NameTable Name`, y (opcionalmente). `Operation` Firehose usa la expresión que usted proporciona para extraer información de los registros de los flujos entrantes para enrutarlos.

El parámetro `Database Name` especifica el nombre de la base de datos de destino. El parámetro `Table Name` especifica el nombre de la tabla de destino. `Operation` es un parámetro opcional que indica si se debe insertar el registro del flujo entrante como un registro nuevo en la tabla de destino o si se debe modificar o eliminar un registro existente en la tabla de destino. El campo Operación debe tener uno de los siguientes valores: `insert`, `update` o `delete`.

Para cada uno de estos tres parámetros, puede proporcionar un valor estático o una expresión dinámica en la que el valor se recupere del registro entrante. Por ejemplo, si desea entregar todos los registros de flujos entrantes a una única base de datos denominada `IoTevents`, el Nombre de la base de datos tendría un valor estático de `“IoTevents”`. Si el nombre de la tabla de destino debe obtenerse de un campo del registro entrante, el nombre de la tabla es una expresión dinámica que especifica el campo del registro entrante del que se debe recuperar el nombre de la tabla de destino.

En el siguiente ejemplo, utilizamos un valor estático para el Nombre de la base de datos, un valor dinámico para el Nombre de la tabla y un valor estático para la operación. Tenga en cuenta que especificar la operación es opcional. Si no se especifica ninguna operación, Firehose inserta los registros entrantes en la tabla de destino como registros nuevos de forma predeterminada.

```
Database Name : "IoTevents"
Table Name : .deviceId
Operation : "insert"
```

Si el campo `deviceId` está anidado dentro del registro JSON, hay que especificar el nombre de la tabla con la información del campo anidado como `.event.deviceId`.

> **Note**
>
> - Si especificas la operación como `update` o`delete`, debes especificar claves únicas para la tabla de destino cuando configuras tu transmisión Firehose, o establecerla identifier-field-idsen Iceberg cuando ejecutas las operaciones de creación de tabla o modificación de tabla en Iceberg. Si no se especifica, Firehose generará un error y enviará los datos a un bucket de errores de S3.
> - Los valores de `Database Name` y `Table Name` deben coincidir exactamente con los nombres de la base de datos y la tabla de destino. Si no coinciden, Firehose arroja un error y envía los datos a un bucket de errores de S3.

Suministro de información de enrutamiento mediante una función de AWS Lambda

Puede haber situaciones en las que tenga reglas complejas que determinen cómo enrutar los registros entrantes a una tabla de destino. Por ejemplo, puede tener una regla que defina si un campo contiene el valor A, B o F, que deba enrutarse a una tabla de destino denominada `TableX` o puede que desee aumentar el registro de flujos entrantes añadiendo atributos adicionales. Por ejemplo, si un registro contiene un campo `device_id` como 1, puede añadir otro campo `device_type` como “módem” y escribir el campo adicional en la columna de la tabla de destino. En esos casos, puede transformar el flujo de origen mediante una AWS Lambda función de Firehose y proporcionar información de enrutamiento como parte del resultado de la función de transformación Lambda. Para saber cómo puede transformar la transmisión de origen mediante una AWS Lambda función de Firehose, consulte Transformar los datos de origen en Amazon Data Firehose.

Cuando se utiliza Lambda para la transformación del flujo de origen en Firehose, la salida debe contener los parámetros `recordId`, `result` y `data` o `KafkaRecordValue`. El parámetro `recordId` contiene el registro del flujo de entrada, `result` indica si la transformación se ha realizado correctamente, y `data` contiene la salida transformada codificada en Base64 de la función de Lambda. Para obtener más información, consulte ???.

```
{
  "recordId": "49655962066601463032522589543535113056108699331451682818000000",
```

```
  "result": "Ok",
  "data": "1IiwiI6ICJmYWxsIiwgImdgU21IiwiI6ICJmYWxsIiwg==tcHV0ZXIgU2NpZW5jZSIsICJzZW1"
}
```

Para especificar la información de enrutamiento a Firehose sobre cómo enrutar el registro del flujo a una tabla de destino como parte de la función de Lambda, el resultado de la función de Lambda debe contener una sección adicional para `metadata`. En el siguiente ejemplo, se muestra cómo se agrega la sección de metadatos al resultado de Lambda para un flujo de Firehose que usa Kinesis Data Streams como origen de datos para indicar a Firehose que debe insertar el registro como un registro nuevo en la tabla denominada `Device1234` de la base de datos `IoTevents`.

```
 {
"recordId": "49655962066601463032522589543535113056108699331451682818000000",
    "result": "Ok",
    "data":
 "1IiwiI6ICJmYWxsIiwgImdgU21IiwiI6ICJmYWxsIiwg==tcHV0ZXIgU2NpZW5jZSIsICJzZW1",

    "metadata":{
"otfMetadata":{
            "destinationTableName":"Device1234",
            "destinationDatabaseName":"IoTevents",
            "operation":"insert"
        }
    }
  }
```

Del mismo modo, en el siguiente ejemplo se muestra cómo se puede añadir la sección de metadatos al resultado de Lambda para un Firehose que utilice Amazon Managed Streaming para Apache Kafka como origen de datos para indicarle a Firehose que debe insertar el registro como un registro nuevo en la tabla denominada `Device1234` en la base de datos `IoTevents`.

```
{
"recordId": "49655962066601463032522589543535113056108699331451682818000000",
    "result": "Ok",
    "kafkaRecordValue":
 "1IiwiI6ICJmYWxsIiwgImdgU21IiwiI6ICJmYWxsIiwg==tcHV0ZXIgU2NpZW5jZSIsICJzZW1",

    "metadata":{
"otfMetadata":{
            "destinationTableName":"Device1234",
            "destinationDatabaseName":"IoTevents",
```

```
            "operation":"insert"
        }
    }
}
```

En este ejemplo,

- `destinationDatabaseName` hace referencia al nombre de la base de datos de destino y es un campo obligatorio.
- `destinationTableName` hace referencia al nombre de la tabla de destino y es un campo obligatorio.
- `operation` es un campo opcional con los siguientes valores posibles: `insert`, `update` y `delete`. Si no especifica un valor, la operación predeterminada es `insert`.

> **Note**
>
> - Si especificas la operación como `update` o`delete`, debes especificar claves únicas para la tabla de destino cuando configuras tu transmisión Firehose, o establecerla identifier-field-idsen Iceberg cuando ejecutas las operaciones de creación de tabla o modificación de tabla en Iceberg. Si no se especifica, Firehose generará un error y enviará los datos a un bucket de errores de S3.
> - Los valores de `Database Name` y `Table Name` deben coincidir exactamente con los nombres de la base de datos y la tabla de destino. Si no coinciden, Firehose arroja un error y envía los datos a un bucket de errores de S3.
> - Cuando la transmisión de Firehose tiene una función de transformación Lambda y una expresión JSONQuery , Firehose comprueba primero el campo de metadatos en la salida de Lambda para determinar cómo enrutar el registro a la tabla de destino adecuada y, a continuación, examina el resultado de la expresión para ver si faltan campos. JSONQuery
>
>   Si la Lambda o la JSONQuery expresión no proporcionan la información de enrutamiento requerida, Firehose asume que se trata de un escenario de una sola tabla y busca la información de una sola tabla en la configuración de claves únicas.
>
>   Para obtener más información, consulte la tabla Enrutar los registros entrantes a una única tabla de Iceberg. Si Firehose no determina la información de enrutamiento ni hace coincidir

el registro con una tabla de destino específica, enviará los datos al bucket de errores de S3 especificado.

Función de Lambda de ejemplo

Esta función de Lambda es un ejemplo de código de Python que analiza los registros del flujo entrante y agrega los campos obligatorios para especificar cómo se deben escribir los datos en tablas específicas. Puede usar este código de ejemplo para agregar la sección de metadatos para la información de enrutamiento.

```
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
```

Monitorizar métricas

Para la entrega de datos a Apache Iceberg Tables, Firehose emite las CloudWatch siguientes métricas a nivel de flujo.

Métrica	Description (Descripción)
`DeliveryToIceberg.Bytes`	El número de bytes enviados a las tablas de Apache Iceberg durante el periodo de tiempo especificado. Unidades: bytes
`DeliveryToIceberg.IncomingRowCount`	Número de registros que Firehose intenta entregar a las tablas de Apache Iceberg. Unidades: recuento
`DeliveryToIceberg.SuccessfulRowCount`	Número de filas entregadas correctamente a las tablas de Apache Iceberg. Unidades: recuento
`DeliveryToIceberg.FailedRowCount`	Número de filas con errores que se enviaron al bucket de copias de seguridad de S3.

Métrica	Description (Descripción)
	Unidades: recuento
`DeliveryToIceberg.DataFreshness`	Antigüedad (desde que se incorporó a Firehose hasta ahora) del registro más antiguo de Firehose. Los registros anteriores a este valor se han enviado a las tablas de Apache Iceberg. Unidades: segundos
`DeliveryToIceberg.Success`	Suma de las confirmaciones correctas en las tablas de Apache Iceberg.
`JQProcessing.Duration`	El tiempo que se tardó en ejecutar la expresión JQ. Unidades: milisegundos

Comprensión de los tipos de datos compatibles

Firehose admite todos los tipos de datos primitivos y complejos que admite Apache Iceberg. Para obtener más información, consulte Esquemas y tipos de datos. Al enviar datos binarios como una cadena, debe usar los tipos de codificación compatibles con Firehose: Basic Base64, MIME Base64, Base64 seguro para URL y nombre de archivo y Hex. Los tipos de datos de marca de tiempo deben enviarse siempre en microsegundos.

Ejemplos de tipos de datos

En la siguiente sección, se muestran ejemplos de diferentes tipos de datos.

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 (Época en microsegundos LocalTime , objeto Java)

```
{
    "destination_column_0": 68175096000,
    "destination_column_1": "18:56:15.096"
}
```

TimestampType.withZone (Época en microsegundos, objeto Java, OffsetDateTime 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 (Época en microsegundos, objeto Java, objeto LocalDateTime Java, objeto Java) OffsetDateTime 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

Utilice los siguientes recursos para obtener más información:

- Ingresar datos en tiempo real a las tablas de Apache Iceberg en Amazon S3 mediante Amazon Data Firehose
- Optimice el análisis de AWS WAF registros con Apache Iceberg y Amazon Data Firehose
- Crear un lago de datos para transmitir datos con las tablas de Amazon S3 y Amazon Data Firehose

Etiquetado de un flujo de Firehose

Puede asignar sus propios metadatos a los flujos de Firehose que cree en Amazon Data Firehose mediante etiquetas. Una etiqueta es un par clave-valor definido por el usuario para un flujo. El uso de etiquetas es una forma sencilla pero eficaz de gestionar AWS los recursos y organizar los datos, incluidos los datos de facturación.

Puedes especificar etiquetas al invocar CreateDeliveryStreampara crear una nueva transmisión de Firehose. Para los flujos de Firehose existentes, puede añadir, enumerar y eliminar etiquetas con las siguientes tres operaciones:

- TagDeliveryStream
- ListTagsForDeliveryStream
- UntagDeliveryStream

Comprensión de los conceptos básicos sobre etiquetas

Puede utilizar la API de Amazon Data Firehose para llevar a cabo las siguientes tareas:

- Añadir etiquetas a un flujo de Firehose.
- Enumerar las etiquetas para sus flujos de Firehose.
- Eliminar etiquetas de un flujo de Firehose.

Puede utilizar las etiquetas para categorizar sus flujos de Firehose. Por ejemplo, puede clasificar los flujos de Firehose en categorías por objetivo, propietario o entorno. Dado que se define la clave y el valor de cada etiqueta, es posible crear un conjunto de categorías personalizadas para satisfacer sus necesidades específicas. Por ejemplo, podría definir un conjunto de etiquetas que le ayude a realizar un seguimiento de los flujos de Firehose por propietario y aplicaciones asociadas.

A continuación, se muestran varios ejemplos de etiquetas:

- `Project: Project name`
- `Owner: Name`
- `Purpose: Load testing`
- `Application: Application name`

- `Environment: Production`

Si se especifican etiquetas en la acción `CreateDeliveryStream`, Amazon Data Firehose realiza una autorización adicional en la acción `firehose:TagDeliveryStream` para verificar que los usuarios tengan permisos para crear etiquetas. Si no concede este permiso, las solicitudes para crear nuevos flujos de Firehose con etiquetas de recursos de IAM fallarán con un resultado `AccessDeniedException` como el siguiente.

```
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.
```

En el ejemplo siguiente, se muestra una política que permite que los usuarios creen un flujo de Firehose y apliquen etiquetas.

Seguimiento de los costos mediante el etiquetado

Puedes usar etiquetas para categorizar y hacer un seguimiento de tus costos. AWS Cuando aplicas etiquetas a tus AWS recursos, incluidas las transmisiones de Firehose, tu informe de asignación de AWS costos incluye el uso y los costos agregados por etiquetas. Si aplica etiquetas que representen categorías de negocio (por ejemplo, centros de costos, nombres de aplicaciones o propietarios), puede organizar los costos entre diferentes servicios. Para obtener más información, consulte Utilizar etiquetas de asignación de costos para informes de facturación personalizados en la Guía del usuario de AWS Billing .

Conozca las restricciones de las etiquetas

Se aplican las siguientes restricciones a las etiquetas en Amazon Data Firehose.

Restricciones básicas

- El número máximo de etiquetas por recurso (flujo) es 50.
- Las claves y los valores de las etiquetas distinguen mayúsculas de minúsculas.
- No se pueden cambiar ni editar etiquetas de un flujo eliminado.

Restricciones de clave de etiqueta

- Cada clave de etiqueta debe ser única. Si agrega una etiqueta con una clave que ya está en uso, la nueva etiqueta sobrescribe el par clave-valor existente.
- Una clave de etiqueta no puede comenzar por `aws:` porque este prefijo está reservado para su utilización por AWS. AWS crea etiquetas cuyo nombre comienza por este prefijo por usted, pero usted no puede editarlas ni eliminarlas.
- Las claves de etiqueta deben tener entre 1 y 128 caracteres Unicode de longitud.
- Las claves de etiquetas deben constar de los siguientes caracteres: letras Unicode, números, espacios en blanco y los siguientes caracteres especiales: `_ . / = + - @`.

Restricciones de valor de etiqueta

- Los valores de etiqueta deben tener entre 0 y 255 caracteres Unicode de longitud.
- Los valores de etiqueta pueden estar en blanco. De lo contrario, deben constar de los siguientes caracteres: letras Unicode, números, espacios en blanco y cualquiera de los siguientes caracteres especiales: `_ . / = + - @`.

Seguridad en Amazon Data Firehose

La seguridad en la nube AWS es la máxima prioridad. Como AWS cliente, se beneficiará de una arquitectura de centro de datos y red diseñada para cumplir con los requisitos de las organizaciones más sensibles a la seguridad.

La seguridad es una responsabilidad compartida entre usted AWS y usted. El modelo de responsabilidad compartida la describe como seguridad de la nube y seguridad en la nube:

- Seguridad de la nube: AWS es responsable de proteger la infraestructura que ejecuta AWS los servicios en la AWS nube. AWS también le proporciona servicios que puede utilizar de forma segura. Auditores externos prueban y verifican periódicamente la eficacia de nuestra seguridad en el marco de los programas de conformidad de AWS. Para más información acerca de los programas de conformidad que se aplican a Data Firehose, consulte Servicios de AWS en el ámbito del programa de conformidad.
- Seguridad en la nube: su responsabilidad viene determinada por el AWS servicio que utilice. Usted también es responsable de otros factores, incluida la confidencialidad de los datos, los requisitos de la empresa y la legislación y los reglamentos aplicables.

Esta documentación lo ayuda a comprender cómo aplicar el modelo de responsabilidad compartida cuando se utiliza Data Firehose. En los siguientes temas, se muestra el proceso de configuración de Data Firehose para satisfacer sus objetivos de seguridad y conformidad. También aprenderás a usar otros AWS servicios que pueden ayudarte a monitorear y proteger tus recursos de Data Firehose.

Temas

Protección de datos en Amazon Data Firehose

Amazon Data Firehose cifra todos los datos en tránsito mediante el protocolo TLS. Además, en el caso de los datos almacenados en un almacenamiento provisional durante el procesamiento, Amazon Data Firehose cifra los datos mediante AWS Key Management Service y verifica su integridad con la verificación por suma de comprobación.

Si tiene datos confidenciales, puede habilitar el cifrado de datos del servidor al utilizar Amazon Data Firehose. La forma de hacerlo dependerá del origen de los datos.

> **Note**
>
> Si necesita módulos criptográficos validados por FIPS 140-2 para acceder a AWS través de una interfaz de línea de comandos o una API, utilice un terminal FIPS. Para obtener más información sobre los puntos de conexión de FIPS disponibles, consulte Estándar de procesamiento de la información federal (FIPS) 140-2.

Cifrado del servidor con Kinesis Data Streams

Cuando envía datos de sus productores de datos a su transmisión de datos, Kinesis Data Streams cifra los datos mediante AWS Key Management Service una clave AWS KMS() antes de almacenarlos en reposo. Cuando el flujo de Firehose lee los datos del flujo de datos, Kinesis Data Streams primero descifra los datos y, a continuación, los envía a Amazon Data Firehose. Amazon Data Firehose almacena los datos en la memoria búfer en función de las sugerencias de almacenamiento en búfer que se hayan especificado. A continuación, los envía a los destinos sin tener que almacenar los datos no cifrados en reposo.

Para obtener información sobre cómo habilitar el cifrado del servidor para Kinesis Data Streams, consulte Using Server-Side Encryption en la Guía para desarrolladores de Amazon Kinesis Data Streams.

Cifrado del servidor con Direct PUT u otros orígenes de datos

Si envías datos a tu transmisión de Firehose mediante Amazon Logs PutRecordo Events PutRecordBatch, o si envías los datos mediante AWS IoT Amazon CloudWatch Logs o CloudWatch Events, puedes activar el cifrado del lado del servidor mediante esta operación. StartDeliveryStreamEncryption

Para detenerlo server-side-encryption, usa la operación. StopDeliveryStreamEncryption

También puede habilitar SSE al crear el flujo de Firehose. Para ello, especifique DeliveryStreamEncryptionConfigurationInputcuándo se invoca CreateDeliveryStream.

Cuando la CMK es de tipo `CUSTOMER_MANAGED_CMK`, si el servicio Amazon Data Firehose no puede descifrar registros debido a una excepción `KMSNotFoundException`, `KMSInvalidStateException`, `KMSDisabledException` o `KMSAccessDeniedException`, el servicio espera hasta 24 horas (el periodo de retención) para resolver el problema. Si el problema continúa después del periodo de retención, el servicio omite los registros que han superado el periodo de retención y no se pudieron descifrar y, a continuación, descarta los datos. Amazon Data Firehose proporciona las siguientes cuatro CloudWatch métricas que puede utilizar para realizar un seguimiento de las cuatro AWS KMS excepciones:

- `KMSKeyAccessDenied`
- `KMSKeyDisabled`
- `KMSKeyInvalidState`
- `KMSKeyNotFound`

Para obtener más información sobre estas métricas, consulte the section called “Supervisión con métricas de CloudWatch”.

Important

Para cifrar tu transmisión de Firehose, usa symmetric. CMKs Amazon Data Firehose no admite la asimetría. CMKs Para obtener información acerca de lo simétrico y lo asimétrico CMKs, consulte Acerca de lo simétrico y lo CMKs asimétrico en la guía para desarrolladores. AWS Key Management Service

Note

Cuando utiliza una clave administrada por el cliente (CUSTOMER_MANAGED_CMK) para habilitar el cifrado del servidor (SSE) para el flujo de Firehose, el servicio de Firehose establece un contexto de cifrado siempre que utilice su clave. Como este contexto de cifrado representa un caso en el que se utilizó una clave propiedad de su AWS cuenta, se registra como parte de los registros de AWS CloudTrail eventos de su cuenta. AWS Este contexto

de cifrado es un sistema generado por el servicio de Firehose. Su aplicación no debe hacer suposiciones sobre el formato o el contenido del contexto de cifrado establecido por el servicio de Firehose.

Control del acceso con Amazon Data Firehose

En las siguientes secciones, se explica cómo controlar el acceso con los recursos de Amazon Data Firehose como origen y destino. Estas secciones incluyen información sobre cómo conceder acceso a su aplicación para que pueda enviar datos a su flujo de Firehose. También describen cómo puede conceder a Amazon Data Firehose acceso a su bucket de Amazon Simple Storage Service (Amazon S3), clúster de Amazon Redshift o clúster de Amazon OpenSearch Service, así como los permisos de acceso que necesita si utiliza Datadog, Dynatrace, LogicMonitor MongoDB, New Relic, Splunk o Sumo Logic como destino. Por último, en este tema encontrará instrucciones de configuración de Amazon Data Firehose para que pueda entregar datos en un destino que pertenezca a otra cuenta de AWS . La tecnología para administrar todas estas formas de acceso es (IAM). AWS Identity and Access Management Para obtener más información acerca de IAM, consulte ¿Qué es IAM?.

Contenido

- Ingesta de registros de flujo de VPC en Splunk mediante Amazon Data Firehose
- Acceso al punto de conexión HTTP o Snowflake
- Concesión a Firehose de acceso a un destino de Snowflake
- Acceso a Snowflake en VPC
- Concesión a Firehose de acceso a un destino de punto de conexión HTTP
- Entrega entre cuentas desde Amazon MSK
- Entrega entre cuentas en un destino de Amazon S3
- Entrega multicuenta a un OpenSearch destino del servicio
- Uso de etiquetas para controlar el acceso

Concesión de acceso a los recursos de Firehose

Para concederle a su aplicación acceso a su flujo de Firehose, utilice una política similar a este ejemplo. Puede ajustar las operaciones individuales de las API a las que concede acceso modificando la sección `Action` o concediendo acceso a todas las operaciones `"firehose:*"`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "firehose:DeleteDeliveryStream",
                "firehose:PutRecord",
                "firehose:PutRecordBatch",
                "firehose:UpdateDestination"
            ],
            "Resource": [
                "arn:aws:firehose:us-east-1:123456789012:deliverystream/delivery-stream-name"
            ]
        }
    ]
}
```

Concesión a Firehose de acceso a un clúster privado de Amazon MSK

Si el origen del flujo de Firehose es un clúster privado de Amazon MSK, utilice una política similar a la de este ejemplo.

Debe agregar una política como esta a la política basada en recursos del clúster para concederle a la entidad principal del servicio de Firehose el permiso para invocar la operación de la API `CreateVpcConnection` de Amazon MSK.

Permiso para que Firehose asuma un rol de IAM

En esta sección, se describen los permisos y las políticas que conceden a Amazon Data Firehose acceso para ingerir, procesar y entregar los datos del origen al destino.

> **Note**
>
> Si utilizas la consola para crear una transmisión de Firehose y eliges la opción de crear una nueva función, AWS adjunta la política de confianza necesaria a la función. Si quiere que Amazon Data Firehose utilice un rol de IAM existente o si crea un rol por su cuenta, adjunte las siguientes políticas de confianza a dicho rol para que Amazon Data Firehose pueda asumirlo. Edita las políticas para sustituirlas por tu *`account-id`* ID de AWS cuenta. Para obtener información acerca de cómo modificar la relación de confianza de un rol, consulte Modificación de un rol.

Amazon Data Firehose utiliza un rol de IAM para todos los permisos que necesita el flujo de Firehose para procesar y entregar los datos. Asegúrese de que las siguientes políticas de confianza se hayan adjuntado a ese rol para que Amazon Data Firehose pueda asumirlo.

Si elige Amazon MSK como origen para su flujo de Firehose, debe especificar otro rol de IAM que conceda a Amazon Data Firehose permisos para ingerir datos de origen del clúster de Amazon MSK especificado. Asegúrese de que las siguientes políticas de confianza se hayan adjuntado a ese rol para que Amazon Data Firehose pueda asumirlo.

JSON

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

```
  "Statement": [
    {
      "Principal": {
        "Service": [
          "firehose.amazonaws.com"
        ]
      },
      "Effect": "Allow",
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Asegúrese de que este rol que concede a Amazon Data Firehose permisos para ingerir datos de origen del clúster de Amazon MSK especificado conceda los siguientes permisos:

Conceda a Firehose acceso a AWS Glue para la conversión de formatos de datos

Si el flujo de Firehose lleva a cabo la conversión de formatos de datos, Amazon Data Firehose hace referencia a las definiciones de tabla almacenadas en AWS Glue. Para conceder a Amazon Data Firehose el acceso necesario AWS Glue, añade la siguiente declaración a tu política. Para obtener información sobre cómo encontrar el ARN de la tabla, consulte Especificación del recurso AWS Glue. ARNs

```
{
    "Sid": "",
    "Effect": "Allow",
    "Action": [
        "glue:GetTable",
        "glue:GetTableVersion",
        "glue:GetTableVersions"
    ],
    "Resource": [
        "arn:aws:glue:us-east-1:123456789012:catalog",
        "arn:aws:glue:us-east-1:123456789012:database/b",
        "arn:aws:glue:us-east-1:123456789012:table/b/easd"
    ]
},
{
```

```
      actions: ['glue:GetSchemaVersion'],
      grantee: options.role,
      resourceArns: ['*'],
}
```

La política recomendada para obtener esquemas del registro de esquemas no tiene restricciones de recursos. Para obtener más información, consulte los ejemplos de deserializadores de IAM en la Guía para desarrolladores. AWS Glue

Concesión de acceso a Firehose a un destino de Amazon S3

Cuando utiliza un destino de Amazon S3, Amazon Data Firehose entrega los datos a su bucket de S3 y, si lo desea, puede utilizar una AWS KMS clave de su propiedad para el cifrado de datos. Si el registro de errores está activado, Amazon Data Firehose también envía los errores de entrega de datos al grupo de CloudWatch registros y a las transmisiones. Es obligatorio contar con un rol de IAM al crear un flujo de Firehose. Amazon Data Firehose asume esa función de IAM y obtiene acceso al bucket, la clave y el grupo de CloudWatch registros y las transmisiones especificados.

Utilice la siguiente política de acceso para permitir a Amazon Data Firehose acceder al bucket de S3 y a la clave de AWS KMS . Si no es el propietario del bucket de S3, agregue `s3:PutObjectAcl` a la lista de acciones de Amazon S3. De esta forma, se concede al propietario del bucket acceso completo a los objetos que entrega 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:us-east-1:123456789012:stream/stream-
name"
        },
        {
           "Effect": "Allow",
           "Action": [
               "kms:Decrypt",
               "kms:GenerateDataKey"
           ],
           "Resource": [
               "arn:aws:kms:us-east-1:123456789012:key/key-id"
           ],
           "Condition": {
               "StringEquals": {
                   "kms:ViaService": "s3.us-east-1.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:us-east-1:123456789012:log-group:log-group-name:log-
stream:log-stream-name"
           ]
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction",
                "lambda:GetFunctionConfiguration"
            ],
            "Resource": [
                "arn:aws:lambda:us-east-1:123456789012:function:function-
name:function-version"
            ]
        }
    ]
}
```

La política anterior también incluye una declaración que permite el acceso a Amazon Kinesis Data Streams. Si no utiliza Kinesis Data Streams como origen de datos, puede eliminar dicha declaración. Si utiliza Amazon MSK como origen, puede sustituir esa declaración por la siguiente:

```
{
   "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 obtener más información sobre cómo permitir que otros AWS servicios accedan a sus AWS recursos, consulte Creación de un rol para delegar permisos a un AWS servicio en la Guía del usuario de IAM.

Para obtener información sobre cómo conceder a Amazon Data Firehose acceso a un destino de Amazon S3 de otra cuenta, consulte the section called “Entrega entre cuentas en un destino de Amazon S3”.

Conceder a Firehose acceso a las tablas de Amazon S3

Debe disponer de un rol de IAM antes de crear un flujo de Firehose. Siga estos pasos para crear una política y un rol de IAM. Firehose asume este rol de IAM y lleva a cabo las acciones necesarias.

Inicie sesión en la consola de IAM Consola de administración de AWS y ábrala en. https://console.aws.amazon.com/iam/

Cree una política y elija JSON en el editor de políticas. Añada la siguiente política en línea que conceda permisos a Amazon S3, como read/write permisos, permisos para actualizar la tabla en el catálogo de datos y otros.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "S3TableAccessViaGlueFederation",
      "Effect": "Allow",
      "Action": [
        "glue:GetTable",
        "glue:GetDatabase",
        "glue:UpdateTable"
      ],
```

```
      "Resource": [
        "arn:aws:glue:us-east-1:123456789012:catalog/s3tablescatalog/*",
        "arn:aws:glue:us-east-1:123456789012:catalog/s3tablescatalog",
        "arn:aws:glue:us-east-1:123456789012:catalog",
        "arn:aws:glue:us-east-1:123456789012:database/*",
        "arn:aws:glue:us-east-1:123456789012: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:us-east-1:123456789012:stream/<stream-name>"
    },
    {
      "Sid":
"RequiredWhenDoingMetadataReadsANDDataAndMetadataWriteViaLakeformation",
      "Effect": "Allow",
      "Action": [
        "lakeformation:GetDataAccess"
      ],
      "Resource": "*"
    },
```

```
    {
      "Sid": "RequiredWhenUsingKMSEncryptionForS3ErrorBucketDelivery",
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
      ],
      "Resource": [
        "arn:aws:kms:us-east-1:123456789012:key/<KMS-key-id>"
      ],
      "Condition": {
        "StringEquals": {
          "kms:ViaService": "s3.us-east-1.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:us-east-1:123456789012:log-group:log-group-name>:log-
stream:<log-stream-name>"
      ]
    },
    {
      "Sid": "RequiredWhenAttachingLambdaToFirehose",
      "Effect": "Allow",
      "Action": [
        "lambda:InvokeFunction",
        "lambda:GetFunctionConfiguration"
      ],
      "Resource": [
        "arn:aws:lambda:us-east-1:123456789012:function:<function-
name>:<function-version>"
      ]
    }
  ]
```

```
}
```

La política incluye declaraciones que permiten el acceso a Amazon Kinesis Data Streams, la invocación de funciones de Lambda y el acceso a las claves. AWS KMS Si no utiliza ninguno de estos recursos, puede eliminar las declaraciones correspondientes. Si el registro de errores está activado, Amazon Data Firehose también envía los errores de entrega de datos al grupo de CloudWatch registros y a las transmisiones. Debe configurar los nombres de los grupos de registro y los flujos de registro para utilizar esta opción. Para obtener información sobre nombres de grupos de registros y flujos de registros, consulte Supervisión de Amazon Data Firehose mediante Registros de CloudWatch.

En las políticas integradas, `<error delivery bucket>` sustitúyalo por el nombre del bucket de Amazon S3 `aws-account-id` y la región por un Cuenta de AWS número y una región válidos del recurso.

Tras crear la política, abra la consola de IAM en https://console.aws.amazon.com/iam/y cree un rol de IAM con el tipo de Servicio de AWSentidad de confianza.

En Servicio o caso de uso, elija Kinesis. En Caso de uso, elija Kinesis Firehose.

En la página siguiente, elija la política creada en el paso anterior para asociarla a este rol. En la página de revisión, encontrará una política de confianza ya adjunta a este rol que otorga permisos al servicio de Firehose para que lo asuma. Al crear la función, Amazon Data Firehose puede asumir que realizará las operaciones necesarias en AWS Glue los buckets de S3. Agregue la entidad principal del servicio de Firehose a la política de confianza del rol que se crea. Para obtener más información, consulte Permiso para que Firehose asuma un rol de IAM.

Concesión a Firehose de acceso a un destino de tablas de Apache Iceberg

Debe tener un rol de IAM antes de crear un flujo de Firehose y tablas de Apache Iceberg mediante AWS Glue. Siga estos pasos para crear una política y un rol de IAM. Firehose asume este rol de IAM y lleva a cabo las acciones necesarias.

1. Inicie sesión en la consola de IAM Consola de administración de AWS y ábrala en. https://console.aws.amazon.com/iam/
2. Cree una política y elija JSON en el editor de políticas.
3. Añada la siguiente política en línea que conceda permisos a Amazon S3, como los read/write permisos, los permisos para actualizar la tabla en el catálogo de datos, etc.

Esta política incluye una declaración que permite el acceso a Amazon Kinesis Data Streams, la invocación de funciones de Lambda y el acceso a las claves de KMS. Si no utiliza ninguno de estos recursos, puede eliminar las declaraciones correspondientes.

Si el registro de errores está activado, Firehose también envía los errores de entrega de datos al grupo de CloudWatch registros y a las transmisiones. Para ello, debe configurar los nombres de los grupos de registro y los flujos de registro. Para obtener información sobre nombres de grupos de registros y flujos de registros, consulte Supervisión de Amazon Data Firehose mediante Registros de CloudWatch.

4. En las políticas integradas, *amzn-s3-demo-bucket* sustitúyalo por el nombre de tu bucket de Amazon S3 aws-account-id y la región por un Cuenta de AWS número y una región válidos de los recursos.

> **Note**
>
> Este rol otorga permisos a todas las bases de datos y tablas de su catálogo de datos. Si lo desea, puede conceder permisos solo a tablas y bases de datos específicas.

5. Tras crear la política, abra la consola de IAM y cree un rol de IAM con Servicio de AWS como el tipo de entidad de confianza.
6. En Servicio o caso de uso, elija Kinesis. Elija Kinesis Firehose como su caso de uso.
7. En la página siguiente, elija la política creada en el paso anterior para asociarla a este rol. En la página de revisión, encontrará una política de confianza ya adjunta a este rol que otorga permisos al servicio de Firehose para que lo asuma. Al crear el rol, Amazon Data Firehose puede asumirlo para realizar las operaciones necesarias en AWS Glue y los buckets de S3.

Concesión a Firehose de acceso a un destino de Amazon Redshift

Consulte la siguiente información al conceder acceso a Amazon Data Firehose mientras utiliza un destino de Amazon Redshift.

Temas

- Rol de IAM y política de acceso
- Acceso mediante VPC a un clúster aprovisionado de Amazon Redshift o un grupo de trabajo de Amazon Redshift sin servidor

Rol de IAM y política de acceso

Cuando se utiliza un destino de Amazon Redshift, Amazon Data Firehose entrega los datos en el bucket de S3 como una ubicación intermedia. Opcionalmente, puede usar una AWS KMS clave de su propiedad para el cifrado de datos. A continuación, Amazon Data Firehose carga los datos del bucket de S3 en el clúster aprovisionado de Amazon Redshift o el grupo de trabajo de Amazon Redshift sin servidor. Si el registro de errores está activado, Amazon Data Firehose también envía los errores de entrega de datos al grupo de CloudWatch registros y a las transmisiones. Amazon Data Firehose utiliza el nombre de usuario y la contraseña de Amazon Redshift especificados para acceder al clúster aprovisionado o al grupo de trabajo Amazon Redshift Serverless, y utiliza una función de IAM para acceder al bucket, la clave, el grupo de registros y las transmisiones especificados. CloudWatch Es obligatorio contar con un rol de IAM al crear un flujo de Firehose.

Utilice la siguiente política de acceso para permitir a Amazon Data Firehose acceder al bucket de S3 y a la clave de AWS KMS . Si el bucket de S3 no es de su propiedad, agregue `s3:PutObjectAcl` a la lista de acciones de Amazon S3, ya que le concede al propietario del bucket acceso completo a los objetos entregados por Amazon Data Firehose. Esta política también incluye una declaración que permite el acceso a Amazon Kinesis Data Streams. Si no utiliza Kinesis Data Streams como origen de datos, puede eliminar dicha declaración.

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:us-east-1:123456789012:key/key-id"
            ],
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "s3.us-east-1.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:us-east-1:123456789012:stream/stream-
name"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:123456789012:log-group:log-group-name:log-
stream:log-stream-name"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "lambda:InvokeFunction",
                "lambda:GetFunctionConfiguration"
            ],
            "Resource": [
                "arn:aws:lambda:us-east-1:123456789012:function:function-
name:function-version"
            ]
        }
    ]
}
```

Para obtener más información sobre cómo permitir que otros AWS servicios accedan a sus AWS recursos, consulte Creación de un rol para delegar permisos a un AWS servicio en la Guía del usuario de IAM.

Acceso mediante VPC a un clúster aprovisionado de Amazon Redshift o un grupo de trabajo de Amazon Redshift sin servidor

Si el clúster aprovisionado de Amazon Redshift o el grupo de trabajo de Amazon Redshift sin servidor está en una nube privada virtual (VPC), debe ser de acceso público y tener una dirección IP pública. Además, conceda a Amazon Data Firehose acceso a su clúster aprovisionado de Amazon Redshift o grupo de trabajo Amazon Redshift sin servidor; para ello, desbloquee las direcciones IP de Amazon Data Firehose. Actualmente, Amazon Data Firehose utiliza un bloque de CIDR para cada región disponible.

Region	Bloques CIDR
Este de EE. UU. (Ohio)	`13.58.135.96/27`
Este de EE. UU. (Norte de Virginia)	`52.70.63.192/27`
Oeste de EE. UU. (Norte de California)	`13.57.135.192/27`
Oeste de EE. UU. (Oregón)	`52.89.255.224/27`

Region	Bloques CIDR
AWS GovCloud (Este de EE. UU.)	18.253.138.96/27
AWS GovCloud (Estados Unidos-Oeste)	52.61.204.160/27
Canadá (centro)	35.183.92.128/27
Oeste de Canadá (Calgary)	40.176.98.192/27
Asia-Pacífico (Hong Kong)	18.162.221.32/27
Asia-Pacífico (Mumbai)	13.232.67.32/27
Asia-Pacífico (Hyderabad)	18.60.192.128/27
Asia-Pacífico (Seúl)	13.209.1.64/27
Asia-Pacífico (Singapur)	13.228.64.192/27
Asia-Pacífico (Sídney)	13.210.67.224/27
Asia-Pacífico (Yakarta)	108.136.221.64/27
Asia-Pacífico (Tokio)	13.113.196.224/27
Asia-Pacífico (Osaka)	13.208.177.192/27
Asia-Pacífico (Tailandia)	43.208.112.96/27
Asia-Pacífico (Taipéi)	43.212.53.160/27
China (Pekín)	52.81.151.32/27
China (Ningxia)	161.189.23.64/27

Region	Bloques CIDR
Europa (Zúrich)	16.62.183.32/27
Europa (Fráncfort)	35.158.127.160/27
Europa (Irlanda)	52.19.239.192/27
Europa (Londres)	18.130.1.96/27
Europa (París)	35.180.1.96/27
Europa (Estocolmo)	13.53.63.224/27
Europa (España)	18.100.71.96/27
Middle East (Bahrain)	15.185.91.0/27
México (centro)	78.12.207.32/27
América del Sur (São Paulo)	18.228.1.128/27
Europa (Milán)	15.161.135.128/27
África (Ciudad del Cabo)	13.244.121.224/27
Medio Oriente (EAU)	3.28.159.32/27
Israel (Tel Aviv)	51.16.102.0/27
Asia-Pacífico (Melbourne)	16.50.161.128/27
Asia-Pacífico (Malasia)	43.216.58.0/27

Para obtener más información acerca de cómo desbloquear direcciones IP, consulte el paso Autorización de acceso al clúster en la Guía de introducción a Amazon Redshift.

Conceda a Firehose acceso a un destino de servicio público OpenSearch

Cuando utiliza un destino de OpenSearch servicio, Amazon Data Firehose envía los datos a su clúster de OpenSearch servicios y, al mismo tiempo, realiza copias de seguridad de todos los documentos fallidos o de todos los documentos en su bucket de S3. Si el registro de errores está activado, Amazon Data Firehose también envía los errores de entrega de datos al grupo de CloudWatch registros y a las transmisiones. Amazon Data Firehose utiliza una función de IAM para acceder al dominio de OpenSearch servicio, al bucket de S3, a la AWS KMS clave y al grupo de CloudWatch registros y a las transmisiones especificados. Es obligatorio contar con un rol de IAM al crear un flujo de Firehose.

Utilice la siguiente política de acceso para permitir que Amazon Data Firehose acceda a su bucket, dominio de OpenSearch servicio y AWS KMS clave de S3. Si el bucket de S3 no es de su propiedad, agregue `s3:PutObjectAcl` a la lista de acciones de Amazon S3, ya que le concede al propietario del bucket acceso completo a los objetos entregados por Amazon Data Firehose. Esta política también incluye una declaración que permite el acceso a Amazon Kinesis Data Streams. Si no utiliza Kinesis Data Streams como origen de datos, puede eliminar dicha declaración.

Para obtener más información sobre cómo permitir que otros AWS servicios accedan a sus AWS recursos, consulte Creación de un rol para delegar permisos a un AWS servicio en la Guía del usuario de IAM.

Para obtener información sobre cómo conceder a Amazon Data Firehose acceso a un clúster de OpenSearch servicios de otra cuenta, consulte. the section called “Entrega multicuenta a un OpenSearch destino del servicio”

Otorgue a Firehose acceso a un destino de OpenSearch servicio en una VPC

Si su dominio de OpenSearch servicio está en una VPC, asegúrese de conceder a Amazon Data Firehose los permisos que se describen en la sección anterior. Además, debe conceder a Amazon Data Firehose los siguientes permisos para que pueda acceder a la VPC de su dominio OpenSearch de servicio.

- `ec2:DescribeVpcs`
- `ec2:DescribeVpcAttribute`
- `ec2:DescribeSubnets`
- `ec2:DescribeSecurityGroups`

- `ec2:DescribeNetworkInterfaces`
- `ec2:CreateNetworkInterface`
- `ec2:CreateNetworkInterfacePermission`
- `ec2:DeleteNetworkInterface`

> **Important**
>
> No revoque estos permisos después de crear el flujo de Firehose. Si revocas estos permisos, tu transmisión de Firehose se degradará o dejará de entregar datos a OpenSearch tu dominio de servicio cada vez que el servicio intente realizar consultas o actualizarlos. ENIs

> **Important**
>
> Cuando especifique subredes para entregar datos al destino en una VPC privada, asegúrese de tener una cantidad suficiente de direcciones IP libres en las subredes elegidas. Si no hay una dirección IP libre disponible en una subred específica, Firehose no podrá crear ni ENIs añadir para la entrega de datos en la VPC privada, y la entrega se degradará o fallará.

Cuando creas o actualizas tu transmisión de Firehose, especificas un grupo de seguridad para que Firehose lo utilice cuando envíe datos a tu dominio de servicio. OpenSearch Puedes usar el mismo grupo de seguridad que usa el dominio del OpenSearch servicio o uno diferente. Si especifica un grupo de seguridad diferente, asegúrese de que permita el tráfico HTTPS saliente al grupo de seguridad del dominio del OpenSearch servicio. Asegúrese también de que el grupo de seguridad del dominio de OpenSearch servicio permita el tráfico HTTPS desde el grupo de seguridad que especificó al configurar la transmisión de Firehose. Si utilizas el mismo grupo de seguridad tanto para la transmisión de Firehose como para el dominio de OpenSearch servicio, asegúrate de que la regla de entrada del grupo de seguridad permita el tráfico HTTPS. Para obtener más información acerca de las reglas de los grupos de seguridad, consulte Reglas del grupo de seguridad en la documentación de Amazon VPC.

Conceda a Firehose acceso a un destino público sin servidor OpenSearch

Cuando utiliza un destino OpenSearch sin servidor, Amazon Data Firehose envía los datos a OpenSearch su colección sin servidor y, al mismo tiempo, realiza copias de seguridad de todos

los documentos fallidos o de todos los documentos en su bucket de S3. Si el registro de errores está activado, Amazon Data Firehose también envía los errores de entrega de datos al grupo de CloudWatch registros y a las transmisiones. Amazon Data Firehose utiliza una función de IAM para acceder a la colección OpenSearch Serverless, el bucket de S3, la AWS KMS clave y el grupo de CloudWatch registros y las transmisiones especificados. Es obligatorio contar con un rol de IAM al crear un flujo de Firehose.

Utilice la siguiente política de acceso para permitir que Amazon Data Firehose acceda al bucket de S3, al dominio OpenSearch sin servidor y a la clave. AWS KMS Si el bucket de S3 no es de su propiedad, agregue `s3:PutObjectAcl` a la lista de acciones de Amazon S3, ya que le concede al propietario del bucket acceso completo a los objetos entregados por Amazon Data Firehose. Esta política también incluye una declaración que permite el acceso a Amazon Kinesis Data Streams. Si no utiliza Kinesis Data Streams como origen de datos, puede eliminar dicha declaración.

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:us-east-1:123456789012:key/key-id"
            ],
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "s3.us-east-1.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:us-east-1:123456789012:stream/stream-name"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:123456789012:log-group:log-group-name:log-
stream:log-stream-name"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction",
                "lambda:GetFunctionConfiguration"
            ],
            "Resource": [
                "arn:aws:lambda:us-east-1:123456789012:function:function-
name:function-version"
            ]
        },
        {
```

```
            "Effect": "Allow",
            "Action": "aoss:APIAccessAll",
            "Resource": "arn:aws:aoss:us-east-1:123456789012:collection/collection-
id"
        }
    ]
}
```

Además de la política anterior, también debe configurar Amazon Data Firehose para que se asignen los siguientes permisos mínimos en una política de acceso a los datos:

```
[
   {
      "Rules":[
         {
            "ResourceType":"index",
            "Resource":[
               "index/target-collection/target-index"
            ],
            "Permission":[
               "aoss:WriteDocument",
               "aoss:UpdateIndex",
               "aoss:CreateIndex"
            ]
         }
      ],
      "Principal":[
         "arn:aws:sts::123456789012:assumed-role/firehose-delivery-role-name/*"
      ]
   }
]
```

Para obtener más información sobre cómo permitir que otros AWS servicios accedan a sus AWS recursos, consulte Creación de un rol para delegar permisos a un AWS servicio en la Guía del usuario de IAM.

Otorgue a Firehose acceso a un destino OpenSearch sin servidor en una VPC

Si su colección OpenSearch sin servidor se encuentra en una VPC, asegúrese de conceder a Amazon Data Firehose los permisos que se describen en la sección anterior. Además, debe conceder a Amazon Data Firehose los siguientes permisos para que pueda acceder a la VPC de su colección OpenSearch Serverless.

- `ec2:DescribeVpcs`
- `ec2:DescribeVpcAttribute`
- `ec2:DescribeSubnets`
- `ec2:DescribeSecurityGroups`
- `ec2:DescribeNetworkInterfaces`
- `ec2:CreateNetworkInterface`
- `ec2:CreateNetworkInterfacePermission`
- `ec2:DeleteNetworkInterface`

> **Important**
>
> No revoque estos permisos después de crear el flujo de Firehose. Si revocas estos permisos, tu transmisión de Firehose se degradará o dejará de entregar datos a OpenSearch tu dominio de servicio cada vez que el servicio intente realizar consultas o actualizarlos. ENIs

> **Important**
>
> Cuando especifique subredes para entregar datos al destino en una VPC privada, asegúrese de tener una cantidad suficiente de direcciones IP libres en las subredes elegidas. Si no hay una dirección IP libre disponible en una subred específica, Firehose no podrá crear ni ENIs añadir para la entrega de datos en la VPC privada, y la entrega se degradará o fallará.

Cuando creas o actualizas tu transmisión de Firehose, especificas un grupo de seguridad para que Firehose lo use cuando envíe datos a tu colección Serverless. OpenSearch Puedes usar el mismo

grupo de seguridad que usa la colección OpenSearch Serverless o uno diferente. Si especifica un grupo de seguridad diferente, asegúrese de que permita el tráfico HTTPS saliente al grupo de seguridad de la colección OpenSearch Serverless. Asegúrese también de que el grupo de seguridad de la colección OpenSearch Serverless permita el tráfico HTTPS desde el grupo de seguridad que especificó al configurar la transmisión Firehose. Si utilizas el mismo grupo de seguridad tanto para la transmisión de Firehose como para la colección OpenSearch Serverless, asegúrate de que la regla de entrada del grupo de seguridad permita el tráfico HTTPS. Para obtener más información acerca de las reglas de los grupos de seguridad, consulte Reglas del grupo de seguridad en la documentación de Amazon VPC.

Concesión a Firehose de acceso a un destino de Splunk

Cuando se utiliza un destino de Splunk, Amazon Data Firehose entrega los datos en el punto de conexión del recopilador de eventos HTTP (HEC) de Splunk. También hace una copia de seguridad de esos datos en el depósito de Amazon S3 que especifique y, si lo desea, puede utilizar una AWS KMS clave de su propiedad para el cifrado del lado del servidor de Amazon S3. Si el registro de errores está activado, Firehose envía los errores de entrega de datos a sus flujos de CloudWatch registro. También se puede utilizar AWS Lambda para la transformación de datos.

Si utilizas un balanceador de AWS cargas, asegúrate de que sea un Classic Load Balancer o un Application Load Balancer. Además, habilite las sesiones persistentes basadas en la duración con la caducidad de las cookies deshabilitada para el equilibrador de carga clásico y la caducidad establecida en el máximo (7 días) para el equilibrador de carga de aplicación. Para obtener información sobre cómo hacerlo, consulte Persistencia de la sesión basada en la duración para el equilibrador de carga clásico o el equilibrador de carga de aplicación.

Debe disponer de un rol de IAM al crear un flujo de Firehose. Firehose asume esa función de IAM y obtiene acceso al bucket, la clave y el grupo de CloudWatch registros y las transmisiones especificados.

Utilice la siguiente política de acceso para permitir a Amazon Data Firehose acceder al bucket de S3. Si el bucket de S3 no es de su propiedad, agregue `s3:PutObjectAcl` a la lista de acciones de Amazon S3, ya que le concede al propietario del bucket acceso completo a los objetos entregados por Amazon Data Firehose. Esta política también otorga a Amazon Data Firehose acceso para el registro de errores y CloudWatch AWS Lambda para la transformación de datos. La política también incluye una declaración que permite el acceso a Amazon Kinesis Data Streams. Si no utiliza Kinesis Data Streams como origen de datos, puede eliminar dicha declaración. Amazon Data Firehose no usa IAM para acceder a Splunk. Para tener acceso a Splunk, utiliza el token de 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:us-east-1:123456789012:key/key-id"
           ],
           "Condition": {
               "StringEquals": {
                   "kms:ViaService": "s3.us-east-1.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:us-east-1:123456789012:stream/stream-
name"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:123456789012:log-group:log-group-name:log-
stream:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction",
                "lambda:GetFunctionConfiguration"
            ],
            "Resource": [
                "arn:aws:lambda:us-east-1:123456789012:function:function-
name:function-version"
            ]
        }
    ]
}
```

Para obtener más información sobre cómo permitir que otros AWS servicios accedan a sus AWS recursos, consulte Creación de un rol para delegar permisos a un AWS servicio en la Guía del usuario de IAM.

Acceso a Splunk en VPC

Si la plataforma Splunk está en una VPC, debe ser de acceso público y tener una dirección IP pública. Además, conceda a Amazon Data Firehose acceso a la plataforma Splunk; para ello,

desbloquee las direcciones IP de Amazon Data Firehose. Actualmente, Amazon Data Firehose utiliza los siguientes bloques de CIDR.

Region	Bloques CIDR
Este de EE. UU. (Ohio)	`18.216.68.160/27, 18.216.170.64/27, 18.216.170.96/27 \`
Este de EE. UU. (Norte de Virginia)	`34.238.188.128/26, 34.238.188.192/26, 34.238.195.0/26`
Oeste de EE. UU. (Norte de California)	`13.57.180.0/26`
Oeste de EE. UU. (Oregón)	`34.216.24.32/27, 34.216.24.192/27, 34.216.24.224/27`
AWS GovCloud (Este de EE. UU.)	`18.253.138.192/26`
AWS GovCloud (Estados Unidos-Oeste)	`52.61.204.192/26`
Asia-Pacífico (Hong Kong)	`18.162.221.64/26`
Asia-Pacífico (Mumbai)	`13.232.67.64/26`
Asia-Pacífico (Seúl)	`13.209.71.0/26`
Asia-Pacífico (Singapur)	`13.229.187.128/26`
Asia-Pacífico (Sídney)	`13.211.12.0/26`
Asia-Pacífico (Tailandia)	`43.208.112.128/26`
Asia-Pacífico (Tokio)	`13.230.21.0/27, 13.230.21.32/27`
Canadá (centro)	`35.183.92.64/26`

Region	Bloques CIDR
Oeste de Canadá (Calgary)	40.176.98.128/26
Europa (Fráncfort)	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 (París)	35.180.112.0/26
Europa (España)	18.100.194.0/26
Europa (Estocolmo)	13.53.191.0/26
Middle East (Bahrain)	15.185.91.64/26
México (centro)	78.12.207.64/26
América del Sur (São Paulo)	18.228.1.192/26
Europa (Milán)	15.161.135.192/26
África (Ciudad del Cabo)	13.244.165.128/26
Asia-Pacífico (Osaka)	13.208.217.0/26
China (Pekín)	52.81.151.64/26
China (Ningxia)	161.189.23.128/26
Asia-Pacífico (Yakarta)	108.136.221.128/26
Medio Oriente (EAU)	3.28.159.64/26
Israel (Tel Aviv)	51.16.102.64/26

Region	Bloques CIDR
Europa (Zúrich)	`16.62.183.64/26`
Asia-Pacífico (Hyderabad)	`18.60.192.192/26`
Asia-Pacífico (Melbourne)	`16.50.161.192/26`
Asia-Pacífico (Malasia)	`43.216.44.192/26`

Ingesta de registros de flujo de VPC en Splunk mediante Amazon Data Firehose

Para obtener más información sobre cómo crear una suscripción de registro de flujo de VPC, publicar en Firehose y enviar los registros de flujo de VPC a un destino compatible, consulte Ingesta de registros de flujo de VPC en Splunk mediante Amazon Data Firehose.

Acceso al punto de conexión HTTP o Snowflake

No hay ningún subconjunto de rangos de direcciones IP de AWS específico de Amazon Data Firehose cuando el destino es un punto de conexión HTTP o clústeres públicos de Snowflake.

Para añadir Firehose a una lista de permitidos para clústeres públicos de Snowflake o a los puntos de conexión HTTP o HTTPS públicos, añada todos los rangos de direcciones IP de AWS actuales a las reglas de entrada.

> **Note**
>
> Las notificaciones no siempre provienen de direcciones IP de la misma AWS región que el tema asociado. Debes incluir el rango de direcciones AWS IP de todas las regiones.

Concesión a Firehose de acceso a un destino de Snowflake

Cuando utiliza Snowflake como destino, Firehose envía los datos a una cuenta de Snowflake mediante la URL de la cuenta de Snowflake. También hace copias de seguridad de los datos de

error en el depósito de Amazon Simple Storage Service que especifique y, si lo desea, puede utilizar una AWS Key Management Service clave de su propiedad para el cifrado del lado del servidor de Amazon S3. Si el registro de errores está activado, Firehose envía los errores de entrega de datos a tus flujos de CloudWatch Logs.

Debe disponer de un rol de IAM antes de crear un flujo de Firehose. Firehose asume esa función de IAM y obtiene acceso al bucket, la clave y el grupo de CloudWatch registros y las transmisiones especificados. Utilice la siguiente política de acceso para permitir a Firehose obtener acceso al bucket de S3. Si el bucket de S3 no es de su propiedad, agregue `s3:PutObjectAcl` a la lista de acciones de Amazon Simple Storage Service, ya que le concede al propietario del bucket acceso completo a los objetos entregados por Firehose. Esta política también otorga a Firehose acceso CloudWatch para el registro de errores. La política también incluye una declaración que permite el acceso a Amazon Kinesis Data Streams. Si no utiliza Kinesis Data Streams como origen de datos, puede eliminar dicha declaración. Firehose no usa IAM para acceder a Snowflake. Para acceder a Snowflake, utiliza la URL de su cuenta de Snowflake y el ID de PrivateLink Vpce en el caso de un clúster 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:us-east-1:123456789012:key/key-id"
            ],
            "Condition": {
"StringEquals": {
"kms:ViaService": "s3.us-east-1.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:us-east-1:123456789012:stream/stream-
name"
        },
        {
"Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:123456789012:log-group:log-group-name:log-
stream:*"
            ]
        }
    ]
}
```

Para obtener más información sobre cómo permitir que otros AWS servicios accedan a sus AWS recursos, consulte Creación de un rol para delegar permisos a un AWS servicio en la Guía del usuario de IAM.

Acceso a Snowflake en VPC

Si su clúster de Snowflake tiene habilitados los enlaces privados, Firehose utilizará uno de los siguientes puntos de conexión de VPC al crear el enlace privado para entregar datos a su clúster privado sin pasar por la Internet pública. Para ello, cree reglas de red de Snowflake que permitan la entrada desde las siguientes fuentes `AwsVpceIds` para el clúster en el que se encuentra Región de AWS su clúster. Para obtener más información, consulte Creating network rule en la Guía del usuario de Snowflake.

ID de punto de conexión de VPC que se utilizará en función de las regiones en las que se encuentre el clúster

Región de AWS	**`VPCE IDs`**
Este de EE. UU. (Ohio)	vpce-0d96cafcd96a50aeb vpce-0cec34343d48f537b
Este de EE. UU. (Norte de Virginia)	vpce-0b4d7e8478e141ba8 vpce-0b75cd681fb507352 vpce-01c03e63820ec00d8 vpce-0c2cfc51dc2882422 vpce-06ca862f019e4e056 vpce-020cda0cfa63f8d1c vpce-0b80504a1a783cd70 vpce-0289b9ff0b5259a96 vpce-0d7add8628bd69a12 vpce-02bfb5966cc59b2af

Región de AWS	**VPCE IDs**
	vpce-09e707674af878bf2 vpce-049b52e96cc1a2165 vpce-0bb6c7b7a8a86cdbb vpce-03b22d599f51e80f3 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

Región de AWS	**VPCE IDs**
	vpce-00e374d9e3f1af5ce vpce-0c1e3d6631ddb442f
Oeste de EE. UU. (Oregón)	vpce-0f60f72da4cd1e4e7 vpce-0c60d21eb8b1669fd vpce-01c4e3e29afdafbef vpce-0cc6bf2a88da139de vpce-0797e08e169e50662 vpce-033cbe480381b5c0e vpce-00debbdd8f9eb10a5 vpce-08ec2f386c809e889 vpce-0856d14310857b545
Europa (Fráncfort)	vpce-068dbb7d71c9460fb vpce-0a7a7f095942d4ec9
Europa (Irlanda)	vpce-06857e59c005a6276 vpce-04390f4f8778b75f2 vpce-011fd2b1f0aa172fd
Asia-Pacífico (Tokio)	vpce-06369e5258144e68a vpce-0f2363cdb8926fbe8
Asia-Pacífico (Singapur)	vpce-049cd46cce7a12d52 vpce-0e8965a1a4bdb8941

Región de AWS	**VPCE IDs**
Asia-Pacífico (Seúl)	vpce-0aa444d9001e1faa1 vpce-04a49d4dcfd02b884
Asia-Pacífico (Sídney)	vpce-048a60a182c52be63 vpce-03c19949787fd1859
Asia-Pacífico (Mumbai)	vpce-0d68cb822f6f0db68 vpce-0517d32692ffcbde2
Europa (Londres)	vpce-0fd1874a0ba3b9374 vpce-08091b1a85e206029
América del Sur (São Paulo)	vpce-065169b8144e4f12e vpce-0493699f0e5762d63
Canadá (centro)	vpce-07e6ed81689d5271f vpce-0f53239730541394c
Europa (París)	vpce-09419680077e6488a vpce-0ea81ba2c08140c14
Asia-Pacífico (Osaka)	vpce-0a9f003e6a7e38c05 vpce-02886510b897b1c5a
Europa (Estocolmo)	vpce-0d96410833219025a vpce-060a32f9a75ba969f
Asia-Pacífico (Yakarta)	vpce-00add4b9a25e5c649 vpce-004ae2de34338a856

Concesión a Firehose de acceso a un destino de punto de conexión HTTP

Puede usar Amazon Data Firehose para entregar datos a cualquier destino de punto de conexión HTTP. Amazon Data Firehose también crea copias de seguridad de dichos datos en el bucket de Amazon S3 que especifique y le ofrece la posibilidad de utilizar una clave de AWS KMS de su propiedad para el cifrado del servidor de Amazon S3. Si el registro de errores está activado, Amazon Data Firehose envía los errores de entrega de datos a sus flujos de CloudWatch registro. También se puede utilizar AWS Lambda para la transformación de datos.

Es obligatorio contar con un rol de IAM al crear un flujo de Firehose. Amazon Data Firehose asume esa función de IAM y obtiene acceso al bucket, la clave y el grupo de CloudWatch registros y las transmisiones especificados.

Utilice la siguiente política de acceso para permitir a Amazon Data Firehose acceder al bucket de S3 que haya especificado para la copia de seguridad de los datos. Si el bucket de S3 no es de su propiedad, agregue `s3:PutObjectAcl` a la lista de acciones de Amazon S3, ya que le concede al propietario del bucket acceso completo a los objetos entregados por Amazon Data Firehose. Esta política también otorga a Amazon Data Firehose acceso para el registro de errores y CloudWatch AWS Lambda para la transformación de datos. La política también incluye una declaración que permite el acceso a Amazon Kinesis Data Streams. Si no utiliza Kinesis Data Streams como origen de datos, puede eliminar dicha declaración.

> **Important**
>
> Amazon Data Firehose no utiliza la IAM para acceder a destinos de punto final HTTP propiedad de proveedores de servicios externos compatibles, como Datadog, Dynatrace, LogicMonitor MongoDB, New Relic, Splunk o Sumo Logic. Para acceder a un destino de punto de conexión HTTP específico de un proveedor de servicios de terceros admitido, póngase en contacto con dicho proveedor de servicios para obtener la clave de la API o la clave de acceso necesaria para permitir la entrega de datos en ese servicio desde Amazon Data Firehose.

Para obtener más información sobre cómo permitir que otros AWS servicios accedan a sus AWS recursos, consulte Creación de un rol para delegar permisos a un servicio en la Guía del usuario de IAM. AWS

> **Important**
>
> Actualmente, Amazon Data Firehose NO admite la entrega de datos en puntos de conexión HTTP en una VPC.

Entrega entre cuentas desde Amazon MSK

Cuando creas una transmisión de Firehose desde tu cuenta de Firehose (por ejemplo, la cuenta B) y tu fuente es un clúster de MSK en otra AWS cuenta (cuenta A), debes tener implementadas las siguientes configuraciones.

Cuenta A:

1. En la consola de Amazon MSK, elija el clúster aprovisionado y, a continuación, seleccione Propiedades.
2. En Configuración de red, seleccione Editar y active Conectividad con varias VPC.
3. En Configuración de seguridad, seleccione Editar política del clúster.

    a. Si el clúster aún no tiene ninguna política configurada, marque Incluir entidad principal de servicio de Firehose y Habilitar entrega de S3 entre cuentas de Firehose. Consola de administración de AWS Generará automáticamente una política con los permisos adecuados.

    b. Si el clúster ya tiene una política configurada, agregue los siguientes permisos a la política existente:

```
{
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::us-east-1:role/mskaasTestDeliveryRole"
      },
      "Action": [
        "kafka:GetBootstrapBrokers",
        "kafka:DescribeCluster",
        "kafka:DescribeClusterV2",
        "kafka-cluster:Connect"
      ],
      "Resource": "arn:aws:kafka:us-east-1:123456789012:cluster/DO-NOT-TOUCH-
mskaas-provisioned-privateLink/xxxxxxxxx-2f3a-462a-ba09-xxxxxxxxxx-20"  // ARN
 of the cluster
```

```
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::us-east-1: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::us-east-1: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. En Entidad principal de AWS , ingrese el ID de la entidad principal de la cuenta B.
5. En Tema, especifique el tema de Apache Kafka del que desea que su flujo de Firehose ingiera datos. Una vez creado el flujo de Firehose, no podrá actualizar este tema.
6. Elija Guardar cambios.

Cuenta B:

1. En la consola de Firehose, elija Crear flujo de Firehose con la cuenta B.
2. En Origen, elija Amazon Managed Streaming para Apache Kafka.
3. En Configuración de origen, en Clúster de Amazon Managed Streaming para Apache, ingrese el ARN del clúster de Amazon MSK de la cuenta A.
4. En Tema, especifique el tema de Apache Kafka del que desea que su flujo de Firehose ingiera datos. Una vez creado el flujo de Firehose, no podrá actualizar este tema.

5. En Nombre del flujo de entrega, indique el nombre para el flujo de Firehose.

En la cuenta B, cuando cree su transmisión de Firehose, debe tener una función de IAM (que se crea de forma predeterminada al usar la Consola de administración de AWS) que conceda a la transmisión de Firehose acceso de «lectura» al clúster multicuenta de Amazon MSK para el tema configurado.

A continuación se indica lo que configura la Consola de administración de AWS:

```
{
    "Sid": "",
    "Effect": "Allow",
    "Action": [
        "kafka:GetBootstrapBrokers",
        "kafka:DescribeCluster",
        "kafka:DescribeClusterV2",
        "kafka-cluster:Connect"
        ],
    "Resource": "arn:aws:kafka:us-east-1:arn:aws: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:aws: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:aws:group/DO-NOT-TOUCH-mskaas-
provisioned-privateLink/xxxxxxxxx-2f3a-462a-ba09-xxxxxxxxxx-20/*" //topic of the
 cluster
    },
 }
```

Luego, puede completar el paso opcional de configurar la transformación de registros y la conversión del formato de registros. Para obtener más información, consulte (Opcional) Configuración de la transformación de registros y de la conversión de formato.

Entrega entre cuentas en un destino de Amazon S3

Puede usar Amazon Data Firehose AWS CLI o Amazon APIs para crear una transmisión de Firehose en una AWS cuenta con un destino de Amazon S3 en otra cuenta. En el siguiente procedimiento, se muestra un ejemplo de cómo configurar un flujo de Firehose propiedad de la cuenta A para entregar datos en un bucket de Amazon S3 propiedad de la cuenta B.

1. Cree un rol de IAM en la cuenta A siguiendo los pasos descritos en Concesión a Firehose de acceso a un destino de Amazon S3.

   > **Note**
   >
   > En este caso, el bucket de Amazon S3 especificado en la política de acceso es propiedad de la cuenta B. Asegúrese de agregar `s3:PutObjectAcl` a la lista de acciones de Amazon S3 en la política de acceso, ya que concede a la cuenta B acceso completo a los objetos entregados por Amazon Data Firehose. Este permiso es necesario para la entrega entre cuentas. Amazon Data Firehose establece el encabezado "x-amz-acl" de la solicitud en "»bucket-owner-full-control.

2. Para permitir el acceso desde el rol de IAM creado anteriormente, cree una política de bucket de S3 en la cuenta B. El código siguiente es un ejemplo de la política del bucket. Para obtener más información, consulte Uso de políticas de bucket y usuario.

   JSON

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

```
    "Statement": [
        {
            "Sid": "StmtID",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012: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. Cree un flujo de Firehose en la cuenta A con el rol de IAM que creó en el paso 1.

Entrega multicuenta a un OpenSearch destino del servicio

Puedes usar la Firehose AWS CLI o la Amazon Data Firehose APIs para crear una transmisión de Firehose en una AWS cuenta con un destino de OpenSearch servicio en otra cuenta. El siguiente procedimiento muestra un ejemplo de cómo se puede crear una transmisión Firehose en la cuenta A y configurarla para que entregue datos a un destino de OpenSearch servicio propiedad de la cuenta B.

1. Cree un rol de IAM en la cuenta A siguiendo los pasos que se describen en the section called “Conceda a Firehose acceso a un destino de servicio público OpenSearch ”.
2. Para permitir el acceso desde el rol de IAM que creó en el paso anterior, cree una política de OpenSearch servicio en la cuenta B. El siguiente JSON es un ejemplo.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:role/firehose_delivery_role "
      },
      "Action": "es:ESHttpGet",
      "Resource": [
        "arn:aws:es:us-east-1:123456789012:domain/cross-account-cluster/_all/
_settings",
        "arn:aws:es:us-east-1:123456789012:domain/cross-account-cluster/
_cluster/stats",
        "arn:aws:es:us-east-1:123456789012:domain/cross-account-cluster/
roletest*/_mapping/roletest",
        "arn:aws:es:us-east-1:123456789012:domain/cross-account-cluster/
_nodes",
        "arn:aws:es:us-east-1:123456789012:domain/cross-account-cluster/
_nodes/stats",
        "arn:aws:es:us-east-1:123456789012:domain/cross-account-cluster/
_nodes/*/stats",
        "arn:aws:es:us-east-1:123456789012:domain/cross-account-cluster/
_stats",
        "arn:aws:es:us-east-1:123456789012:domain/cross-account-cluster/
roletest*/_stats",
        "arn:aws:es:us-east-1:123456789012:domain/cross-account-cluster/"
      ]
    }
  ]
}
```

3. Cree un flujo de Firehose en la cuenta A con el rol de IAM que creó en el paso 1. Cuando cree la transmisión Firehose, utilice Amazon Data Firehose AWS CLI o Amazon APIs y especifique el `ClusterEndpoint` campo en lugar de Servicio. `DomainARN` OpenSearch

> **Note**
>
> Para crear una transmisión de Firehose en una AWS cuenta con un destino de OpenSearch servicio en otra cuenta, debes usar Amazon AWS CLI Data Firehose. APIs No puedes usar el Consola de administración de AWS para crear este tipo de configuración multicuenta.

Uso de etiquetas para controlar el acceso

Puede utilizar el elemento `Condition` opcional (o el bloque `Condition`) en una política de IAM para afinar el acceso a las operaciones de Amazon Data Firehose en función de las claves y los valores de las etiquetas. En las siguientes subsecciones, se describe cómo hacerlo para las distintas operaciones de Amazon Data Firehose. Para obtener más información sobre el uso del elemento `Condition` y los operadores que puede utilizar dentro de él, consulte Elementos de política JSON de IAM: Condition.

CreateDeliveryStream

Para la operación `CreateDeliveryStream`, utilice la clave de condición `aws:RequestTag`. En el ejemplo siguiente, `MyKey` y `MyValue` representan la clave y el valor correspondiente de una etiqueta. Para obtener más información, consulte Comprensión de los conceptos básicos sobre etiquetas

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "firehose:CreateDeliveryStream",
            "firehose:TagDeliveryStream"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/MyKey": "MyValue"
            }
        }
```

```
    }]
}
```

TagDeliveryStream

Para la operación `TagDeliveryStream`, utilice la clave de condición `aws:TagKeys`. En el ejemplo siguiente, `MyKey` es un ejemplo de clave de etiqueta.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "firehose:TagDeliveryStream",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "MyKey"
                 }
            }
        }
    ]
}
```

UntagDeliveryStream

Para la operación `UntagDeliveryStream`, utilice la clave de condición `aws:TagKeys`. En el ejemplo siguiente, `MyKey` es un ejemplo de clave de etiqueta.

JSON

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

```
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "MyKey"
                 }
            }
        }
    ]
}
```

ListDeliveryStreams

No se puede utilizar el control de acceso basado en etiquetas con `ListDeliveryStreams`.

Otras operaciones

Para las operaciones de Firehose distintas de `CreateDeliveryStream`, `TagDeliveryStream`, `UntagDeliveryStream` y `ListDeliveryStreams`, utilice la clave de condición `aws:RequestTag`. En el ejemplo siguiente, `MyKey` y `MyValue` representan la clave y el valor correspondiente de una etiqueta.

`ListDeliveryStreams`, utilice la clave de condición `firehose:ResourceTag` para controlar el acceso en función de las etiquetas de ese flujo de Firehose.

En el ejemplo siguiente, `MyKey` y `MyValue` representan la clave y el valor correspondiente de una etiqueta. La política solo se aplica a los flujos de Data Firehose que tengan una etiqueta denominada `MyKey` con un valor de `MyValue`. Para obtener más información sobre cómo controlar el acceso en función de las etiquetas de los recursos, consulte Controlar el acceso a AWS los recursos mediante etiquetas en la Guía del usuario de IAM.

Autenticate con AWS Secrets Manager Amazon Data Firehose

Amazon Data Firehose se integra AWS Secrets Manager para proporcionar un acceso seguro a sus datos secretos y automatizar la rotación de credenciales. Esta integración permite a Firehose recuperar un secreto de Secrets Manager en tiempo de ejecución para conectarse a los destinos de streaming mencionados anteriormente y entregar sus flujos de datos. De este modo, sus secretos no están visibles en texto plano durante el flujo de trabajo de creación de transmisiones Consola de administración de AWS ni en los parámetros de la API. Es una práctica segura para administrar sus secretos y evita que tenga que implementar actividades complejas de administración de

credenciales, como la configuración de funciones de Lambda personalizadas para administrar la rotación de contraseñas.

Para obtener más información, consulte la Guía del usuario de AWS Secrets Manager.

Temas

Comprensión de los secretos

Un secreto puede ser una contraseña, un conjunto de credenciales, como un nombre de usuario y una contraseña, un OAuth token u otra información secreta que se almacene de forma cifrada en Secrets Manager.

Para cada destino, debe especificar el par clave-valor secreto en el formato JSON correcto, como se muestra en la siguiente sección. Amazon Data Firehose no podrá conectarse al destino si el secreto no tiene el formato JSON correcto según el destino.

Formato de secreto para bases de datos como MySQL y PostgreSQL

```
{
    "username":  "<username>",
    "password":  "<password>"
}
```

Formato de secreto para el clúster aprovisionado Amazon Redshift y el grupo de trabajo Amazon Redshift sin servidor

```
{
    "username":  "<username>",
    "password":  "<password>"
}
```

Formato de secreto para Splunk

```
{
```

```
    "hec_token":  "<hec token>"
}
```

Formato del secreto de Snowflake

```
{
    "user":  "<snowflake-username>",
    "private_key":  "<snowflake-private-key>", // without the beginning and ending
 private key, remove all spaces and newlines
    "key_passphrase":  "<snowflake-private-key-passphrase>" // optional
}
```

Formato de secreto para el punto final HTTP, Coralogix, Datadog, Dynatrace, Elastic, Honeycomb, LogicMonitor Logz.io, MongoDB Cloud y New Relic

```
{
    "api_key":  "<apikey>"
}
```

Creación de un secreto

Para crear un secreto, siga los pasos que se indican en la sección Crear un AWS Secrets Manager secreto de la Guía del AWS Secrets Manager usuario.

Uso del secreto

Le recomendamos que las utilice AWS Secrets Manager para almacenar sus credenciales o claves para conectarse a destinos de streaming como Amazon Redshift, HTTP Endpoint, Snowflake, Splunk, Coralogix, Datadog, Dynatrace, Elastic, Honeycomb, Logz.io, MongoDB Cloud y New Relic. LogicMonitor

Puede configurar la autenticación con Secrets Manager para estos destinos a través de la Consola de administración de AWS en el momento de crear el flujo de Firehose. Para obtener más información, consulte Configuración de los ajustes de destino. Como alternativa, también puede usar las operaciones CreateDeliveryStreamy UpdateDestinationAPI para configurar la autenticación con Secrets Manager.

Firehose guarda en caché los secretos con un cifrado y los utiliza para cada conexión a destinos. Actualiza la memoria caché cada 10 minutos para asegurarse de que se utilizan las credenciales más recientes.

Puede optar por desactivar la capacidad de recuperar datos secretos de Secrets Manager en cualquier momento durante el ciclo de vida del flujo. Si no quieres usar Secrets Manager para recuperar secretos, puedes usar la clave username/password o API en su lugar.

> **Note**
>
> Aunque esta característica no conlleva ningún costo adicional en Firehose, se le cobrará el acceso y el mantenimiento de Secrets Manager. Para obtener más información, consulte la página de precios de AWS Secrets Manager.

Concesión de acceso a Firehose para recuperar el secreto

Para que Firehose recupere un secreto AWS Secrets Manager, debes proporcionar a Firehose los permisos necesarios para acceder al secreto y la clave que lo cifra.

Cuando se utiliza AWS Secrets Manager para almacenar y recuperar secretos, hay varias opciones de configuración diferentes en función de dónde esté almacenado el secreto y de cómo esté cifrado.

- Si el secreto está almacenado en la misma AWS cuenta que tu función de IAM y está cifrado con la clave AWS gestionada predeterminada (`aws/secretsmanager`), la función de IAM que Firehose asume solo necesita `secretsmanager:GetSecretValue` permiso sobre el secreto.

  ```
  // secret role policy
  {
      "Version": "2012-10-17",
      "Statement": [
          {
              "Effect": "Allow",
              "Action": "secretsmanager:GetSecretValue",
              "Resource": "Secret ARN"
          }
      ]
  }
  ```

  Para obtener más información acerca de las políticas de IAM, consulte Ejemplos de políticas de permisos para AWS Secrets Manager.

- Si el secreto se almacena en la misma cuenta que el rol, pero se cifra con una clave administrada por el cliente (CMK), el rol necesita ambos permisos `secretsmanager:GetSecretValue`

y `kms:Decrypt`. La política de CMK también debe permitir que el rol de IAM implemente `kms:Decrypt`.

- Si el secreto se almacena en una AWS cuenta diferente a la de su rol y se cifra con la clave AWS administrada predeterminada, esta configuración no es posible, ya que Secrets Manager no permite el acceso entre cuentas cuando el secreto está cifrado con la clave AWS administrada.
- Si el secreto se almacena en una cuenta diferente y se cifra con una CMK, el rol de IAM necesita tener permiso `secretsmanager:GetSecretValue` sobre el secreto y `kms:Decrypt` sobre la CMK. La política de recursos del secreto y la política de CMK de la otra cuenta también deben concederle al rol de IAM los permisos necesarios. Para obtener más información, consulte Acceso entre cuentas.

Rotación del secreto

La rotación consiste en actualizar periódicamente un secreto. Puede configurarlo AWS Secrets Manager para que rote automáticamente el secreto según el cronograma que especifique. Así, puede reemplazar los secretos a largo plazo por secretos a corto plazo. Esto ayuda a reducir el riesgo de que se comprometa la seguridad. Para obtener más información, consulte Rotar AWS Secrets Manager los secretos en la Guía del AWS Secrets Manager usuario.

Administración de los roles de IAM a través de la consola de Amazon Data Firehose

Amazon Data Firehose es un servicio totalmente administrado para enviar datos de streaming en tiempo real a diferentes destinos. También puede configurar Firehose para transformar y convertir el formato de los datos antes de entregarlos. Para usar estas funciones, primero debe proporcionar roles de IAM para conceder permisos a Firehose cuando cree o edite un flujo de Firehose. Firehose utiliza este rol de IAM para todos los permisos que necesita el flujo de Firehose.

Por ejemplo, considere un escenario en el que crea una transmisión de Firehose que entrega datos a Amazon S3 y esta transmisión de Firehose tiene los registros de origen de Transform con la función habilitada. AWS Lambda En este caso, debe proporcionar roles de IAM para conceder a Firehose permisos de acceso al bucket de S3 e invocar la función de Lambda, como se muestra a continuación.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Sid": "lambdaProcessing",
        "Effect": "Allow",
        "Action": ["lambda:InvokeFunction", "lambda:GetFunctionConfiguration"],
        "Resource": "arn:aws:lambda:us-east-1:123456789012: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>/
*"]
    }]
}
```

La consola de Firehose permite elegir cómo quiere proporcionar estos roles. Puede elegir una de las siguientes opciones.

- Elección de un rol de IAM existente
- Creación de un nuevo rol de IAM en la consola

Elección de un rol de IAM existente

Puede elegir un rol de IAM existente. Con esta opción, asegúrese de que el rol de IAM que elija tenga una política de confianza adecuada y los permisos necesarios para el origen y el destino. Para obtener más información, consulte Control del acceso con Amazon Data Firehose.

Creación de un nuevo rol de IAM en la consola

Como alternativa, también puede utilizar la consola de Firehose para crear un rol nuevo en su nombre.

Cuando Firehose crea un rol de IAM en su nombre, este incluye automáticamente todas las políticas de permisos y de confianza que otorgan los permisos necesarios en función de la configuración del flujo de Firehose.

Por ejemplo, si no se habilitó la característica Transformar los registros de origen con AWS Lambda, la consola generará la siguiente declaración en la política de permisos.

```
{
  "Sid": "lambdaProcessing",
  "Effect": "Allow",
   "Action": [
     "lambda:InvokeFunction",
     "lambda:GetFunctionConfiguration"
   ],
   "Resource": "arn:aws:lambda:us-east-1123456789012:function:
%FIREHOSE_POLICY_TEMPLATE_PLACEHOLDER%"
}
```

> **Note**
>
> Es seguro ignorar las declaraciones de política que contienen `%FIREHOSE_POLICY_TEMPLATE_PLACEHOLDER%`, ya que no otorgan permisos sobre ningún recurso.

La consola que crea y edita los flujos de trabajo del flujo de Firehose también crea una política de confianza y la adjunta al rol de IAM. Esta política de confianza permite que Firehose asuma el rol de IAM. A continuación, se muestra un ejemplo de una política de confianza.

JSON

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

```
    }]
}
```

Important

- Se debe evitar usar el mismo rol de IAM administrado desde la consola para múltiples flujos de Firehose. De lo contrario, el rol de IAM podría volverse demasiado permisivo o provocar errores.
- Para utilizar diferentes declaraciones de política dentro de una política de permisos de un rol de IAM administrado desde una consola, puede crear su propio rol de IAM y copiar las declaraciones de política en una política de permisos adjunta al nuevo rol. Para adjuntar el rol al flujo de Firehose, seleccione la opción Elegir el rol de IAM existente en el Acceso al servicio.
- La consola administra cualquier rol de IAM que contenga la cadena service-role en su ARN. Al elegir la opción de rol de IAM existente, asegúrese de seleccionar un rol de IAM sin la cadena service-role en su ARN para que la consola no realice ningún cambio en él.

Pasos para crear un rol de IAM desde la consola

1. Abre la consola Firehose en. https://console.aws.amazon.com/firehose/
2. Seleccione Crear flujo de Firehose.
3. Elija un origen y un destino. Para obtener más información, consulte Tutorial: Crear un flujo de Firehose desde la consola.
4. Elija los ajustes del destino. Para obtener más información, consulte Configuración de los ajustes de destino.
5. En Configuración avanzada, para Acceso al servicio, seleccione Crear o actualizar un rol de IAM.

   Note

   Esta es una opción predeterminada. Para usar un rol existente, seleccione la opción Elegir rol de IAM existente. La consola de Firehose no realizará ningún cambio en su rol.

6. Seleccione Crear flujo de Firehose.

Edición del rol de IAM desde la consola

Al editar un flujo de Firehose, Firehose actualiza la política de permisos correspondiente en consecuencia para reflejar los cambios de configuración y permisos.

Por ejemplo, si edita el flujo de Firehose y habilita la característica Transformar registros de origen con AWS Lambda con la última versión de la función de Lambda como `exampleLambdaFunction`, obtendrá la siguiente declaración de política en la política de permisos.

```
{
  "Sid": "lambdaProcessing",
  "Effect": "Allow",
  "Action": [
    "lambda:InvokeFunction",
    "lambda:GetFunctionConfiguration"
  ],
  "Resource": "arn:aws:lambda:us-east-1:123456789012:function:exampleLambdaFunction:
$LATEST"
}
```

Important

Un rol de IAM administrado desde una consola está diseñado para ser autónomo. No se recomienda modificar la política de permisos o la política de confianza fuera de la consola.

Pasos para editar el rol de IAM desde la consola

1. Abre la consola Firehose en. https://console.aws.amazon.com/firehose/
2. Elija los flujos de Firehose y elija el nombre del flujo de Firehose que quiera actualizar.
3. En la pestaña Configuración, en la sección Acceso al servidor, seleccione Editar.
4. Actualice la opción de rol de IAM.

Note

De forma predeterminada, la consola siempre actualiza un rol de IAM con el patrón service-role en su ARN. Al elegir la opción de rol de IAM existente, asegúrese de

seleccionar un rol de IAM sin la cadena de service-role en su ARN para que la consola no realice ningún cambio en él.

5. Seleccione Save changes (Guardar cambios).

Comprensión de la conformidad de Amazon Data Firehose

Los auditores externos evalúan la seguridad y la conformidad de Amazon Data Firehose como parte de varios programas de AWS conformidad. Estos incluyen SOC, PCI, FedRAMP, HIPAA y otros.

Para ver una lista de AWS los servicios incluidos en el ámbito de los programas de conformidad específicos, consulte los AWS servicios incluidos en el ámbito de aplicación por programa de conformidad. Para obtener información general, consulte Programas de conformidad de AWS.

Puede descargar informes de auditoría de terceros utilizando AWS Artifact. Para obtener más información, consulta Descarga de informes en AWS Artifact.

Su responsabilidad con la conformidad al utilizar Data Firehose se determina en función de la confidencialidad de los datos, los objetivos de conformidad de su empresa y la legislación y los reglamentos aplicables. Si su uso de Data Firehose está sujeto al cumplimiento de estándares como HIPAA, PCI o FedRAMP, proporciona recursos para ayudarlo a: AWS

- Guías de inicio rápido sobre seguridad y cumplimiento: estas guías de implementación analizan las consideraciones arquitectónicas y proporcionan los pasos para implementar entornos básicos centrados en la seguridad y el cumplimiento. AWS
- Documento técnico sobre cómo diseñar una arquitectura para la seguridad y el cumplimiento de la HIPAA: este documento técnico describe cómo las empresas pueden utilizar para crear aplicaciones que cumplan con la HIPAA. AWS
- AWS Recursos de cumplimiento: esta colección de libros de trabajo y guías puede aplicarse a su sector y ubicación.
- AWS Config— Este AWS servicio evalúa en qué medida las configuraciones de sus recursos cumplen con las prácticas internas, las directrices del sector y las normativas.
- AWS Security Hub CSPM— Este AWS servicio proporciona una visión integral del estado de su seguridad AWS que le ayuda a comprobar el cumplimiento de los estándares y las mejores prácticas del sector de la seguridad.

Resiliencia en Amazon Data Firehose

La infraestructura AWS global se basa en AWS regiones y zonas de disponibilidad. AWS Las regiones proporcionan varias zonas de disponibilidad aisladas y separadas físicamente, que están conectadas mediante redes de baja latencia, alto rendimiento y alta redundancia. Con las zonas de disponibilidad, puede diseñar y utilizar aplicaciones y bases de datos que realizan una conmutación por error automática entre zonas de disponibilidad sin interrupciones. Las zonas de disponibilidad tienen una mayor disponibilidad, tolerancia a errores y escalabilidad que las infraestructuras tradicionales de centros de datos únicos o múltiples.

Para obtener más información sobre AWS las regiones y las zonas de disponibilidad, consulte Infraestructura global.AWS

Además de la infraestructura AWS global, Data Firehose ofrece varias funciones para ayudar a respaldar sus necesidades de respaldo y resiliencia de datos.

Recuperación ante desastres

Amazon Data Firehose se ejecuta en un modo sin servidor y se ocupa de las reducciones del rendimiento del host, la disponibilidad de las zonas de disponibilidad y otros problemas relacionados con la infraestructura al llevar a cabo una migración automática. Al ocurrir esto, Amazon Data Firehose se asegura de que el flujo de Firehose se migre sin perder datos.

Comprensión de la seguridad de la infraestructura en Amazon Data Firehose

Como servicio gestionado, Amazon Data Firehose está protegido por la seguridad de la red AWS global. Para obtener información sobre los servicios AWS de seguridad y cómo se AWS protege la infraestructura, consulte Seguridad AWS en la nube. Para diseñar su AWS entorno utilizando las mejores prácticas de seguridad de la infraestructura, consulte Protección de infraestructuras en un marco de buena AWS arquitectura basado en el pilar de la seguridad.

Utiliza las llamadas a la API AWS publicadas para acceder a Firehose a través de la red. Los clientes deben admitir lo siguiente:

- Seguridad de la capa de transporte (TLS). Exigimos TLS 1.2 y recomendamos TLS 1.3.

- Conjuntos de cifrado con confidencialidad directa total (PFS) como DHE (Ephemeral Diffie-Hellman) o ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

> **Note**
>
> En el caso de las solicitudes HTTPS de salida, Amazon Data Firehose utiliza una biblioteca HTTP que selecciona automáticamente la versión más reciente del protocolo TLS que se admite en el destino.

Uso de Amazon Data Firehose con AWS PrivateLink

Puede utilizar un punto de enlace de VPC de interfaz (AWS PrivateLink) para acceder a Amazon Data Firehose desde su VPC sin necesitar una puerta de enlace de Internet o una puerta de enlace NAT. Los puntos finales de la interfaz VPC no requieren una puerta de enlace a Internet, un dispositivo NAT, una conexión VPN o una conexión. Direct Connect Los puntos de enlace de la interfaz VPC funcionan con una AWS tecnología que permite la comunicación privada entre AWS servicios mediante una interfaz de red elástica y privada en IPs su Amazon VPC. AWS PrivateLink Para obtener más información, consulte Amazon Virtual Private Cloud.

Uso de puntos finales de VPC de interfaz ()AWS PrivateLink para Firehose

Para comenzar, cree un punto de conexión de VPC de interfaz para que el tráfico de Amazon Data Firehose de sus recursos de Amazon VPC comience a circular a través del punto de conexión de VPC de interfaz. Cuando se crea un punto de conexión, puede adjuntar una política de punto de conexión que controle el acceso a Amazon Data Firehose. Para obtener más información acerca del uso de políticas para controlar el acceso desde un punto de conexión de VPC a Amazon Data Firehose, consulte Control del acceso a los servicios con puntos de conexión de VPC.

El siguiente ejemplo muestra cómo puede configurar una AWS Lambda función en una VPC y crear un punto de enlace de VPC para permitir que la función se comunique de forma segura con el servicio Amazon Data Firehose. En este ejemplo, utiliza una política que permite a la función de Lambda generar una lista de los flujos de Firehose de la región actual, pero no describir ningún flujo de Firehose.

Crear un punto de conexión de VPC

1. Inicie sesión en la consola de Amazon VPC Consola de administración de AWS y ábrala en. https://console.aws.amazon.com/vpc/
2. En el panel de la VPC, elija Endpoints (Puntos de enlace).
3. Seleccione Crear punto de conexión.
4. En la lista de nombres de servicio, elija `com.amazonaws.`*`your_region`*`.kinesis-firehose`.
5. Elija la VPC y una o varias subredes en las que se debe crear el punto de enlace.
6. Elija uno o varios grupos de seguridad para asociar con el punto de enlace.
7. En Policy (Política), elija Custom (Personalizada) y pegue la siguiente 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. Seleccione Crear punto de conexión.

Creación de un rol de IAM que se usará con la función de Lambda

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de la izquierda, elija Roles y, a continuación, seleccione Crear rol.
3. En Seleccionar tipo de entidad de confianza, deje la selección predeterminada Servicio de AWS .
4. En Choose the service that will use this role (Elegir el servicio que usará este rol), elija Lambda.
5. Elija Next: Permissions (Siguiente: permisos).
6. En la lista de políticas, busque y añada las dos políticas denominadas `AWS LambdaVPCAccessExecutionRole` y `AmazonDataFirehoseReadOnlyAccess`.

   > **Important**
   >
   > A continuación se muestra un ejemplo: Es posible que necesite políticas más estrictas para su entorno de producción.

7. Elija Siguiente: etiquetas. Para este ejercicio, no es necesario que añada ninguna etiqueta. Elija Siguiente: Revisar.
8. Ingrese un nombre para el rol y, a continuación, seleccione Crear rol.

Creación de una función de Lambda en la VPC

1. Abra la AWS Lambda consola en. https://console.aws.amazon.com/lambda/
2. Elija Create function (Crear función).
3. Elija Crear desde cero.
4. Introduzca un nombre para la función y establezca Runtime (Tiempo de ejecución) en `Python 3.9` o versiones posteriores.
5. En Permissions (Permisos), expanda Choose or create an execution role (Seleccionar o crear un rol de ejecución).
6. En la lista Execution role (Rol de ejecución), elija Use an existing role (Usar un rol existente).
7. En la lista Existing role (Rol existente), elija el rol que creó antes.
8. Elija Crear función.
9. En Function code (Código de la función), pegue el siguiente código.

```
   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. En Basic settings (Configuración básica), establezca el tiempo de espera en 1 minuto.
11. En Network (Red), elija la VPC en la que creó el punto de enlace y, a continuación, elija las subredes y el grupo de seguridad con los que asoció el punto de enlace cuando lo creó.
12. Cerca de la parte superior de la página, elija Guardar.
13. Seleccione Probar
14. Ingrese un nombre de evento y, a continuación, elija Crear.

15. Elija Test (Probar) de nuevo. Esto hace que la función se ejecute. Cuando aparezca el resultado de la ejecución, expanda Details (Detalles) y compare la salida de registro con el código de la función. Si la ejecución se realiza correctamente, los resultados muestran una lista de flujos de Firehose de la región, así como la siguiente salida:

```
Calling describe_delivery_stream.

AccessDeniedException

Access denied test succeeded.
```

Compatible Regiones de AWS

Los puntos de conexión de VPC de interfaz se admiten actualmente en las siguientes regiones.

- Este de EE. UU. (Ohio)
- Este de EE. UU. (Norte de Virginia)
- Oeste de EE. UU. (Norte de California)
- Oeste de EE. UU. (Oregón)
- Asia-Pacífico (Bombay)
- Asia-Pacífico (Seúl)
- Asia-Pacífico (Singapur)
- Asia-Pacífico (Sídney)
- Asia-Pacífico (Tailandia)
- Asia-Pacífico (Tokio)
- Asia-Pacífico (Hong Kong)
- Canadá (centro)
- Oeste de Canadá (Calgary)
- China (Pekín)
- China (Ningxia)
- Europa (Fráncfort)
- Europa (Irlanda)
- Europa (Londres)
- Europa (París)

- México (centro)
- América del Sur (São Paulo)
- AWS GovCloud (Este de EE. UU.)
- AWS GovCloud (Estados Unidos-Oeste)
- Europa (España)
- Medio Oriente (EAU)
- Asia-Pacífico (Yakarta)
- Asia-Pacífico (Osaka)
- Israel (Tel Aviv)
- Asia-Pacífico (Malasia)

Implementación de prácticas recomendadas de seguridad para Amazon Data Firehose

Amazon Data Firehose proporciona una serie de características de seguridad que debe tener en cuenta a la hora de desarrollar e implementar sus propias políticas de seguridad. Las siguientes prácticas recomendadas son directrices generales y no constituyen una solución de seguridad completa. Puesto que es posible que estas prácticas recomendadas no sean adecuadas o suficientes para el entorno, considérelas como consideraciones útiles en lugar de como normas.

Implementación del acceso a los privilegios mínimos

Cuando concede permisos, debe decidir a quién concede cada permiso y para qué recurso de Amazon Data Firehose se lo concede. Habilite las acciones específicas que desea permitir en dichos recursos. Por lo tanto, debe conceder únicamente los permisos obligatorios para realizar una tarea. La implementación del acceso con privilegios mínimos es esencial a la hora de reducir los riesgos de seguridad y el impacto que podrían causar los errores o los intentos malintencionados.

Uso de roles de IAM

Las aplicaciones de clientes y productores deben tener credenciales válidas para acceder a flujos de Firehose, mientras que su flujo de Firehose debe tener credenciales válidas para acceder a los destinos. No debe almacenar AWS las credenciales directamente en una aplicación cliente o en un bucket de Amazon S3. Estas son las credenciales a largo plazo que no rotan automáticamente y que podrían tener un impacto empresarial significativo si se comprometen.

En su lugar, tiene que utilizar un rol de IAM para administrar credenciales temporales de las aplicaciones de clientes y productores con el fin de acceder a los flujos de Firehose. Al utilizar un rol, no tiene que utilizar credenciales a largo plazo (como un nombre de usuario y una contraseña o claves de acceso) para acceder a otros recursos.

Para obtener más información, consulte los siguientes temas de la guía del usuario de IAM:

- Roles de IAM
- Situaciones habituales con los roles: usuarios, aplicaciones y servicios

Implementación del cifrado en el servidor en recursos dependientes

Los datos en reposo y los datos en tránsito se pueden cifrar en Amazon Data Firehose. Para obtener más información, consulte Protección de datos en Amazon Data Firehose.

Úselo CloudTrail para monitorear las llamadas a la API

Amazon Data Firehose está integrado con AWS CloudTrail un servicio que proporciona un registro de las acciones realizadas por un usuario, un rol o un AWS servicio en Amazon Data Firehose.

Con la información recopilada por CloudTrail, puede determinar la solicitud que se realizó a Amazon Data Firehose, la dirección IP desde la que se realizó la solicitud, quién la hizo, cuándo se realizó y detalles adicionales.

Para obtener más información, consulte the section called “Registro de llamadas a la API en Firehose”.

Supervisión de Amazon Data Firehose

Puede supervisar Amazon Data Firehose con las siguientes características:

Temas

Implementación de las prácticas recomendadas con las alarmas de CloudWatch

Agregue alarmas de CloudWatch para que avisen cuando las siguientes métricas superen el límite de almacenamiento en búfer (15 minutos como máximo).

- `DeliveryToS3.DataFreshness`
- `DeliveryToIceberg.DataFreshness`
- `DeliveryToSplunk.DataFreshness`
- `DeliveryToAmazonOpenSearchService.DataFreshness`
- `DeliveryToAmazonOpenSearchServerless.DataFreshness`
- `DeliveryToHttpEndpoint.DataFreshness`

Además, cree alarmas basadas en las siguientes expresiones matemáticas métricas.

- `IncomingBytes (Sum per 5 Minutes) / 300` se acerca a un porcentaje de `BytesPerSecondLimit`.
- `IncomingRecords (Sum per 5 Minutes) / 300` se acerca a un porcentaje de `RecordsPerSecondLimit`.

- `IncomingPutRequests (Sum per 5 Minutes) / 300` se acerca a un porcentaje de `PutRequestsPerSecondLimit`.

Otra métrica para la que recomendamos una alarma es `ThrottledRecords`.

Para obtener más información sobre cómo solucionar problemas cuando las alarmas están en estado `ALARM`, consulte Errores de solución de problemas.

Supervisión de Amazon Data Firehose con métricas de CloudWatch

> **Important**
>
> Asegúrese de habilitar las alarmas en todas las métricas de CloudWatch que pertenezcan a su destino para identificar los errores a tiempo.

Amazon Data Firehose se integra con las métricas de Amazon CloudWatch para que pueda recopilar, ver y analizar las métricas de CloudWatch para los flujos de Firehose. Por ejemplo, puede supervisar las métricas `IncomingBytes` y `IncomingRecords` para hacer un seguimiento de los datos ingeridos en Amazon Data Firehose procedentes de productores de datos.

Amazon Data Firehose recopila y publica las métricas de CloudWatch cada minuto. Sin embargo, si las ráfagas de datos de entrada se producen solo durante unos segundos, es posible que no se capturen por completo ni sean visibles en las métricas de un minuto. Esto se debe a que las métricas de CloudWatch se agregan desde Amazon Data Firehose en intervalos de un minuto.

Las métricas recopiladas para los flujos de Firehose son gratuitas. Para obtener información acerca de las métricas de agente de Kinesis, consulte Supervisión del estado del agente de Kinesis.

Temas

- Métricas de CloudWatch para el particionamiento dinámico
- Métricas de CloudWatch para entrega de datos
- Métricas de ingesta de datos
- Métricas de CloudWatch de nivel de API
- Métricas de CloudWatch de transformación de datos

- Métricas de descompresión de Registros de CloudWatch
- Métricas de CloudWatch de conversión de formatos
- Métricas de CloudWatch de cifrado del servidor (SSE)
- Dimensiones de Amazon Data Firehose
- Métricas de uso de Amazon Data Firehose

Métricas de CloudWatch para el particionamiento dinámico

Si el particionamiento dinámico está habilitado, el espacio de nombres AWS/Firehose incluye las siguientes métricas.

Métrica	Descripción
`ActivePartitionsLimit`	Número máximo de particiones activas que procesa un flujo de Firehose antes de enviar los datos al bucket de errores. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`PartitionCount`	Número de particiones que se procesan, es decir, el recuento de particiones activas. Este número varía entre 1 y el límite del recuento de particiones de 500 (valor predeterminado). Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`PartitionCountExceeded`	Esta métrica indica si supera el límite del recuento de particiones. Emite 1 o 0 en función de si se supera el límite o no.
`JQProcessing.Duration`	Devuelve el tiempo que se tardó en ejecutar la expresión JQ en la función de Lambda JQ.

Métrica	Descripción
	Unidades: milisegundos
`PerPartitionThroughput`	Indica el rendimiento que se procesa por partición. Esta métrica le permite supervisar el rendimiento por partición. Unidades: StandardUnit.BytesSecond
`DeliveryToS3.ObjectCount`	Indica la cantidad de objetos que se van a entregar en su bucket de S3. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento

Métricas de CloudWatch para entrega de datos

El espacio de nombres `AWS/Firehose` incluye las siguientes métricas de nivel de servicio. Si observa pequeñas caídas en el promedio de `BackupToS3.Success`, `DeliveryToS3.Success`, `DeliveryToSplunk.Success`, `DeliveryToAmazonOpenSearchService.Success` o `DeliveryToRedshift.Success`, eso no indica que se estén perdiendo datos. Amazon Data Firehose vuelve a intentar los errores de entrega y no avanza hasta que los registros se entreguen correctamente en el destino configurado o el bucket de S3 de copias de seguridad.

Temas

- Entrega en OpenSearch Service
- Entrega en OpenSearch sin servidor
- Entrega en Amazon Redshift
- Entrega en Amazon S3
- Entrega a Snowflake
- Entrega a Splunk
- Entrega en puntos de conexión HTTP

Entrega en OpenSearch Service

Métrica	Descripción
`DeliveryToAmazonOpenSearchService.Bytes`	Número de bytes indexados en OpenSearch Service durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
`DeliveryToAmazonOpenSearchService.DataFreshness`	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se han entregado en OpenSearch Service. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: segundos
`DeliveryToAmazonOpenSearchService.Records`	Número de registros indexados en OpenSearch Service durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`DeliveryToAmazonOpenSearchService.Success`	La suma de los registros indexados correctamente.
`DeliveryToS3.Bytes`	Número de bytes entregados en Amazon S3 durante el periodo de tiempo especificado. Amazon Data Firehose emite esta métrica solo cuando se habilita la copia de seguridad de todos los documentos. Estadísticas: Minimum, Maximum, Average, Sum, Samples

Métrica	Descripción
	Unidades: recuento
DeliveryToS3.DataFreshness	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se han enviado al bucket de S3. Amazon Data Firehose emite esta métrica solo cuando se habilita la copia de seguridad de todos los documentos. Unidades: segundos
DeliveryToS3.Records	Número de registros entregados en Amazon S3 durante el periodo de tiempo especificado. Amazon Data Firehose emite esta métrica solo cuando se habilita la copia de seguridad de todos los documentos. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
DeliveryToS3.Success	Suma de comandos put de Amazon S3 ejecutados correctamente. Amazon Data Firehose siempre emite esta métrica, independientemente de si la copia de seguridad está habilitada solo para los documentos con errores o para todos los documentos.
DeliveryToAmazonOpenSearchService.AuthFailure	Error de autenticación o autorización. Compruebe la política del clúster de OS/ES y los permisos del rol. Un 0 indica que no hay ningún problema y un 1 indica un error de autenticación.
DeliveryToAmazonOpenSearchService.DeliveryRejected	Error de entrega rechazada. Compruebe la política del clúster de OS/ES y los permisos del rol. Un 0 indica que no hay ningún problema y un 1 indica que se ha producido un error en la entrega.

Entrega en OpenSearch sin servidor

Métrica	Descripción
`DeliveryToAmazonOpenSearchServerless.Bytes`	Número de bytes indexados en OpenSearch sin servidor durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
`DeliveryToAmazonOpenSearchServerless.DataFreshness`	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se han entregado en OpenSearch sin servidor. Unidades: segundos
`DeliveryToAmazonOpenSearchServerless.Records`	Número de registros indexados en OpenSearch sin servidor durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`DeliveryToAmazonOpenSearchServerless.Success`	La suma de los registros indexados correctamente.
`DeliveryToS3.Bytes`	Número de bytes entregados en Amazon S3 durante el periodo de tiempo especificado. Amazon Data Firehose emite esta métrica solo cuando se habilita la copia de seguridad de todos los documentos. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento

Métrica	Descripción
DeliveryToS3.DataFreshness	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se han enviado al bucket de S3. Amazon Data Firehose emite esta métrica solo cuando se habilita la copia de seguridad de todos los documentos. Unidades: segundos
DeliveryToS3.Records	Número de registros entregados en Amazon S3 durante el periodo de tiempo especificado. Amazon Data Firehose emite esta métrica solo cuando se habilita la copia de seguridad de todos los documentos. Unidades: recuento
DeliveryToS3.Success	Suma de comandos put de Amazon S3 ejecutados correctamente. Amazon Data Firehose siempre emite esta métrica, independientemente de si la copia de seguridad está habilitada solo para los documentos con errores o para todos los documentos.
DeliveryToAmazonOpenSearchServerless.AuthFailure	Error de autenticación o autorización. Compruebe la política del clúster de OS/ES y los permisos del rol. Un 0 indica que no hay ningún problema y un 1 indica que se ha producido un error de autenticación.
DeliveryToAmazonOpenSearchServerless.DeliveryRejected	Error de entrega rechazada. Compruebe la política del clúster de OS/ES y los permisos del rol. Un 0 indica que no hay ningún problema y un 1 indica que se ha producido un error en la entrega.

Entrega en Amazon Redshift

Métrica	Descripción
DeliveryToRedshift.Bytes	Número de bytes copiados en Amazon Redshift durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
DeliveryToRedshift.Records	Número de registros copiados en Amazon Redshift durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
DeliveryToRedshift.Success	Suma de comandos COPY de Amazon Redshift ejecutados correctamente.
DeliveryToS3.Bytes	Número de bytes entregados en Amazon S3 durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
DeliveryToS3.DataFreshness	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se entregan en el bucket de S3. Unidades: segundos
DeliveryToS3.Records	Número de registros entregados en Amazon S3 durante el periodo de tiempo especificado.

Métrica	Descripción
	Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`DeliveryToS3.Success`	Suma de comandos put de Amazon S3 ejecutados correctamente.
`DeliveryToRedshift.DataFreshness`	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se entregan en el clúster de Amazon Redshift.
`BackupToS3.Bytes`	Número de bytes entregados en Amazon S3 para copias de seguridad durante el periodo de tiempo especificado. Amazon Data Firehose emite esta métrica cuando se habilita la copia de seguridad en Amazon S3. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`BackupToS3.DataFreshness`	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se han entregado en el bucket de Amazon S3 con fines de copia de seguridad. Amazon Data Firehose emite esta métrica cuando se habilita la copia de seguridad en Amazon S3. Estadísticas: Minimum, Maximum, Average, Samples Unidades: segundos

Métrica	Descripción
`BackupToS3.Records`	Número de registros entregados en Amazon S3 para copias de seguridad durante el periodo de tiempo especificado. Amazon Data Firehose emite esta métrica cuando se habilita la copia de seguridad en Amazon S3. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`BackupToS3.Success`	Suma de comandos put de Amazon S3 ejecutados correctamente para la copia de seguridad. Amazon Data Firehose emite esta métrica cuando se habilita la copia de seguridad en Amazon S3.

Entrega en Amazon S3

Las métricas de la tabla siguiente están relacionadas con la entrega en Amazon S3 cuando es el destino principal del flujo de Firehose.

Métrica	Descripción
`DeliveryToS3.Bytes`	Número de bytes entregados en Amazon S3 durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
`DeliveryToS3.DataFreshness`	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se han enviado al bucket de S3. Estadísticas: Minimum, Maximum, Average, Samples

Métrica	Descripción
	Unidades: segundos
DeliveryToS3.Records	Número de registros entregados en Amazon S3 durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
DeliveryToS3.Success	Suma de comandos put de Amazon S3 ejecutados correctamente.
BackupToS3.Bytes	Número de bytes entregados en Amazon S3 para copias de seguridad durante el periodo de tiempo especificado. Amazon Data Firehose emite esta métrica cuando la copia de seguridad está habilitada (lo que solo es posible cuando la transformación de datos también está habilitad a). Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
BackupToS3.DataFreshness	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se han entregado en el bucket de Amazon S3 con fines de copia de seguridad. Amazon Data Firehose emite esta métrica cuando la copia de seguridad está habilitada (lo que solo es posible cuando la transform ación de datos también está habilitada). Estadísticas: Minimum, Maximum, Average, Samples Unidades: segundos

Métrica	Descripción
BackupToS3.Records	Número de registros entregados en Amazon S3 para copias de seguridad durante el periodo de tiempo especificado. Amazon Data Firehose emite esta métrica cuando la copia de seguridad está habilitada (lo que solo es posible cuando la transformación de datos también está habilitada). Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
BackupToS3.Success	Suma de comandos put de Amazon S3 ejecutados correctamente para la copia de seguridad. Amazon Data Firehose emite esta métrica cuando la copia de seguridad está habilitada (lo que solo es posible cuando la transformación de datos también está habilitada).

Entrega a Snowflake

Métrica	Descripción
DeliveryToSnowflake.Bytes	El número de bytes enviados a Snowflake durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
DeliveryToSnowflake.DataFreshness	Antigüedad (desde que se introdujo en Firehose hasta la fecha actual) del registro más antiguo de Firehose. Los registros anteriores a este valor que se han enviado a Snowflake. Tenga en cuenta que el envío de datos en Snowflake puede tardar unos segundos después de que la llamada de inserción de Firehose se haya realizado

Métrica	Descripción
	correctamente. Para ver el tiempo que se tarda en enviar los datos a Snowflake, consulte la métrica `DeliveryToSnowflake.DataCommitLatency` . Estadísticas: Minimum, Maximum, Average, Samples Unidades: segundos
`DeliveryToSnowflake.DataCommitLatency`	Tiempo que tardan los datos en enviarse a Snowflake después de que Firehose haya insertado los registros correctamente. Estadísticas: Minimum, Maximum, Average, Samples Unidades: segundos
`DeliveryToSnowflake.Records`	El número de registros enviados a Snowflake durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`DeliveryToSnowflake.Success`	La suma de las llamadas de inserción realizadas correctamente a Snowflake.
`DeliveryToS3.Bytes`	Número de bytes entregados en Amazon S3 durante el periodo de tiempo especificado. Esta métrica solo está disponible cuando se produce un error en la entrega a Snowflake, y Firehose intenta hacer copias de seguridad de los datos fallidos en S3. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes

Métrica	Descripción
DeliveryToS3.Records	Número de registros entregados en Amazon S3 durante el periodo de tiempo especificado. Esta métrica solo está disponible cuando se produce un error en la entrega a Snowflake, y Firehose intenta hacer copias de seguridad de los datos fallidos en S3. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
DeliveryToS3.Success	Suma de comandos put de Amazon S3 ejecutado s correctamente. Esta métrica solo está disponible cuando se produce un error en la entrega a Snowflake, y Firehose intenta hacer copias de seguridad de los datos fallidos en S3.
BackupToS3.DataFreshness	Antigüedad (desde que se introdujo en Firehose hasta la fecha actual) del registro más antiguo de Firehose. Los registros anteriores a este valor tienen copia de seguridad en el bucket de Amazon S3. Esta métrica está disponible cuando el flujo de Firehose está configurado para hacer copias de seguridad de todos los datos. Estadísticas: Minimum, Maximum, Average, Samples Unidades: segundos

Métrica	Descripción
BackupToS3.Records	Número de registros entregados en Amazon S3 para copias de seguridad durante el periodo de tiempo especificado. Esta métrica está disponible cuando el flujo de Firehose está configurado para hacer copias de seguridad de todos los datos. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
BackupToS3.Bytes	Número de bytes entregados en Amazon S3 para copias de seguridad durante el periodo de tiempo especificado. Esta métrica está disponible cuando el flujo de Firehose está configurado para hacer copias de seguridad de todos los datos. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
BackupToS3.Success	Suma de comandos put de Amazon S3 ejecutados correctamente para la copia de seguridad. Firehose emite esta métrica cuando el flujo de Firehose está configurado para hacer copias de seguridad de todos los documentos.

Entrega a Splunk

Métrica	Descripción
DeliveryToSplunk.Bytes	El número de bytes enviados a Splunk durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples

Métrica	Descripción
	Unidades: bytes
DeliveryToSplunk.DataAckLatency	Duración aproximada que se tarda en recibir la confirmación de Splunk una vez que Amazon Data Firehose envía los datos. La tendencia creciente o decreciente de esta métrica es más útil que el valor aproximado absoluto. Las tendencias crecientes pueden indicar velocidades de indexación y de reconocimiento más lentas de los indexadores de Splunk. Estadísticas: Minimum, Maximum, Average, Samples Unidades: segundos
DeliveryToSplunk.DataFreshness	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se han enviado a Splunk. Estadísticas: Minimum, Maximum, Average, Samples Unidades: segundos
DeliveryToSplunk.Records	El número de registros enviados a Splunk durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
DeliveryToSplunk.Success	La suma de los registros indexados correctamente.
DeliveryToS3.Success	Suma de comandos put de Amazon S3 ejecutados correctamente. Esta métrica se emite cuando la copia de seguridad en Amazon S3 está habilitada.

Métrica	Descripción
BackupToS3.Bytes	Número de bytes entregados en Amazon S3 para copias de seguridad durante el periodo de tiempo especificado. Amazon Data Firehose emite esta métrica cuando el flujo de Firehose está configurado para hacer copias de seguridad de todos los documentos. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
BackupToS3.DataFreshness	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se han entregado en el bucket de Amazon S3 con fines de copia de seguridad. Amazon Data Firehose emite esta métrica cuando el flujo de Firehose está configurado para hacer copias de seguridad de todos los documentos. Estadísticas: Minimum, Maximum, Average, Samples Unidades: segundos
BackupToS3.Records	Número de registros entregados en Amazon S3 para copias de seguridad durante el periodo de tiempo especificado. Amazon Data Firehose emite esta métrica cuando el flujo de Firehose está configurado para hacer copias de seguridad de todos los documentos. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento

Métrica	Descripción
`BackupToS3.Success`	Suma de comandos put de Amazon S3 ejecutados correctamente para la copia de seguridad. Amazon Data Firehose emite esta métrica cuando el flujo de Firehose está configurado para hacer copias de seguridad de todos los documentos.

Entrega en puntos de conexión HTTP

Métrica	Descripción
`DeliveryToHttpEndpoint.Bytes`	Número de bytes entregados correctamente en el punto de conexión HTTP. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
`DeliveryToHttpEndpoint.Records`	Número de registros entregados correctamente en el punto de conexión HTTP. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuentos.
`DeliveryToHttpEndpoint.DataFreshness`	Antigüedad del registro más antiguo de Amazon Data Firehose. Estadísticas: Minimum, Maximum, Average, Samples Unidades: segundos
`DeliveryToHttpEndpoint.Success`	Suma de todas las solicitudes de entrega de datos realizadas correctamente al punto de conexión HTTP.

Métrica	Descripción
	Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`DeliveryToHttpEndpoint.ProcessedBytes`	Número de bytes procesados intentados, incluidos los reintentos.
`DeliveryToHttpEndpoint.ProcessedRecords`	Número de registros intentados, incluidos los reintentos.

Métricas de ingesta de datos

Temas

- Ingesta de datos a través de Kinesis Data Streams
- Ingesta de datos a través de Direct PUT
- Ingesta de datos de MSK

Ingesta de datos a través de Kinesis Data Streams

Métrica	Descripción
`DataReadFromKinesisStream.Bytes`	Cuando el origen de datos es un flujo de datos de Kinesis, esta métrica indica el número de bytes leídos de dicho flujo. Este número incluye las repeticiones de lecturas debido a conmutaciones por error. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
`DataReadFromKinesisStream.Records`	Cuando el origen de datos es un flujo de datos de Kinesis, esta métrica indica el número de registros leídos

Métrica	Descripción
	de dicho flujo. Este número incluye las repeticiones de lecturas debido a conmutaciones por error. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`ThrottledDescribeStream`	El número total de veces que se limita la operación `DescribeStream` cuando el origen de datos es una secuencia de datos de Kinesis. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`ThrottledGetRecords`	El número total de veces que se limita la operación `GetRecords` cuando el origen de datos es una secuencia de datos de Kinesis. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`ThrottledGetShardIterator`	El número total de veces que se limita la operación `GetShardIterator` cuando el origen de datos es una secuencia de datos de Kinesis. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento

Métrica	Descripción
KinesisMillisBehindLatest	Cuando el origen de datos es una secuencia de datos de Kinesis, esta métrica indica el número de milisegundos de retraso que lleva el último registro leído con respecto al registro más reciente de la secuencia de datos de Kinesis. Estadísticas: Minimum, Maximum, Average, Samples Unidades: milisegundos

Ingesta de datos a través de Direct PUT

Métrica	Descripción
BackupToS3.Bytes	Número de bytes entregados en Amazon S3 para copias de seguridad durante el periodo de tiempo especificado. Amazon Data Firehose emite esta métrica cuando la transformación de datos está habilitada para los destinos de Amazon S3 o Amazon Redshift. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
BackupToS3.DataFreshness	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se han entregado en el bucket de Amazon S3 con fines de copia de seguridad. Amazon Data Firehose emite esta métrica cuando la transformación de datos está habilitada para los destinos de Amazon S3 o Amazon Redshift. Estadísticas: Minimum, Maximum, Average, Samples

Métrica	Descripción
	Unidades: segundos
`BackupToS3.Records`	Número de registros entregados en Amazon S3 para copias de seguridad durante el periodo de tiempo especificado. Amazon Data Firehose emite esta métrica cuando la transformación de datos está habilitada para los destinos de Amazon S3 o Amazon Redshift. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`BackupToS3.Success`	Suma de comandos put de Amazon S3 ejecutados correctamente para la copia de seguridad. Amazon Data Firehose emite esta métrica cuando la transformación de datos está habilitada para los destinos de Amazon S3 o Amazon Redshift.
`BytesPerSecondLimit`	Número máximo actual de bytes por segundo que un flujo de Firehose puede ingerir antes de la limitación. Para solicitar un aumento de este límite, vaya a AWS Support Center y elija Crear caso y, a continuación, seleccione Aumento del límite de servicio.
`DeliveryToAmazonOpenSearchService.Bytes`	Número de bytes indexados en OpenSearch Service durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes

Métrica	Descripción
DeliveryToAmazonOpenSearchService.DataFreshness	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se han entregado en OpenSearch Service. Estadísticas: Minimum, Maximum, Average, Samples Unidades: segundos
DeliveryToAmazonOpenSearchService.Records	Número de registros indexados en OpenSearch Service durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
DeliveryToAmazonOpenSearchService.Success	La suma de los registros indexados correctamente.
DeliveryToRedshift.Bytes	Número de bytes copiados en Amazon Redshift durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
DeliveryToRedshift.Records	Número de registros copiados en Amazon Redshift durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
DeliveryToRedshift.Success	Suma de comandos COPY de Amazon Redshift ejecutados correctamente.

Métrica	Descripción
DeliveryToS3.Bytes	Número de bytes entregados en Amazon S3 durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
DeliveryToS3.DataFreshness	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se han enviado al bucket de S3. Estadísticas: Minimum, Maximum, Average, Samples Unidades: segundos
DeliveryToS3.Records	Número de registros entregados en Amazon S3 durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
DeliveryToS3.Success	Suma de comandos put de Amazon S3 ejecutados correctamente.
DeliveryToSplunk.Bytes	El número de bytes enviados a Splunk durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes

Métrica	Descripción
DeliveryToSplunk.DataAckLatency	Duración aproximada que se tarda en recibir la confirmación de Splunk una vez que Amazon Data Firehose envía los datos. La tendencia creciente o decreciente de esta métrica es más útil que el valor aproximado absoluto. Las tendencias crecientes pueden indicar velocidades de indexación y de reconocimiento más lentas de los indexadores de Splunk. Estadísticas: Minimum, Maximum, Average, Samples Unidades: segundos
DeliveryToSplunk.DataFreshness	Antigüedad (desde que se incorporó a Amazon Data Firehose hasta ahora) del registro más antiguo de Amazon Data Firehose. Los registros anteriores a este valor se han enviado a Splunk. Estadísticas: Minimum, Maximum, Average, Samples Unidades: segundos
DeliveryToSplunk.Records	El número de registros enviados a Splunk durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
DeliveryToSplunk.Success	La suma de los registros indexados correctamente.

Métrica	Descripción
`IncomingBytes`	El número de bytes ingeridos correctamente en el flujo de Firehose durante el periodo de tiempo especificado. La ingesta de datos se puede limitar si se supera uno de los límites del flujo de Firehose. Los datos limitados no se tendrán en cuenta para `IncomingBytes` . Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
`IncomingPutRequests`	Número de solicitudes PutRecord y PutRecordBatch correctas durante un periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`IncomingRecords`	El número de registros ingeridos correctamente en el flujo de Firehose durante el periodo de tiempo especificado. La ingesta de datos se puede limitar si se supera uno de los límites del flujo de Firehose. Los datos limitados no se tendrán en cuenta para `IncomingRecords` . Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`RecordsPerSecondLimit`	Número máximo actual de registros por segundo que un flujo de Firehose puede ingerir antes de la limitación. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento

Métrica	Descripción
`ThrottledRecords`	Número de registros que se limitaron porque la ingesta de datos superó uno de los límites del flujo de Firehose. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento

Ingesta de datos de MSK

Métrica	Descripción
`DataReadFromSource.Records`	Número de registros leídos del tema de Kafka de origen. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`DataReadFromSource.Bytes`	Número de bytes leídos del tema de Kafka de origen. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
`SourceThrottled.Delay`	Tiempo que tarda el clúster de Kafka de origen en devolver los registros del tema de Kafka de origen. Estadísticas: Minimum, Maximum, Average, Samples Unidades: milisegundos
`BytesPerSecondLimit`	Límite actual de rendimiento al que Firehose va a leer desde cada partición del tema de Kafka de origen. Estadísticas: Minimum, Maximum, Average, Sum, Samples

Métrica	Descripción
	Unidades: bytes/segundo
KafkaOffsetLag	Diferencia entre el mayor desplazamiento del registro que Firehose ha leído del tema de Kafka de origen y el mayor desplazamiento del registro disponible del tema de Kafka de origen. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
FailedValidation.Records	Número de registros que no han superado la validación de registros. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
FailedValidation.Bytes	Número de bytes que no han superado la validación de registros. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
DataReadFromSource .Backpressured	Indica que un flujo de Firehose se retrasa en la lectura de los registros de la partición de origen, ya sea porque se ha superado el valor de BytesPerSecondLimit por partición o porque el flujo normal de entrega es lento o se ha detenido. Unidades: booleano

Métricas de CloudWatch de nivel de API

El espacio de nombres `AWS/Firehose` incluye las siguientes métricas de nivel de API.

Métrica	Descripción
`DescribeDeliveryStream.Latency`	El tiempo que tarda cada operación `DescribeDeliveryStream` , medido durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Samples Unidades: milisegundos
`DescribeDeliveryStream.Requests`	El número total de solicitudes `DescribeDeliveryStream` . Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`ListDeliveryStreams.Latency`	El tiempo que tarda cada operación `ListDeliveryStream` , medido durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Samples Unidades: milisegundos
`ListDeliveryStreams.Requests`	El número total de solicitudes `ListFirehose` . Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`PutRecord.Bytes`	Número de bytes insertados en el flujo de Firehose mediante `PutRecord` en el periodo de tiempo especificado.

Métrica	Descripción
	Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
PutRecord.Latency	El tiempo que tarda cada operación PutRecord , medido durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Samples Unidades: milisegundos
PutRecord.Requests	El número total de solicitudes PutRecord , que es igual al número total de registros de las operaciones PutRecord . Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
PutRecordBatch.Bytes	Número de bytes insertados en el flujo de Firehose mediante PutRecordBatch en el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
PutRecordBatch.Latency	El tiempo que tarda cada operación PutRecordBatch , medido durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Samples Unidades: milisegundos

Métrica	Descripción
PutRecordBatch.Records	El número total de registros de las operaciones PutRecordBatch . Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
PutRecordBatch.Requests	El número total de solicitudes PutRecordBatch . Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
PutRequestsPerSecondLimit	Número máximo de solicitudes de put por segundo que un flujo de Firehose puede gestionar antes de la limitación. Este número incluye solicitudes PutRecord y PutRecordBatch. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
ThrottledDescribeStream	El número total de veces que se limita la operación DescribeStream cuando el origen de datos es una secuencia de datos de Kinesis. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento

Métrica	Descripción
ThrottledGetRecords	El número total de veces que se limita la operación GetRecords cuando el origen de datos es una secuencia de datos de Kinesis. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
ThrottledGetShardIterator	El número total de veces que se limita la operación GetShardIterator cuando el origen de datos es una secuencia de datos de Kinesis. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
UpdateDeliveryStream.Latency	El tiempo que tarda cada operación UpdateDeliveryStream , medido durante el periodo de tiempo especificado. Estadísticas: Minimum, Maximum, Average, Samples Unidades: milisegundos
UpdateDeliveryStream.Requests	El número total de solicitudes UpdateDeliveryStream . Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento

Métricas de CloudWatch de transformación de datos

Si la transformación de datos con Lambda está habilitada, el espacio de nombres `AWS/Firehose` incluye las siguientes métricas.

Métrica	Descripción
`ExecuteProcessing.Duration`	Tiempo que tarda cada invocación de una función de Lambda llevada a cabo por Firehose. Unidades: milisegundos
`ExecuteProcessing.Success`	Suma de las invocaciones de funciones de Lambda correctas con respecto a la suma del total de invocaciones de funciones de Lambda.
`SucceedProcessing.Records`	Número de registros procesados correctamente durante el periodo de tiempo especificado. Unidades: recuento
`SucceedProcessing.Bytes`	Número de bytes procesados correctamente durante el periodo de tiempo especificado. Unidades: bytes

Métricas de descompresión de Registros de CloudWatch

Si la descompresión está habilitada para la entrega de Registros de CloudWatch, el espacio de nombres de `AWS/Firehose` incluye las siguientes métricas.

Métrica	Descripción
`OutputDecompressedBytes.Success`	Se descomprimieron correctamente los datos en bytes Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes

Métrica	Descripción
`OutputDecompressedBytes.Failed`	Error al descomprimir los datos en bytes Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: bytes
`OutputDecompressedRecords.Success`	Número de registros descomprimidos correctamente Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`OutputDecompressedRecords.Failed`	Número de registros descomprimidos con errores Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento

Métricas de CloudWatch de conversión de formatos

Si la conversión del formato está habilitada, el espacio de nombres `AWS/Firehose` incluye las siguientes métricas.

Métrica	Descripción
`SucceedConversion.Records`	El número de registros convertidos correctamente. Unidades: recuento
`SucceedConversion.Bytes`	El tamaño de los registros convertidos correctamente. Unidades: bytes

Métrica	Descripción
`FailedConversion.Records`	El número de registros que no se han podido convertir. Unidades: recuento
`FailedConversion.Bytes`	El tamaño de los registros que no se han podido convertir. Unidades: bytes

Métricas de CloudWatch de cifrado del servidor (SSE)

El espacio de nombres `AWS/Firehose` incluye las siguientes métricas relacionadas con SSE.

Métrica	Descripción
`KMSKeyAccessDenied`	Número de veces que el servicio encuentra una excepción `KMSAccessDeniedException` del flujo de Firehose. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`KMSKeyDisabled`	Número de veces que el servicio encuentra una excepción `KMSDisabledException` del flujo de Firehose. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
`KMSKeyInvalidState`	Número de veces que el servicio encuentra una excepción `KMSInvalidStateException` del flujo de Firehose.

Métrica	Descripción
	Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento
KMSKeyNotFound	Número de veces que el servicio encuentra una excepción KMSNotFoundException del flujo de Firehose. Estadísticas: Minimum, Maximum, Average, Sum, Samples Unidades: recuento

Dimensiones de Amazon Data Firehose

Para filtrar las métricas por flujo de Firehose, utilice la dimensión DeliveryStreamName.

Métricas de uso de Amazon Data Firehose

Puede utilizar las métricas de uso de CloudWatch para proporcionar visibilidad sobre el uso de los recursos de su cuenta. Utilice estas métricas para visualizar el uso actual del servicio en paneles y gráficos de CloudWatch.

Las métricas de uso de cuotas de servicio se encuentran en el espacio de nombres AWS/Usage y se recopilan cada tres minutos.

Actualmente, el único nombre de métrica de este espacio de nombres que CloudWatch publica es ResourceCount. Esta métrica se publica con las dimensiones Service, Class, Type y Resource.

Métrica	Descripción
ResourceCount	El número de los recursos especificados que se ejecutan en su cuenta. Los recursos se definen por las dimension es asociadas a la métrica.

Métrica	Descripción
	La estadística más útil para esta métrica es MAXIMUM, que representa el número máximo de recursos utilizados durante el periodo de 3 minutos.

Las siguientes dimensiones se utilizan para ajustar las métricas de uso publicadas por Amazon Data Firehose.

Dimensión	Descripción
`Service`	El nombre del servicio de AWS que contiene el recurso. En el caso de las métricas de uso de Amazon Data Firehose, el valor de esta dimensión es `Firehose`.
`Class`	La clase de recurso a la que se realiza el seguimiento. Las métricas de uso de la API de Amazon Data Firehose utilizan esta dimensión con el valor `None`.
`Type`	El tipo de recurso al que se realiza el seguimiento. Actualmente, cuando la dimensión Service es `Firehose`, el único valor válido para Type es `Resource`.
`Resource`	Nombre del recurso de AWS. Actualmente, cuando la dimensión Service es `Firehose`, el único valor válido para Resource es `DeliveryStreams` .

Acceso a las métricas de CloudWatch para Amazon Data Firehose

Puede supervisar las métricas de Amazon Data Firehose con la consola de CloudWatch, la línea de comandos o la API de CloudWatch. Los siguientes procedimientos le muestran cómo obtener acceso a las métricas a través de los distintos métodos descritos a continuación.

Acceso a las métricas a través de la consola de CloudWatch

1. Abra la consola de CloudWatch en https://console.aws.amazon.com/cloudwatch/.
2. Seleccione una región en la barra de navegación.

3. En el panel de navegación, seleccione Métricas.
4. Elija el espacio de nombres de Firehose.
5. Seleccione Métricas del flujo de Firehose o Métricas de Firehose.
6. Seleccione una métrica para añadirla al gráfico.

Acceso a las métricas mediante la AWS CLI

Utilice los comandos list-metrics y get-metric-statistics.

```
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
```

Supervisión de Amazon Data Firehose mediante Registros de CloudWatch

Amazon Data Firehose se integra con Registros de Amazon CloudWatch para permitirle consultar los registros de los errores específicos que puedan presentarse al invocar Lambda para transformar o entregar los datos. Puede habilitar el registro de errores de Amazon Data Firehose al crear el flujo de Firehose.

Si habilita el registro de errores de Amazon Data Firehose en la consola de Amazon Data Firehose, se crean un grupo de registros y los flujos de registros correspondientes para el flujo de Firehose en su nombre. El formato del nombre del grupo de registros es `/aws/kinesisfirehose/`*`delivery-stream-name`*, donde *`delivery-stream-name`* es el nombre del flujo de Firehose correspondiente. `DestinationDelivery` es un flujo de registro que se crea y se utiliza para registrar cualquier error relacionado con la entrega en el destino principal. Otro flujo de registro denominado `BackupDelivery` se crea solo si la copia de seguridad de S3 está habilitada para el destino. El flujo de registro `BackupDelivery` se utiliza para registrar cualquier error relacionado con la entrega en la copia de seguridad de S3.

Por ejemplo, si crea un flujo de Firehose “MyStream” con Amazon Redshift como destino y habilita el registro de errores de Amazon Data Firehose, se crean los siguientes elementos en su

nombre: un grupo de registros denominado `aws/kinesisfirehose/MyStream` y dos flujos de registros denominados `DestinationDelivery` y BackupDelivery. En este ejemplo, se utilizará `DestinationDelivery` para registrar cualquier error relacionado con la entrega en el destino de Amazon Redshift y también en el destino intermedio de S3. `BackupDelivery`, en caso de que la copia de seguridad de S3 esté habilitada, se utilizará para registrar cualquier error relacionado con la entrega en el bucket de copias de seguridad de S3.

Puede habilitar el registro de errores de Amazon Data Firehose a través de la AWS CLI, la API o CloudFormation mediante la configuración `CloudWatchLoggingOptions`. Para ello, cree previamente un grupo de registro y un flujo de registro. Le recomendamos destinar el grupo de registro y el flujo de registro exclusivamente al registro de errores de Amazon Data Firehose. Asegúrese también de que la política de IAM asociada tenga el permiso `"logs:putLogEvents"`. Para obtener más información, consulte Control del acceso con Amazon Data Firehose.

Tenga en cuenta que Amazon Data Firehose no garantiza el envío de todos los registros de errores de entrega a Registros de CloudWatch. Si la tasa de errores de entrega es alta, Amazon Data Firehose toma muestras de los registros de errores de entrega antes de enviarlos a Registros de CloudWatch.

Se aplica un cargo nominal por los registros de errores enviados a Registros de CloudWatch. Para obtener más información, consulte Precios de Amazon CloudWatch.

Contenido

Errores de entrega de datos

A continuación, se ofrece una lista de códigos y mensajes de error de entrega de datos según el destino de Amazon Data Firehose. Cada mensaje de error también describe qué debe hacerse para solucionar el problema.

Errores

- Errores de entrega de datos de ElasticSearch
- Errores de entrega de datos de puntos de conexión HTTPS
- Errores de entrega de datos de Amazon OpenSearch Service
- Errores de invocación de Lambda
- Errores de invocación de Kinesis
- Errores de invocación de DirectPut de Kinesis
- Errores de invocación de AWS Glue
- Errores de invocación de DataFormatConversion

Errores de entrega de datos de Amazon S3

Amazon Data Firehose puede enviar los siguientes errores relacionados con Amazon S3 a Registros de CloudWatch.

Código de error	Mensaje de error e información
`S3.KMS.NotFoundException`	"La clave de AWS KMS proporcionada no se ha encontrado. Si cree que la clave de AWS KMS que está utilizando es válida y que su rol es el adecuado, compruebe si hay un problema con la cuenta a la que está vinculada la clave de AWS KMS."
`S3.KMS.RequestLimitExceeded`	"El límite de solicitudes de KMS por segundo se ha superado al intentar cifrar objetos de S3. Aumente el límite de solicitudes por segundo." Para obtener más información, consulte Límites en la Guía para desarrolladores de AWS Key Management Service.
`S3.AccessDenied`	"Acceso denegado. Asegúrese de que la política de confianza del rol de IAM proporcionado permita que Amazon Data Firehose asuma el rol y que la política de acceso permita el acceso al bucket de S3.”
`S3.AccountProblem`	“There is a problem with your AWS account that prevents the operation from completing successfully. Contact AWS Support.”
`S3.AllAccessDisabled`	"El acceso a la cuenta proporcionada se ha deshabilitado. Contact AWS Support.”

Código de error	Mensaje de error e información
`S3.InvalidPayer`	"El acceso a la cuenta proporcionada se ha deshabilitado. Contact AWS Support.”
`S3.NotSignedUp`	“The account is not signed up for Amazon S3. Inscríbase o utilice otra cuenta."
`S3.NoSuchBucket`	"El bucket especificado no existe. Créelo o utilice otro que exista."
`S3.Method NotAllowed`	"Este recurso no admite el método especificado. Modify the bucket’s policy to allow the correct Amazon S3 operation permissions.”
`InternalError`	"Se ha producido un error interno al intentar entregar los datos. Delivery will be retried; if the error persists, then it will be reported to AWS for resolution.”
`S3.KMS.KeyDisabled`	“The provided KMS key is disabled. Enable the key or use a different key.”
`S3.KMS.InvalidStateException`	“The provided KMS key is in an invalid state. Please use a different key.”
`KMS.InvalidStateException`	“The provided KMS key is in an invalid state. Please use a different key.”
`KMS.DisabledException`	“The provided KMS key is disabled. Please fix the key or use a different key.”
`S3.SlowDown`	“The rate of put request to the specified bucket was too high. Increase Firehose stream buffer size or reduce put requests from other applications.”
`S3.SubscriptionRequired`	“Access was denied when calling S3. Ensure that the IAM role and the KMS Key (if provided) passed in has Amazon S3 subscription.”

Código de error	Mensaje de error e información
`S3.InvalidToken`	“The provided token is malformed or otherwise invalid. Please check the credentials provided.”
`S3.KMS.KeyNotConfigured`	“KMS key not configured. Configure your KMSMasterKeyID, or disable encryption for your S3 bucket.”
`S3.KMS.AsymmetricCMKNotSupported`	“Amazon S3 supports only symmetric CMKs. You cannot use an asymmetric CMK to encrypt your data in Amazon S3. To get the type of your CMK, use the KMS DescribeKey operation.”
`S3.IllegalLocationConstraintException`	“Firehose currently uses s3 global endpoint for data delivery to the configured s3 bucket. The region of the configured s3 bucket doesn't support s3 global endpoint. Please create a Firehose stream in the same region as the s3 bucket or use s3 bucket in the region that supports global endpoint.”
`S3.InvalidPrefixConfigurationException`	“The custom s3 prefix used for the timestamp evaluation is invalid. Check your s3 prefix contains valid expressions for the current date and time of the year.”
`DataFormatConversion.MalformedData`	“Illegal character found between tokens.”

Errores de entrega de datos de tablas de Apache Iceberg

Para ver los errores de entrega de datos de tablas de Apache Iceberg, consulte Entrega de datos a tablas de Apache Iceberg.

Errores de entrega de datos de Amazon Redshift

Amazon Data Firehose puede enviar los siguientes errores relacionados con Amazon Redshift a Registros de CloudWatch.

Código de error	Mensaje de error e información
`Redshift.TableNotFound`	"La tabla a la que cargar datos no se ha encontrado. Asegúrese de que la tabla especificada exista." The destination table in Amazon Redshift to which data should be copied from S3 was not found. Tenga en cuenta que Amazon Data Firehose no crea la tabla de Amazon Redshift si no existe.
`Redshift.SyntaxError`	"El comando COPY contiene un error de sintaxis. Reintente el comando."
`Redshift.AuthenticationFailed`	"Error de autenticación del nombre de usuario y la contraseña. Proporcione un nombre de usuario y contraseña válidos."
`Redshift.AccessDenied`	"Acceso denegado. Asegúrese de que la política de confianza para el rol de IAM proporcionado permita que Amazon Data Firehose asuma el rol.”
`Redshift.S3BucketAccessDenied`	"El comando COPY no ha podido obtener acceso al bucket de S3. Ensure that the access policy for the provided IAM role allows access to the S3 bucket.”
`Redshift.DataLoadFailed`	"Se ha producido un error al cargar los datos en la tabla. Revise la tabla de sistema STL_LOAD_ERRORS para obtener más información."
`Redshift.ColumnNotFound`	"Una columna del comando COPY no existe en la tabla. Especifique un nombre de columna válido."
`Redshift.DatabaseNotFound`	“The database specified in the Amazon Redshift destination configuration or JDBC URL was not found. Especifique un nombre de base de datos válido."
`Redshift.IncorrectCopyOptions`	"Se han proporcionado opciones de COPY redundantes o en conflicto. Algunas opciones no son compatibles en determinadas combinaciones. Consulte la referencia de comandos COPY para obtener más información."

Código de error	Mensaje de error e información
	Para obtener más información, consulte Comando COPY de Amazon Redshift en la Guía para desarrolladores de bases de datos de Amazon Redshift.
`Redshift.MissingColumn`	"El esquema de la tabla incluye una columna definida como NO NULL sin un valor DEFAULT, pero que no se encuentra en la lista de columnas. Exclude this column, ensure that the loaded data always provides a value for this column, or add a default value to the Amazon Redshift schema for this table.”
`Redshift.ConnectionFailed`	“The connection to the specified Amazon Redshift cluster failed. Ensure that security settings allow Amazon Data Firehose connections, that the cluster or database specified in the Amazon Redshift destination configuration or JDBC URL is correct, and that the cluster is available.”
`Redshift.ColumnMismatch`	"La cantidad de jsonpath del comando COPY y la cantidad de columnas de la tabla de destino deben coincidir. Reintente el comando."
`Redshift.IncorrectOrMissingRegion`	“Amazon Redshift attempted to use the wrong region endpoint for accessing the S3 bucket. Either specify a correct region value in the COPY command options or ensure that the S3 bucket is in the same region as the Amazon Redshift database.”
`Redshift.IncorrectJsonPathsFile`	"El formato del archivo jsonpath proporcionado no es un formato JSON compatible. Reintente el comando."
`Redshift.MissingS3File`	“One or more S3 files required by Amazon Redshift have been removed from the S3 bucket. Revise las políticas del bucket de S3 para borrar cualquier eliminación automática de archivos de S3."
`Redshift.InsufficientPrivilege`	"El usuario no tiene permisos para cargar datos en la tabla. Check the Amazon Redshift user permissions for the INSERT privilege.”

Código de error	Mensaje de error e información
`Redshift.ReadOnlyCluster`	"La consulta no se puede ejecutar porque el sistema está en modo de cambio de tamaño. Intente ejecutar la consulta de nuevo más tarde."
`Redshift.DiskFull`	"No se han podido cargar los datos ya que el disco está lleno. Increase the capacity of the Amazon Redshift cluster or delete unused data to free disk space."
`InternalError`	"Se ha producido un error interno al intentar entregar los datos. Delivery will be retried; if the error persists, then it will be reported to AWS for resolution."
`Redshift.ArgumentNotSupported`	"The COPY command contains unsupported options."
`Redshift.AnalyzeTableAccessDenied`	"Access denied. Copy from S3 to Redshift is failing because analyze table can only be done by table or database owner."
`Redshift.SchemaNotFound`	"The schema specified in the DataTableName of Amazon Redshift destination configuration was not found. Specify a valid schema name."
`Redshift.ColumnSpecifiedMoreThanOnce`	"There is a column specified more than once in the column list. Ensure that duplicate columns are removed."
`Redshift.ColumnNotNullWithoutDefault`	"There is a non-null column without DEFAULT that is not included in the column list. Ensure that such columns are included in the column list."

Código de error	Mensaje de error e información
`Redshift.IncorrectBucketRegion`	“Redshift attempted to use a bucket in a different region from the cluster. Please specify a bucket within the same region as the cluster.”
`Redshift.S3SlowDown`	“High request rate to S3. Reduce the rate to avoid getting throttled.”
`Redshift.InvalidCopyOptionForJson`	“Please use either auto or a valid S3 path for json copyOption.”
`Redshift.InvalidCopyOptionJSONPathFormat`	“COPY failed with error \"Invalid JSONPath format. Array index is out of range.\” Please rectify the JSONPath expression.”
`Redshift.InvalidCopyOptionRBACAclNotAllowed`	“COPY failed with error \"Cannot use RBAC acl framework while permission propagation is not enabled.\”
`Redshift.DiskSpaceQuotaExceeded`	“Transaction aborted due to disk space quota exceed. Free up disk space or request increased quota for the schema(s).”
`Redshift.ConnectionsLimitExceeded`	“Connection limit exceeded for user.”
`Redshift.SslNotSupported`	“The connection to the specified Amazon Redshift cluster failed because the server does not support SSL. Please check your cluster settings.”

Código de error	Mensaje de error e información
`Redshift.HoseNotFound`	“The hose has been deleted. Please check the status of your hose.”
`Redshift.Delimiter`	“The copyOptions delimiter in the copyCommand is invalid. Ensure that it is a single character.”
`Redshift.QueryCancelled`	“The user has canceled the COPY operation.”
`Redshift.CompressionMismatch`	“Hose is configured with UNCOMPRESSED, but copyOption includes a compression format.”
`Redshift.EncryptionCredentials`	“The ENCRYPTED option requires credentials in the format: 'aws_iam_role=...;master_symmetric_key=...' or 'aws_access_key_id=...;aws_secret_access_key=...[;token=...];master_symmetric_key=...'.”
`Redshift.InvalidCopyOptions`	“Invalid COPY configuration options.”
`Redshift.InvalidMessageFormat`	“Copy command contains an invalid character.”
`Redshift.TransactionIdLimitReached`	“Transaction ID limit reached.”
`Redshift.DestinationRemoved`	“Please verify that the redshift destination exists and is configured correctly in the Firehose configuration.”
`Redshift.OutOfMemory`	“The Redshift cluster is running out of memory. Please ensure the cluster has sufficient capacity.”

Código de error	Mensaje de error e información
`Redshift.CannotForkProcess`	“The Redshift cluster is running out of memory. Please ensure the cluster has sufficient capacity.”
`Redshift.SslFailure`	“The SSL connection closed during the handshake.”
`Redshift.Resize`	“The Redshift cluster is resizing. Firehose will not be able to deliver data while the cluster is resizing.”
`Redshift.ImproperQualifiedName`	“The qualified name is improper (too many dotted names).”
`Redshift.InvalidJsonPathFormat`	“Invalid JSONPath Format.”
`Redshift.TooManyConnectionsException`	“Too many connections to Redshift.”
`Redshift.PSQLException`	“PSQlException observed from Redshift.”
`Redshift.DuplicateSecondsSpecification`	“Duplicate seconds specification in date/time format.”
`Redshift.RelationCouldNotBeOpened`	“Encountered Redshift error, relation could not be opened. Check Redshift logs for the specified DB.”

Código de error	Mensaje de error e información
`Redshift.TooManyClients`	“Encountered too many clients exception from Redshift. Revisit max connections to the database if there are multiple producers writing to it simultaneously.”

Errores de entrega de datos de Snowflake

Firehose puede enviar los siguientes errores relacionados con Snowflake a Registros de CloudWatch.

Código de error	Mensaje de error e información
`Snowflake.InvalidUrl`	“Firehose no se puede conectar a Snowflake. Asegúrese de que la URL de la cuenta esté especificada correctamente en la configuración de destino de Snowflake.”
`Snowflake.InvalidUser`	“Firehose no se puede conectar a Snowflake. Asegúrese de que el usuario esté especificado correctamente en la configuración de destino de Snowflake.”
`Snowflake.InvalidRole`	“El rol de snowflake especificado no existe o no está autorizado. Asegúrese de que el rol esté asignado al usuario especificado”
`Snowflake.InvalidTable`	“La tabla proporcionada no existe o no está autorizada”
`Snowflake.InvalidSchema`	“El esquema proporcionado no existe o no está autorizado”
`Snowflake.InvalidDatabase`	“La base de datos proporcionada no existe o no está autorizada”
`Snowflake.InvalidPrivateKeyOrPassphrase`	“La clave privada o la frase de contraseña especificada no es válida. Tenga en cuenta que la clave privada proporcionada debe ser una clave privada PEM RSA válida”

Código de error	Mensaje de error e información
`Snowflake.MissingColumns`	“La solicitud de inserción se rechaza porque faltan columnas en la carga útil de entrada. Asegúrese de que los valores estén especificados para todas las columnas que no admiten valores NULL”
`Snowflake.ExtraColumns`	“La solicitud de inserción se rechaza debido a que hay columnas adicionales. No se deben especificar las columnas que no estén presentes en la tabla”
`Snowflake.InvalidInput`	“Ocurrió un error en la entrega debido a un formato de entrada no válido. Asegúrese de que la carga útil de entrada proporcionada esté en el formato JSON aceptable”
`Snowflake.IncorrectValue`	“La entrega ha fallado debido a un tipo de datos incorrecto en la carga útil de entrada. Asegúrese de que los valores JSON especificados en la carga útil de entrada se ajusten al tipo de datos declarado en la definición de la tabla de Snowflake”

Errores de entrega de datos de Splunk

Amazon Data Firehose puede enviar los siguientes errores relacionados con Splunk a Registros de CloudWatch.

Código de error	Mensaje de error e información
`Splunk.ProxyWithoutStickySessions`	“Si tiene un proxy (ELB u otro) entre Amazon Data Firehose y el nodo de HEC, debe habilitar las sesiones persistentes para que sea compatible con los ACK de HEC.”
`Splunk.DisabledToken`	"El token de HEC está deshabilitado. Habilite el token para permitir la entrega de datos a Splunk.".
`Splunk.InvalidToken`	"El token de HEC no es válido. Actualice Amazon Data Firehose con un token de HEC válido”.

Código de error	Mensaje de error e información
`Splunk.InvalidDataFormat`	"Los datos no están en el formato correcto. Para ver cómo dar a los datos el formato correcto para puntos de enlace de HEC de eventos o sin procesar, consulte Splunk Event Data".
`Splunk.InvalidIndex`	"El token o la entrada de HEC están configurados con un índice no válido. Compruebe la configuración del índice e inténtelo de nuevo.".
`Splunk.ServerError`	"Data delivery to Splunk failed due to a server error from the HEC node. Amazon Data Firehose volverá a intentar enviar los datos si la duración del reintento en su Amazon Data Firehose es superior a 0. If all the retries fail, Amazon Data Firehose backs up the data to Amazon S3."
`Splunk.DisabledAck`	"El reconocimiento de indexadores está deshabilitado en el token de HEC. Habilite el reconocimiento de indexadores e inténtelo de nuevo. Para obtener más información, consulte Enable indexer acknowledgement.".
`Splunk.AckTimeout`	"No recibió ningún reconocimiento de parte del HEC antes de que el tiempo de espera de reconocimiento del HEC se agotara. 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."
`Splunk.MaxRetriesFailed`	"Error al entregar datos a Splunk o al recibir confirmación. Compruebe el estado del HEC y vuelva a intentarlo.".
`Splunk.ConnectionTimeout`	"Se ha agotado el tiempo de espera de conexión a Splunk. This might be a transient error and the request will be retried. Amazon Data Firehose backs up the data to Amazon S3 if all retries fail."
`Splunk.InvalidEndpoint`	"No se ha podido establecer una conexión con el punto de enlace del HEC. Asegúrese de que la URL del punto de conexión HEC es válido y Amazon Data Firehose puede llegar a ella."

Código de error	Mensaje de error e información
`Splunk.ConnectionClosed`	"No se pueden enviar datos a Splunk debido a un error de conexión. Posiblemente sea un error temporal. Si aumenta la duración de demora en la configuración de Amazon Data Firehose, se pueden prevenir estos errores temporales.”
`Splunk.SSLUnverified`	"No se ha podido establecer una conexión con el punto de enlace del HEC. El host no coincide con el certificado proporcionadas por el homólogo. Asegúrese de que el certificado y el host son válidos."
`Splunk.SSLHandshake`	"No se ha podido establecer una conexión con el punto de enlace del HEC. Asegúrese de que el certificado y el host son válidos."
`Splunk.URLNotFound`	“The requested URL was not found on the Splunk server. Please check the Splunk cluster and make sure it is configured correctly.”
`Splunk.ServerError.ContentTooLarge`	“Data delivery to Splunk failed due to a server error with a statusCode: 413, message: the request your client sent was too large. See splunk docs to configure max_content_length.”
`Splunk.IndexerBusy`	“Data delivery to Splunk failed due to a server error from the HEC node. Make sure HEC endpoint or the Elastic Load Balancer is reachable and is healthy.”
`Splunk.ConnectionRecycled`	“The connection from Firehose to Splunk has been recycled. Delivery will be retried.”
`Splunk.AcknowledgementsDisabled`	“Could not get acknowledgements on POST. Make sure that acknowled gements are enabled on HEC endpoint.”
`Splunk.InvalidHecResponseCharacter`	“Invalid characters found in HEC response, make sure to check to the service and HEC configuration.”

Errores de entrega de datos de ElasticSearch

Amazon Data Firehose puede enviar los siguientes errores relacionados con ElasticSearch a Registros de CloudWatch.

Código de error	Mensaje de error e información
`ES.AccessDenied`	"Acceso denegado. Ensure that the provided IAM role associated with firehose is not deleted.”
`ES.ResourceNotFound`	“The specified AWS Elasticsearch domain does not exist.”

Errores de entrega de datos de puntos de conexión HTTPS

Amazon Data Firehose puede enviar los siguientes errores relacionados con los puntos de conexión HTTP a Registros de CloudWatch. Si ninguno de estos errores coincide con el problema que experimenta, el error predeterminado es el siguiente: “An internal error occurred while attempting to deliver data. Delivery will be retried; if the error persists, then it will be reported to AWS for resolution.”

Código de error	Mensaje de error e información
`HttpEndpoint.RequestTimeout`	Se agotó el tiempo de espera de la entrega antes de recibir una respuesta y se volverá a intentar. Si el error persiste, póngase en contacto con el equipo del servicio Firehose de AWS.
`HttpEndpoint.ResponseTooLarge`	“The response received from the endpoint is too large. Contact the owner of the endpoint to resolve this issue.”
`HttpEndpoint.InvalidResponseFromDestination`	“The response received from the specified endpoint is invalid. Contact the owner of the endpoint to resolve the issue.”
`HttpEndpoint.Desti`	“The following response was received from the endpoint destination.”

Código de error	Mensaje de error e información
`nationExc eption`	
`HttpEndpo int.Conne ctionFailed`	“Unable to connect to the destination endpoint. Contact the owner of the endpoint to resolve this issue.”
`HttpEndpo int.Conne ctionReset`	“Unable to maintain connection with the endpoint. Contact the owner of the endpoint to resolve this issue.”
`HttpEndpo int.Conne ctionReset`	“Trouble maintaining connection with the endpoint. Please reach out to the owner of the endpoint.”
`HttpEndpo int.Respo nseReason PhraseExc eededLimit`	“The response reason phrase received from the endpoint exceed the configured limit of 64 characters.”
`HttpEndpo int.Inval idRespons eFromDest ination`	“The response received from the endpoint is invalid. See Troubleshooting HTTP Endpoints in the Firehose documentation for more information. Reason: .”
`HttpEndpo int.Desti nationExc eption`	“Delivery to the endpoint was unsuccessful. See Troubleshooting HTTP Endpoints in the Firehose documentation for more information. Response received with status code .”
`HttpEndpo int.Inval idStatusCode`	“Received an invalid response status code.”

Código de error	Mensaje de error e información
`HttpEndpoint.SSLHandshakeFailure`	“Unable to complete an SSL Handshake with the endpoint. Contact the owner of the endpoint to resolve this issue.”
`HttpEndpoint.SSLHandshakeFailure`	“Unable to complete an SSL Handshake with the endpoint. Contact the owner of the endpoint to resolve this issue.”
`HttpEndpoint.SSLFailure`	“Unable to complete TLS handshake with the endpoint. Contact the owner of the endpoint to resolve this issue.”
`HttpEndpoint.SSLHandshakeCertificatePathFailure`	“Unable to complete an SSL Handshake with the endpoint due to invalid certification path. Contact the owner of the endpoint to resolve this issue.”
`HttpEndpoint.SSLHandshakeCertificatePathValidationFailure`	“Unable to complete an SSL Handshake with the endpoint due to certification path validation failure. Contact the owner of the endpoint to resolve this issue.”
`HttpEndpoint.MakeRequestFailure.IllegalUriException`	“HttpEndpoint request failed due to invalid input in URI. Please make sure all the characters in the input URI are valid.”

Código de error	Mensaje de error e información
`HttpEndpoint.MakeRequestFailure.IllegalCharacterInHeaderValue`	“HttpEndpoint request failed due to illegal response error. Illegal character '\n' in header value.”
`HttpEndpoint.IllegalResponseFailure`	“HttpEndpoint request failed due to illegal response error. HTTP message must not contain more than one Content-Type header.”
`HttpEndpoint.IllegalMessageStart`	“HttpEndpoint request failed due to illegal response error. Illegal HTTP message start. See Troubleshooting HTTP Endpoints in the Firehose documentation for more information.”

Errores de entrega de datos de Amazon OpenSearch Service

En el caso del destino OpenSearch Service, Amazon Data Firehose envía los errores a Registros de CloudWatch a medida que OpenSearch Service los devuelve.

Además de los errores que pueden provenir de los clústeres de OpenSearch, es posible que se produzcan los dos errores siguientes:

- Se produce un error de autenticación o autorización al intentar entregar los datos en el clúster de OpenSearch Service de destino. Esto puede ocurrir debido a problemas de permisos o de forma intermitente cuando se modifica la configuración del dominio de OpenSearch Service de destino de Amazon Data Firehose. Compruebe la política del clúster y los permisos del rol.
- No se pudieron entregar los datos en el clúster de OpenSearch Service de destino debido a errores de autenticación o autorización. Esto puede ocurrir debido a problemas de permisos o de forma intermitente cuando se modifica la configuración del dominio de OpenSearch Service de destino de Amazon Data Firehose. Compruebe la política del clúster y los permisos del rol.

Código de error	Mensaje de error e información
`OS.AccessDenied`	"Acceso denegado. Ensure that the trust policy for the provided IAM role allows Firehose to assume the role, and the access policy allows access to the Amazon OpenSearch Service API.”
`OS.AccessDenied`	"Acceso denegado. Ensure that the trust policy for the provided IAM role allows Firehose to assume the role, and the access policy allows access to the Amazon OpenSearch Service API.”
`OS.AccessDenied`	"Acceso denegado. Ensure that the provided IAM role associated with firehose is not deleted.”
`OS.AccessDenied`	"Acceso denegado. Ensure that the provided IAM role associated with firehose is not deleted.”
`OS.ResourceNotFound`	“The specified Amazon OpenSearch Service domain does not exist.”
`OS.ResourceNotFound`	“The specified Amazon OpenSearch Service domain does not exist.”
`OS.AccessDenied`	"Acceso denegado. Ensure that the trust policy for the provided IAM role allows Firehose to assume the role, and the access policy allows access to the Amazon OpenSearch Service API.”
`OS.RequestTimeout`	“Request to the Amazon OpenSearch Service cluster or OpenSearch Serverless collection timed out. Ensure that the cluster or collection has sufficient capacity for the current workload.”
`OS.ClusterError`	“The Amazon OpenSearch Service cluster returned an unspecified error.”
`OS.RequestTimeout`	“Request to the Amazon OpenSearch Service cluster timed out. Ensure that the cluster has sufficient capacity for the current workload.”
`OS.ConnectionFailed`	“Trouble connecting to the Amazon OpenSearch Service cluster or OpenSearch Serverless collection. Ensure that the cluster or collection is healthy and reachable.”

Código de error	Mensaje de error e información
`OS.ConnectionReset`	“Unable to maintain connection with the Amazon OpenSearch Service cluster or OpenSearch Serverless collection. Contact the owner of the cluster or collection to resolve this issue.”
`OS.ConnectionReset`	“Trouble maintaining connection with the Amazon OpenSearch Service cluster or OpenSearch Serverless collection. Ensure that the cluster or collection is healthy and has sufficient capacity for the current workload.”
`OS.ConnectionReset`	“Trouble maintaining connection with the Amazon OpenSearch Service cluster or OpenSearch Serverless collection. Ensure that the cluster or collection is healthy and has sufficient capacity for the current workload.”
`OS.AccessDenied`	"Acceso denegado. Ensure that the access policy on the Amazon OpenSearch Service cluster grants access to the configured IAM role.”
`OS.ValidationException`	“The OpenSearch cluster returned a ESServiceException. One of the reasons is that the cluster has been upgraded to OS 2.x or higher, but the hose still has the TypeName parameter configured. Update the hose configuration by setting the TypeName to an empty string, or change the endpoint to the cluster, that supports the Type parameter.”
`OS.ValidationException`	“Member must satisfy regular expression pattern: [a-z][a-z0-9\\-]+.”
`OS.JsonParseException`	“The Amazon OpenSearch Service cluster returned a JsonParse Exception. Ensure that the data being put is valid.”
`OS.AmazonOpenSearchServiceParseException`	“The Amazon OpenSearch Service cluster returned an AmazonOpenSearchServiceParseException. Ensure that the data being put is valid.”
`OS.ExplicitIndexInBulkNotAllowed`	“Ensure rest.action.multi.allow_explicit_index is set to true on the Amazon OpenSearch Service cluster.”

Código de error	Mensaje de error e información
`OS.ClusterError`	“The Amazon OpenSearch Service cluster or OpenSearch Serverless collection returned an unspecified error.”
`OS.ClusterBlockException`	“The cluster returned a ClusterBlockException. It may be overloaded.”
`OS.InvalidARN`	“The Amazon OpenSearch Service ARN provided is invalid. Please check your DeliveryStream configuration.”
`OS.MalformedData`	“One or more records are malformed. Please ensure that each record is single valid JSON object and that it does not contain newlines.”
`OS.InternalError`	“An internal error occurred when attempting to deliver data. Delivery will be retried; if the error persists, it will be reported to AWS for resolution.”
`OS.AliasWithMultipleIndicesNotAllowed`	“Alias has more than one indices associated with it. Ensure that the alias has only one index associated with it.”
`OS.UnsupportedVersion`	“Amazon OpenSearch Service 6.0 is not currently supported by Amazon Data Firehose. Contact AWS Support for more information.”
`OS.CharConversionException`	“One or more records contained an invalid character.”
`OS.InvalidDomainNameLength`	“The domain name length is not within valid OS limits.”
`OS.VPCDomainNotSupported`	“Amazon OpenSearch Service domains within VPCs are currently not supported.”

Código de error	Mensaje de error e información
`OS.ConnectionError`	“The http server closed the connection unexpectedly, please verify the health of the Amazon OpenSearch Service cluster or OpenSearch Serverless collection.”
`OS.LargeFieldData`	“The Amazon OpenSearch Service cluster aborted the request as it contained a field data larger than allowed.”
`OS.BadGateway`	“The Amazon OpenSearch Service cluster or OpenSearch Serverless collection aborted the request with a response: 502 Bad Gateway.”
`OS.ServiceException`	“Error received from the Amazon OpenSearch Service cluster or OpenSearch Serverless collection. If the cluster or collection is behind a VPC, ensure network configuration allows connectivity.”
`OS.GatewayTimeout`	“Firehose encountered timeout errors when connecting to the Amazon OpenSearch Service cluster or OpenSearch Serverless collection.”
`OS.MalformedData`	“Amazon Data Firehose does not support Amazon OpenSearch Service Bulk API commands inside the Firehose record.”
`OS.ResponseEntryCountMismatch`	“The response from the Bulk API contained more entries than the number of records sent. Ensure that each record contains only one JSON object and that there are no newlines.”

Errores de invocación de Lambda

Amazon Data Firehose puede enviar los siguientes errores de invocación de Lambda a Registros de CloudWatch.

Código de error	Mensaje de error e información
`Lambda.AssumeRoleAccessDenied`	"Acceso denegado. Asegúrese de que la política de confianza para el rol de IAM proporcionado permita que Amazon Data Firehose asuma el rol.”

Código de error	Mensaje de error e información
`Lambda.InvokeAccessDenied`	"Acceso denegado. Ensure that the access policy allows access to the Lambda function.”
`Lambda.JsonProcessingException`	“There was an error parsing returned records from the Lambda function. Ensure that the returned records follow the status model required by Amazon Data Firehose.” Para obtener más información, consulte Parámetros necesarios para la transformación de datos.
`Lambda.InvokeLimitExceeded`	“The Lambda concurrent execution limit is exceeded. Aumente el límite de ejecución simultánea." Para obtener más información, consulte Límites de AWS Lambda en la Guía para desarrolladores de AWS Lambda.
`Lambda.DuplicatedRecordId`	"Se han devuelto varios registros con el mismo ID de registro. Ensure that the Lambda function returns unique record IDs for each record.” Para obtener más información, consulte Parámetros necesarios para la transformación de datos.
`Lambda.MissingRecordId`	"Uno o varios ID de registro no se devolverán. Ensure that the Lambda function returns all received record IDs.” Para obtener más información, consulte Parámetros necesarios para la transformación de datos.
`Lambda.ResourceNotFound`	“The specified Lambda function does not exist. Use otra función que exista."
`Lambda.InvalidSubnetIDException`	“The specified subnet ID in the Lambda function VPC configuration is invalid. Asegúrese de que el ID de subred sea válido."

Código de error	Mensaje de error e información
`Lambda.InvalidSecurityGroupIDException`	“The specified security group ID in the Lambda function VPC configuration is invalid. Asegúrese de que el ID del grupo de seguridad sea válido."
`Lambda.SubnetIPAddressLimitReachedException`	“AWS Lambda was not able to set up the VPC access for the Lambda function because one or more configured subnets have no available IP addresses. Aumente el límite de direcciones IP." Para obtener más información, consulte Límites de Amazon VPC: VPC y subredes en la Guía del usuario de Amazon VPC.
`Lambda.ENILimitReachedException`	“AWS Lambda was not able to create an Elastic Network Interface (ENI) in the VPC, specified as part of the Lambda function configuration, because the limit for network interfaces has been reached. Aumente el límite de interfaces de red." Para obtener más información, consulte Límites de Amazon VPC: interfaces de red en la Guía del usuario de Amazon VPC.
`Lambda.FunctionTimedOut`	Se agotó el tiempo de espera de la función de Lambda. Aumente la configuración de tiempo de espera en la función de Lambda. Para obtener más información, consulte Configuración del tiempo de espera de la función.

Código de error	Mensaje de error e información
`Lambda.FunctionError`	Puede deberse a uno de los siguientes errores: • La estructura de la salida no es válida. Compruebe su función y asegúrese de que la salida esté en el formato requerido. Además, asegúrese de que los registros procesados contengan un estado de resultado válido (`Dropped`, `Ok` o `ProcessingFailed`). • La función de Lambda se invocó correctamente, pero devolvió un resultado de error. • Lambda no pudo descifrar las variables de entorno porque se denegó el acceso a la clave de KMS. Compruebe la configuración de las claves de KMS de la función, así como la política de claves. Para obtener más información, consulte Troubleshooting Key Access.
`Lambda.FunctionRequestTimedOut`	Amazon Data Firehose detectó que la solicitud no se completó antes del error de configuración del tiempo de espera de la solicitud al invocar Lambda. Revise el código de Lambda para comprobar si está previsto que se ejecute más allá del tiempo de espera configurado. Si es así, considere la posibilidad de ajustar la configuración de Lambda, incluida la memoria y el tiempo de espera. Para obtener más información, consulte Configuración de las opciones de las funciones de Lambda.
`Lambda.TargetServerFailedToRespond`	Amazon Data Firehose detectó un error. El servidor de destino no respondió al error al llamar al servicio AWS Lambda.
`Lambda.InvalidZipFileException`	Amazon Data Firehose detectó una InvalidZipFileException al invocar la función de Lambda. Compruebe la configuración de la función de Lambda y el archivo zip de código de Lambda.

Código de error	Mensaje de error e información
`Lambda.InternalServerError`	“Amazon Data Firehose detectó un InternalServerError al llamar al servicio de AWS Lambda. Amazon Data Firehose volverá a intentar enviar los datos un número fijo de veces. Puede especificar o anular las opciones de reintento con las API `CreateDeliveryStream` o `UpdateDestination` . Si el error persiste, póngase en contacto con el equipo de asistencia de AWS Lambda.
`Lambda.ServiceUnavailable`	Amazon Data Firehose detectó una ServiceUnavailableException al llamar al servicio de AWS Lambda. Amazon Data Firehose volverá a intentar enviar los datos un número fijo de veces. Puede especificar o anular las opciones de reintento con las API `CreateDeliveryStream` o `UpdateDestination` . Si el error persiste, póngase en contacto con la asistencia de AWS Lambda.
`Lambda.InvalidSecurityToken`	No se puede invocar la función de Lambda debido a que el token de seguridad no es válido. No se admite la invocación de Lambda entre particiones.
`Lambda.InvocationFailure`	Puede deberse a uno de los siguientes errores: • Amazon Data Firehose detectó errores al llamar a AWS Lambda. La operación se reintentará y, si el error persiste, se notificará a AWS para solucionarlo. • Amazon Data Firehose detectó una KMSInvalidStateException de Lambda. Lambda no pudo descifrar las variables de entorno porque la clave KMS utilizada no es un estado válido para Descifrar. Compruebe la clave de KMS de la función de Lambda. • Amazon Data Firehose detectó una AWSLambdaException de Lambda. Lambda no pudo inicializar la imagen del contenedor proporcionada. Verifique la imagen. • Amazon Data Firehose detectó errores de tiempo de espera al llamar a AWS Lambda. El tiempo de espera máximo admitido de la función es de 5 minutos. Para obtener más información, consulte Data Transformation Execution Duration.

Código de error	Mensaje de error e información
`Lambda.JsonMappingException`	Se ha producido un error al analizar los registros devueltos de la función de Lambda. Asegúrese de que el campo de datos esté codificado en base64.

Errores de invocación de Kinesis

Amazon Data Firehose puede enviar los siguientes errores de invocación a Registros de CloudWatch.

Código de error	Mensaje de error e información
`Kinesis.AccessDenied`	“Access was denied when calling Kinesis. Ensure the access policy on the IAM role used allows access to the appropriate Kinesis APIs.”
`Kinesis.ResourceNotFound`	“Firehose failed to read from the stream. If the Firehose is attached with Kinesis Stream, the stream may not exist, or the shard may have been merged or split. If the Firehose is of DirectPut type, the Firehose may not exist any more.”
`Kinesis.SubscriptionRequired`	“Access was denied when calling Kinesis. Ensure that the IAM role passed for Kinesis stream access has AWS Kinesis subscription.”
`Kinesis.Throttling`	“Throttling error encountered when calling Kinesis. This can be due to other applications calling the same APIs as the Firehose stream, or because you have created too many Firehose streams with the same Kinesis stream as the source.”
`Kinesis.Throttling`	“Throttling error encountered when calling Kinesis. This can be due to other applications calling the same APIs as the Firehose stream, or because you have created too many Firehose streams with the same Kinesis stream as the source.”
`Kinesis.AccessDenied`	“Access was denied when calling Kinesis. Ensure the access policy on the IAM role used allows access to the appropriate Kinesis APIs.”

Código de error	Mensaje de error e información
`Kinesis.AccessDenied`	“Access was denied while trying to call API operations on the underlying Kinesis Stream. Ensure that the IAM role is propagated and valid.”
`Kinesis.KMS.AccessDeniedException`	“Firehose does not have access to the KMS Key used to encrypt/decrypt the Kinesis Stream. Please grant the Firehose delivery role access to the key.”
`Kinesis.KMS.KeyDisabled`	“Firehose is unable to read from the source Kinesis Stream because the KMS key used to encrypt/decrypt it is disabled. Enable the key so that reads can proceed.”
`Kinesis.KMS.InvalidStateException`	“Firehose is unable to read from the source Kinesis Stream because the KMS key used to encrypt it is in an invalid state.”
`Kinesis.KMS.NotFoundException`	“Firehose is unable to read from the source Kinesis Stream because the KMS key used to encrypt it was not found.”

Errores de invocación de DirectPut de Kinesis

Amazon Data Firehose puede enviar los siguientes errores de invocación de DirectPut de Kinesis a Registros de CloudWatch.

Código de error	Mensaje de error e información
`Firehose.KMS.AccessDeniedException`	“Firehose does not have access to the KMS Key. Please check the key policy.”
`Firehose.KMS.Inval`	“Firehose is unable to decrypt the data because the KMS key used to encrypt it is in an invalid state.”

Código de error	Mensaje de error e información
`idStateEx ception`	
`Firehose. KMS.NotFo undException`	“Firehose is unable to decrypt the data because the KMS key used to encrypt it was not found.”
`Firehose. KMS.KeyDi sabled`	“Firehose is unable to decrypt the data because the KMS key used to encrypt the data is disabled. Enable the key so that data delivery can proceed.”

Errores de invocación de AWS Glue

Amazon Data Firehose puede enviar los siguientes errores de invocación de AWS Glue a Registros de CloudWatch.

Código de error	Mensaje de error e información
`DataForma tConversi on.Invali dSchema`	“The schema is invalid.”
`DataForma tConversi on.Entity NotFound`	“The specified table/database could not be found. Please ensure that the table/database exists and that the values provided in the schema configuration are correct, especially with regards to casing.”
`DataForma tConversi on.Invali dInput`	“Could not find a matching schema from glue. Please make sure the specified database with the supplied catalog ID exists.”
`DataForma tConversi`	“Could not find a matching schema from glue. Please make sure the passed ARN is in the correct format.”

Código de error	Mensaje de error e información
`on.Invali dInput`	
`DataForma tConversi on.Invali dInput`	“Could not find a matching schema from glue. Please make sure the catalogId provided is valid.”
`DataForma tConversi on.Invali dVersionId`	“Could not find a matching schema from glue. Please make sure the specified version of the table exists.”
`DataForma tConversi on.NonExi stentColumns`	“Could not find a matching schema from glue. Please make sure the table is configured with a non-null storage descriptor containing the target columns.”
`DataForma tConversi on.Access Denied`	“Access was denied when assuming role. Please ensure that the role specified in the data format conversion configuration has granted the Firehose service permission to assume it.”
`DataForma tConversi on.Thrott ledByGlue`	“Throttling error encountered when calling Glue. Either increase the request rate limit or reduce the current rate of calling glue through other applications.”
`DataForma tConversi on.Access Denied`	“Access was denied when calling Glue. Please ensure that the role specified in the data format conversion configuration has the necessary permissions.”

Código de error	Mensaje de error e información
`DataFormatConversion.InvalidGlueRole`	“Invalid role. Please ensure that the role specified in the data format conversion configuration exists.”
`DataFormatConversion.InvalidGlueRole`	“The security token included in the request is invalid. Ensure that the provided IAM role associated with firehose is not deleted.”
`DataFormatConversion.GlueNotAvailableInRegion`	“AWS Glue is not yet available in the region you have specified; please specify a different region.”
`DataFormatConversion.GlueEncryptionException`	“There was an error retrieving the master key. Ensure that the key exists and has the correct access permissions.”
`DataFormatConversion.SchemaValidationTimeout`	“Timed out while retrieving table from Glue. If you have a large number of Glue table versions, please add 'glue:GetTableVersion' permission (recommended) or delete unused table versions. If you do not have a large number of tables in Glue, please contact AWS Support.”
`DataFirehose.InternalError`	“Timed out while retrieving table from Glue. If you have a large number of Glue table versions, please add 'glue:GetTableVersion' permission (recommended) or delete unused table versions. If you do not have a large number of tables in Glue, please contact AWS Support.”

Código de error	Mensaje de error e información
`DataFormatConversion.GlueEncryptionException`	“There was an error retrieving the master key. Ensure that the key exists and state is correct.”

Errores de invocación de DataFormatConversion

Amazon Data Firehose puede enviar los siguientes errores de invocación de DataFormatConversion a Registros de CloudWatch.

Código de error	Mensaje de error e información
`DataFormatConversion.InvalidSchema`	“The schema is invalid.”
`DataFormatConversion.ValidationException`	“Column names and types must be non-empty strings.”
`DataFormatConversion.ParseError`	“Encountered malformed JSON.”
`DataFormatConversion.MalformedData`	“Data does not match the schema.”
`DataFormatConversi`	“Length of json key must not be greater than 262144”

Código de error	Mensaje de error e información
`on.Malfor medData`	
`DataForma tConversi on.Malfor medData`	“The data cannot be decoded as UTF-8.”
`DataForma tConversi on.Malfor medData`	“Illegal character found between tokens.”
`DataForma tConversi on.Invali dTypeFormat`	“The type format is invalid. Check the type syntax.”
`DataForma tConversi on.Invali dSchema`	“Invalid Schema. Please ensure that there are no special characters or white spaces in column names.”
`DataForma tConversi on.Invali dRecord`	“Record is not as per schema. One or more map keys were invalid for map<string,string>.”
`DataForma tConversi on.Malfor medData`	“The input JSON contained a primitive at the top level. The top level must be an object or array.”

Código de error	Mensaje de error e información
`DataFormatConversion.MalformedData`	“The input JSON contained a primitive at the top level. The top level must be an object or array.”
`DataFormatConversion.MalformedData`	“The record was empty or contained only whitespace.”
`DataFormatConversion.MalformedData`	“Encountered invalid characters.”
`DataFormatConversion.MalformedData`	“Encountered invalid or unsupported timestamp format. Please see the Firehose developer guide for supported timestamp formats.”
`DataFormatConversion.MalformedData`	“A scalar type was found in the data but a complex type was specified on the schema.”
`DataFormatConversion.MalformedData`	“Data does not match the schema.”
`DataFormatConversion.MalformedData`	“A scalar type was found in the data but a complex type was specified on the schema.”

Código de error	Mensaje de error e información
`DataFormatConversion.ConversionFailureException`	“ConversionFailureException”
`DataFormatConversion.DataFormatConversionCustomerErrorException`	“DataFormatConversionCustomerErrorException”
`DataFormatConversion.DataFormatConversionCustomerErrorException`	“DataFormatConversionCustomerErrorException”
`DataFormatConversion.MalformedData`	“Data does not match the schema.”
`DataFormatConversion.InvalidSchema`	“The schema is invalid.”

Código de error	Mensaje de error e información
`DataFormatConversion.MalformedData`	“Data does not match the schema. Invalid format for one or more dates.”
`DataFormatConversion.MalformedData`	“Data contains a highly nested JSON structure that is not supported.”
`DataFormatConversion.EntityNotFound`	“The specified table/database could not be found. Please ensure that the table/database exists and that the values provided in the schema configuration are correct, especially with regards to casing.”
`DataFormatConversion.InvalidInput`	“Could not find a matching schema from glue. Please make sure the specified database with the supplied catalog ID exists.”
`DataFormatConversion.InvalidInput`	“Could not find a matching schema from glue. Please make sure the passed ARN is in the correct format.”
`DataFormatConversion.InvalidInput`	“Could not find a matching schema from glue. Please make sure the catalogId provided is valid.”
`DataFormatConversion.InvalidVersionId`	“Could not find a matching schema from glue. Please make sure the specified version of the table exists.”

Código de error	Mensaje de error e información
`DataFormatConversion.NonExistentColumns`	“Could not find a matching schema from glue. Please make sure the table is configured with a non-null storage descriptor containing the target columns.”
`DataFormatConversion.AccessDenied`	“Access was denied when assuming role. Please ensure that the role specified in the data format conversion configuration has granted the Firehose service permission to assume it.”
`DataFormatConversion.ThrottledByGlue`	“Throttling error encountered when calling Glue. Either increase the request rate limit or reduce the current rate of calling glue through other applications.”
`DataFormatConversion.AccessDenied`	“Access was denied when calling Glue. Please ensure that the role specified in the data format conversion configuration has the necessary permissions.”
`DataFormatConversion.InvalidGlueRole`	“Invalid role. Please ensure that the role specified in the data format conversion configuration exists.”
`DataFormatConversion.GlueNotAvailableInRegion`	“AWS Glue is not yet available in the region you have specified; please specify a different region.”
`DataFormatConversion.GlueEncryptionException`	“There was an error retrieving the master key. Ensure that the key exists and has the correct access permissions.”

Código de error	Mensaje de error e información
`DataFormatConversion.SchemaValidationTimeout`	“Timed out while retrieving table from Glue. If you have a large number of Glue table versions, please add 'glue:GetTableVersion' permission (recommended) or delete unused table versions. If you do not have a large number of tables in Glue, please contact AWS Support.”
`DataFirehose.InternalError`	“Timed out while retrieving table from Glue. If you have a large number of Glue table versions, please add 'glue:GetTableVersion' permission (recommended) or delete unused table versions. If you do not have a large number of tables in Glue, please contact AWS Support.”
`DataFormatConversion.MalformedData`	“One or more fields have incorrect format.”

Acceso a los registros de CloudWatch para Amazon Data Firehose

Puede ver los registros de errores relacionados con los errores de entrega de datos de Amazon Data Firehose con la consola de Amazon Data Firehose o la consola de CloudWatch. Los siguientes procedimientos explican cómo obtener acceso a los logs de errores.

Acceso a los registros de errores mediante la consola de Amazon Data Firehose

1. Inicie sesión en la Consola de administración de AWS y abra la consola de Firehose en https://console.aws.amazon.com/firehose/.
2. En la barra de navegación, elija una región de AWS.
3. Elija un nombre de flujo de Firehose para ir a la página de detalles del flujo de Firehose.
4. Seleccione Error Log para ver una lista de logs de errores relacionados con los errores de entrega de datos.

Acceso a los registros de errores mediante la consola de CloudWatch

1. Abra la consola de CloudWatch en https://console.aws.amazon.com/cloudwatch/.

2. Seleccione una región en la barra de navegación.
3. En el panel de navegación, elija Logs (Registros).
4. Elija un grupo y una secuencia de registros para ver una lista de los registros de errores relacionados con el error de entrega de datos.

Supervisión del estado del agente de Kinesis

El agente de Kinesis publica métricas de CloudWatch personalizadas con un espacio de nombres de AWSKinesisAgent. Ayuda a evaluar si el agente se encuentra en buen estado, si envía datos a Amazon Data Firehose tal como se ha especificado y si consume la cantidad adecuada de recursos de CPU y memoria en el productor de datos.

Las métricas como el número de registros y bytes enviados resultan útiles para comprender la velocidad a la que el agente está enviando datos al flujo de Firehose. Cuando estas métricas caen por debajo de los umbrales previstos en determinado porcentaje o pasan a ser cero, esto podría indicar que existen problemas de configuración, errores de red o problemas con el estado del agente. Las métricas como, por ejemplo, el consumo de CPU y memoria de host y los contadores de errores del agente indican el uso de los recursos por parte del productor y proporcionan información útil sobre posibles errores de host o de configuración. Por último, el agente también registra excepciones de servicio para ayudar a investigar los problemas del agente.

Estas métricas relacionadas con el agente se notifican en la región especificada en la opción de configuración del agente `cloudwatch.endpoint`. Para obtener más información, consulte Especificar las opciones de configuración del agente.

Las métricas de CloudWatch publicadas desde varios agentes de Kinesis se agregan o combinan.

Se aplica un cargo nominal por las métricas emitidas desde el agente de Kinesis, que están habilitadas de forma predeterminada. Para obtener más información, consulte Precios de Amazon CloudWatch.

Monitorización con CloudWatch

El agente de Kinesis envía las siguientes métricas a CloudWatch.

Métrica	Descripción
BytesSent	Cantidad de bytes enviados al flujo de Firehose en el periodo especific ado. Unidades: bytes
RecordSen dAttempts	El número de registros que se ha intentado grabar (como primer intento o como repetición) en una llamada a PutRecordBatch durante el periodo de tiempo especificado. Unidades: recuento
RecordSen dErrors	El número de registros que han devuelto un estado de error en una llamada a PutRecordBatch , incluidos los intentos repetidos, durante el periodo de tiempo especificado. Unidades: recuento
ServiceErrors	El número de llamadas a PutRecordBatch que ocasionaron un error de servicio (distinto de un error de limitación controlada) durante el periodo especificado. Unidades: recuento

Registro de llamadas a la API en Amazon Data Firehose con AWS CloudTrail

Amazon Data Firehose se integra con AWS CloudTrail, un servicio que proporciona un registro de las acciones que lleva a cabo un usuario, rol o servicio de AWS en Amazon Data Firehose. CloudTrail captura todas las llamadas a la API de Amazon Data Firehose como eventos. Las llamadas capturadas incluyen las llamadas desde la consola de Amazon Data Firehose y las llamadas desde el código a las operaciones de la API de Amazon Data Firehose. Si crea un registro de seguimiento, puede habilitar la entrega continua de eventos de CloudTrail en un bucket de Amazon S3, incluidos los eventos de Amazon Data Firehose Si no configura un registro de seguimiento, puede ver los eventos más recientes en la consola de CloudTrail en el Historial de eventos. Mediante la información recopilada por CloudTrail, puede determinar la solicitud que se realizó a Amazon Data

Firehose, la dirección IP de origen desde la que se realizó la solicitud, quién y cuándo se realizó, y otros detalles adicionales.

Para obtener más información acerca de CloudTrail, incluso cómo configurarlo y habilitarlo, consulte la Guía del usuario de AWS CloudTrail.

Información de Firehose en CloudTrail

CloudTrail se habilita en su cuenta de AWS cuando la crea. Cuando se produce una actividad de eventos admitida en Amazon Data Firehose, la actividad se registra en un evento de CloudTrail junto con otros eventos de servicios de AWS en Historial de eventos. Puede ver, buscar y descargar los últimos eventos de la cuenta de AWS. Para obtener más información, consulte Visualización de eventos con el historial de eventos de CloudTrail.

Para mantener un registro continuo de los eventos de la cuenta de AWS, incluidos los eventos de Amazon Data Firehose, cree un registro de seguimiento. Un registro de seguimiento permite a CloudTrail enviar archivos de registro a un bucket de Amazon S3. De manera predeterminada, cuando se crea un registro de seguimiento en la consola, el registro de seguimiento se aplica a todas las regiones de AWS. El registro de seguimiento registra los eventos de todas las regiones de la partición de AWS y envía los archivos de registro al bucket de Amazon S3 especificado. También es posible configurar otros servicios de AWS para analizar en profundidad y actuar en función de los datos de eventos recopilados en los registros de CloudTrail. Para más información, consulte los siguientes temas:

- Introducción a la creación de registros de seguimiento
- Servicios e integraciones compatibles con CloudTrail
- Configuración de notificaciones de Amazon SNS para CloudTrail
- Recepción de archivos de registro de CloudTrail de varias regiones y Recepción de archivos de registro de CloudTrail de varias cuentas

Amazon Data Firehose admite el registro de las siguientes acciones como eventos en archivos de registros de CloudTrail:

- CreateDeliveryStream
- DeleteDeliveryStream
- DescribeDeliveryStream
- ListDeliveryStreams

- ListTagsForDeliveryStream
- TagDeliveryStream
- StartDeliveryStreamEncryption
- StopDeliveryStreamEncryption
- UntagDeliveryStream
- UpdateDestination

Cada entrada de registro o evento contiene información sobre quién generó la solicitud. La información de identidad del usuario lo ayuda a determinar lo siguiente:

- Si la solicitud se realizó con credenciales de usuario de AWS Identity and Access Management (IAM) o credenciales de usuario raíz.
- Si la solicitud se realizó con credenciales de seguridad temporales de un rol o fue un usuario federado.
- Si la solicitud la realizó otro servicio de AWS.

Para obtener más información, consulte el Elemento userIdentity de CloudTrail.

Ejemplo: entradas de archivos de registro de Firehose

Un registro de seguimiento es una configuración que permite la entrega de eventos como archivos de registros en un bucket de Amazon S3 que especifique. Los archivos de registro de CloudTrail pueden contener una o varias entradas de registro. Un evento representa una solicitud específica realizada desde un origen cualquiera y contiene información sobre la acción solicitada, la fecha y la hora de la acción, los parámetros de la solicitud, etc. Los archivos de registro de CloudTrail no rastrean el orden en la pila de las llamadas públicas a la API, por lo que estas no aparecen en ningún orden específico.

En el ejemplo siguiente, se muestra una entrada de registro de CloudTrail que ilustra las acciones `CreateDeliveryStream`, `DescribeDeliveryStream`, `ListDeliveryStreams`, `UpdateDestination` y `DeleteDeliveryStream`.

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

Ejemplos de código para el uso de Firehose AWS SDKs

Los siguientes ejemplos de código muestran cómo usar Firehose con un kit de desarrollo de AWS software (SDK).

Las acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Mientras las acciones muestran cómo llamar a las distintas funciones de servicio, es posible ver las acciones en contexto en los escenarios relacionados.

Los escenarios son ejemplos de código que muestran cómo llevar a cabo una tarea específica a través de llamadas a varias funciones dentro del servicio o combinado con otros Servicios de AWS.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consultaUso de Firehose con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Ejemplos de código

- Ejemplos básicos de uso de Firehose AWS SDKs
  - Acciones para el uso de Firehose AWS SDKs
    - Úselo PutRecord con un AWS SDK o CLI
    - Úselo PutRecordBatch con un AWS SDK o CLI
- Escenarios para el uso de Firehose AWS SDKs
  - Utilice Amazon Data Firehose para procesar registros individuales y por lotes

Ejemplos básicos de uso de Firehose AWS SDKs

Los siguientes ejemplos de código muestran cómo utilizar los conceptos básicos de Amazon Data Firehose con. AWS SDKs

Ejemplos

- Acciones para el uso de Firehose AWS SDKs
  - Úselo PutRecord con un AWS SDK o CLI
  - Úselo PutRecordBatch con un AWS SDK o CLI

Acciones para el uso de Firehose AWS SDKs

Los siguientes ejemplos de código muestran cómo realizar acciones individuales de Firehose con. AWS SDKs Cada ejemplo incluye un enlace a GitHub, donde puede encontrar instrucciones para configurar y ejecutar el código.

Estos fragmentos llaman a la API de Firehose y son fragmentos de código de programas más grandes que deben ejecutarse en contexto. Puede ver las acciones en contexto en Escenarios para el uso de Firehose AWS SDKs .

Los siguientes ejemplos incluyen solo las acciones que se utilizan con mayor frecuencia. Para ver una lista completa, consulte la Referencia de la API de Amazon Data Firehose.

Ejemplos

- Úselo PutRecord con un AWS SDK o CLI
- Úselo PutRecordBatch con un AWS SDK o CLI

Úselo **`PutRecord`** con un AWS SDK o CLI

Los siguientes ejemplos de código muestran cómo utilizar `PutRecord`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Colocar los registros en Firehose

CLI

AWS CLI

Escritura de un registro en un flujo

En el siguiente ejemplo de `put-record`, se escriben datos en un flujo. Los datos se codifican en formato Base64.

```
aws firehose put-record \
    --delivery-stream-name my-stream \
    --record '{"Data":"SGVsbG8gd29ybGQ="}'
```

Salida:

```
{
    "RecordId": "RjB5K/nnoGFHqwTsZlNd/
TTqvjE8V5dsyXZTQn2JXrdpMTOwssyEb6nfC8fwf1whhwnItt4mvrn+gsqeK5jB7QjuLg283+Ps4Sz/
j1Xujv31iDhnPdaLw4BOyM9Amv7PcCuB2079RuM0NhoakbyUymlwY8yt20G8X2420wu1jlFafhci4erAt7QhDEvpw
    "Encrypted": false
}
```

Para obtener más información, consulte Envío de una secuencia de entrega de Amazon Kinesis Data Firehose en la Guía para desarrolladores de Amazon Kinesis Data Firehose.

- Para obtener más información sobre la API, consulte PutRecordla Referencia de AWS CLI comandos.

Java

SDK para Java 2.x

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   /**
    * 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 obtener más información sobre la API, consulta PutRecordla Referencia AWS SDK for Java 2.x de la API.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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 obtener más información sobre la API, consulta PutRecordla AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Uso de Firehose con un SDK de AWS En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Úselo **PutRecordBatch** con un AWS SDK o CLI

Los siguientes ejemplos de código muestran cómo utilizar `PutRecordBatch`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Colocar los registros en Firehose

CLI

AWS CLI

Para escribir varios registros en un flujo

En el siguiente ejemplo `put-record-batch`, se escriben tres registros en una secuencia. Los datos se codifican en formato Base64.

```
aws firehose put-record-batch \
    --delivery-stream-name my-stream \
    --records file://records.json
```

Contenido de `myfile.json`:

```
[
```

```
    {"Data": "Rmlyc3QgdGhpbmc="},
    {"Data": "U2Vjb25kIHRoaW5n"},
    {"Data": "VGhpcmQgdGhpbmc="}
]
```

Salida:

```
{
    "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 obtener más información, consulte Envío de una secuencia de entrega de Amazon Kinesis Data Firehose en la Guía para desarrolladores de Amazon Kinesis Data Firehose.

- Para obtener más información sobre la API, consulte PutRecordBatchla Referencia de AWS CLI comandos.

Java

SDK para Java 2.x

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   /**
    * 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 obtener más información sobre la API, consulta PutRecordBatchla Referencia AWS SDK for Java 2.x de la API.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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 obtener más información sobre la API, consulta PutRecordBatchla AWS Referencia de API de SDK for Python (Boto3).

Rust

SDK para Rust

> **Note**
>
> Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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 obtener más información sobre la API, consulta PutRecordBatchla referencia sobre la API de AWS SDK para Rust.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulteUso de Firehose con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Escenarios para el uso de Firehose AWS SDKs

Los siguientes ejemplos de código muestran cómo implementar escenarios comunes en Firehose con. AWS SDKs Estas situaciones muestran cómo llevar a cabo tareas específicas con llamadas a varias funciones dentro de Firehose o en combinación con otros Servicios de AWS. En cada escenario se incluye un enlace al código fuente completo, con instrucciones de configuración y ejecución del código.

Los escenarios requieren un nivel intermedio de experiencia para ayudarlo a entender las acciones de servicio en su contexto.

Ejemplos

- Utilice Amazon Data Firehose para procesar registros individuales y por lotes

Utilice Amazon Data Firehose para procesar registros individuales y por lotes

Los siguientes ejemplos de código muestran cómo usar Firehose para procesar registros individuales y por lotes.

Java

SDK para Java 2.x

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Este ejemplo coloca los registros individuales y por lotes en 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 obtener detalles sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK for Java 2.x .
  - PutRecord
  - PutRecordBatch

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Este script coloca los registros individuales y por lotes en 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 archivo contiene la configuración del script anterior.

```
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 obtener información sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK para Python (Boto3).
  - PutRecord
  - PutRecordBatch

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulteUso de Firehose con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Solución de problemas en Amazon Data Firehose

Si Firehose encuentra errores al entregar o procesar datos, vuelve a intentarlo hasta que venza la duración de reintentos configurada. Si la duración de reintentos finaliza antes de que los datos se entreguen correctamente, Firehose realiza una copia de seguridad de los datos en el bucket de copia de seguridad de S3 configurado. Si el destino es Amazon S3 y no se puede realizar la entrega o si se produce un error en la entrega en el bucket de S3 de copias de seguridad, Firehose sigue intentándolo hasta que finaliza el periodo de retención.

Para obtener información sobre el seguimiento de los errores de entrega utilizados CloudWatch, consultethe section called “Supervisión con Registros de CloudWatch”.

Direct PUT

En el caso de los flujos de Firehose de `DirectPut`, Firehose conserva los registros durante 24 horas. Para un flujo de Firehose cuyo origen de datos sea un flujo de datos de Kinesis, puede cambiar el periodo de retención como se describe en Cambiar el periodo de retención de datos. En este caso, Firehose vuelve a intentar las siguientes operaciones indefinidamente: `DescribeStream`, `GetRecords` y `GetShardIterator`.

Si el flujo de Firehose utiliza `DirectPut`, compruebe las métricas `IncomingBytes` y `IncomingRecords` para ver si hay tráfico entrante. Si utiliza `PutRecord` o `PutRecordBatch`, asegúrese de que detecta las excepciones y vuelva a intentarlo. Le recomendamos que utilice una política de reintentos con retardo exponencial con fluctuaciones y varios reintentos. Además, si utilizas la `PutRecordBatch` API, asegúrate de que el código compruebe el valor de FailedPutCounten la respuesta incluso cuando la llamada a la API se realice correctamente.

Kinesis Data Stream

Si el flujo de Firehose utiliza un flujo de datos de Kinesis como origen, compruebe las métricas `IncomingBytes` y `IncomingRecords` para el flujo de datos de origen. Además, asegúrese de que se emiten las métricas `DataReadFromKinesisStream.Bytes` y `DataReadFromKinesisStream.Records` para el flujo de Firehose.

Problemas comunes

A continuación, se ofrecen algunas sugerencias para solucionar problemas comunes al trabajar con un flujo de Firehose.

flujo de Firehose no disponible

La transmisión de Firehose no está disponible como destino para CloudWatch registros, CloudWatch eventos o acciones de AWS IoT, ya que algunos AWS servicios solo pueden enviar mensajes y eventos a una transmisión de Firehose que se encuentre en la misma. Región de AWS Compruebe que el flujo de Firehose está emplazado en la misma región que los demás servicios.

Sin datos en el destino

Si no hay problemas de ingesta de datos, y las métricas emitidas para el flujo de Firehose son correctas, pero no ve datos en el destino, compruebe la lógica del lector. Asegúrese de que su lector esté analizando correctamente todos los datos.

Métrica de antigüedad de los datos incrementada o no emitida

La antigüedad de los datos es una medida de lo actuales que son sus datos dentro del flujo de Firehose. Es la antigüedad del registro de datos más antiguo del flujo de Firehose, medida desde el momento en que Firehose ingirió los datos hasta el momento actual. Firehose proporciona métricas que puede utilizar para supervisar la actualización de los datos. Para identificar la métrica de antigüedad de los datos para un destino determinado, consulte the section called “Supervisión con métricas de CloudWatch”.

Si habilita la copia de seguridad para todos los eventos o todos los documentos, monitorice dos métricas de antigüedad de los datos distintas: una para el destino principal y otra para la copia de seguridad.

Si no se emite la métrica de antigüedad de los datos, esto significa que no hay ninguna entrega activa para el flujo de Firehose. Esto sucede cuando la entrega de datos está completamente bloqueada o cuando no hay datos entrantes.

Si la métrica de antigüedad de los datos aumenta constantemente, esto significa que la entrega de los datos está retrasada. Esto puede suceder por una de las siguientes razones.

- El destino no es capaz de soportar la velocidad de entrega. Si Firehose encuentra errores transitorios debido a un tráfico elevado, la entrega podría retrasarse. Esto puede ocurrir en destinos distintos de Amazon S3 (puede ocurrir en OpenSearch Service, Amazon Redshift o Splunk). Asegúrese de que su destino tiene suficiente capacidad para tratar el tráfico entrante.
- El destino es lento. La entrega de datos puede retrasarse si Firehose encuentra una latencia alta. Monitorice la métrica de latencia del destino.

- La función de Lambda es lenta. Esto podría dar lugar a una velocidad de entrega de datos inferior a la velocidad de ingesta de datos del flujo de Firehose. Si es posible, mejore la eficiencia de la función de Lambda. Por ejemplo, si la función realiza operaciones de E/S de red, use varios subprocesos o utilice operaciones de E/S asincrónicas para aumentar el paralelismo. Además, considere la posibilidad de aumentar el tamaño de memoria de la función de Lambda para que la asignación de CPU pueda aumentar en consecuencia. Esto podría producir invocaciones de Lambda más rápidas. Para obtener información sobre la configuración de las funciones de Lambda, consulte Configuración de funciones de Lambda AWS.
- Hay errores durante la entrega de datos. Para obtener información sobre cómo supervisar los errores mediante Amazon CloudWatch Logs, consultethe section called “Supervisión con Registros de CloudWatch”.
- Si el origen de datos del flujo de Firehose es un flujo de datos de Kinesis, es posible que se aplique una limitación. Compruebe las métricas `ThrottledGetRecords`, `ThrottledGetShardIterator` y `ThrottledDescribeStream`. Si hay varios consumidores asociados a la secuencia de datos de Kinesis, tenga en cuenta lo siguiente:
  - Si las métricas `ThrottledGetShardIterator` y `ThrottledGetRecords` son elevadas, le recomendamos que aumente el número de particiones aprovisionadas para la secuencia de datos.
  - Si el `ThrottledDescribeStream` valor es alto, le recomendamos que añada el `kinesis:listshards` permiso al rol configurado en KinesisStreamSourceConfiguration.
- Sugerencias de almacenamiento en búfer bajo para el destino. Esto podría aumentar el número de viajes de ida y vuelta que Firehose tiene que hacer al destino, lo que podría causar un retraso en la entrega. Considere la posibilidad de aumentar el valor de las sugerencias de almacenamiento en búfer. Para obtener más información, consulte BufferingHints.
- Una duración de reintentos alta podría hacer que la entrega se retrasara cuando los errores son frecuentes. Considere la posibilidad de reducir la duración de los reintentos. Además, monitorice los errores e intente reducirlos. Para obtener información sobre cómo supervisar los errores mediante Amazon CloudWatch Logs, consultethe section called “Supervisión con Registros de CloudWatch”.
- Si el destino es Splunk y la métrica `DeliveryToSplunk.DataFreshness` es elevada pero `DeliveryToSplunk.Success` tiene un valor correcto, es posible que el clúster de Splunk esté ocupado. Libere el clúster de Splunk si es posible. También puede ponerse en contacto con AWS Support y solicitar un aumento del número de canales que Firehose utiliza para comunicarse con el clúster de Splunk.

Error al convertir el formato de registro a Apache Parquet

Esto ocurre si toma datos de DynamoDB que incluyen ese tipo, `Set` los transmite a través de Lambda a una transmisión de Firehose y utiliza AWS Glue Data Catalog an para convertir el formato de registro a Apache Parquet.

Cuando el AWS Glue rastreador indexa los tipos de datos del conjunto de DynamoDB (`StringSet`, y`BinarySet`)`NumberSet`, los almacena en el catálogo de datos como`SET<STRING>`, y, respectivamente. `SET<BIGINT>` `SET<BINARY>` Sin embargo, para que Firehose convierta los registros de datos al formato Apache Parquet, requiere los tipos de datos de Apache Hive. Dado que los tipos de conjunto no son tipos de datos de Apache Hive válidos, la conversión produce un error. Para que la conversión funcione, actualice el catálogo de datos con los tipos de datos de Apache Hive. Puede hacerlo cambiando `set` a `array` en el catálogo de datos.

Para cambiar uno o más tipos de datos de a dentro de un catálogo **`set`** de **`array`** datos AWS Glue

1. Inicie sesión en Consola de administración de AWS y abra la AWS Glue consola en https://console.aws.amazon.com/glue/.
2. En el panel izquierdo, en el encabezado Catálogo de datos , elija Tablas.
3. En la lista de tablas, elija el nombre de la tabla en la que desea modificar uno o varios tipos de datos. Esto le lleva a la página de detalles de la tabla.
4. Elija el botón Editar esquema situado en la esquina superior derecha de la página de detalles.
5. En la columna Tipo de datos , elija el primer tipo de datos `set` .
6. En la lista desplegable Tipo de columna , cambie el tipo de `set` a `array`.
7. En el ArraySchemacampo, introduzca `array<string>`array<int>, `oarray<binary>`, según el tipo de datos apropiado para su escenario.
8. Elija Actualizar.
9. Repita los pasos anteriores para convertir otros tipos `set` a tipos `array` .
10. Seleccione Save.

Faltan campos para el objeto transformado para Lambda

Al utilizar la transformación de datos de Lambda para cambiar los datos JSON a un objeto Parquet, es posible que falten algunos campos después de la transformación. Esto ocurre si el objeto JSON tiene letras mayúsculas, y la distinción entre mayúsculas y minúsculas está configurada en `false`,

lo que puede provocar una discordancia en las claves JSON tras la transformación de los datos y provocar que falten datos en el objeto Parquet resultante del bucket s3.

Para solucionar este problema, asegúrese de que la configuración del conducto tenga `deserializationOption: case.insensitive` configurado en `true` para que las claves JSON coincidan después de la transformación.

Solución de problemas de Amazon S3

Compruebe lo siguiente si los datos no se entregan en su bucket de Amazon Simple Storage Service (Amazon S3).

- Compruebe las métricas `IncomingBytes` y `IncomingRecords` de Firehose para asegurarse de que los datos se envían correctamente al flujo de Firehose. Para obtener más información, consulte Supervisión de Amazon Data Firehose con métricas de CloudWatch.
- Si la transformación de datos con Lambda está habilitada, compruebe la métrica `ExecuteProcessingSuccess` de Firehose para asegurarse de que ha intentado invocar la función de Lambda. Para obtener más información, consulte Supervisión de Amazon Data Firehose con métricas de CloudWatch.
- Compruebe la métrica `DeliveryToS3.Success` de Firehose para asegurarse de que ha intentado poner los datos en su bucket de Amazon S3. Para obtener más información, consulte Supervisión de Amazon Data Firehose con métricas de CloudWatch.
- Habilite el registro de errores si aún no está habilitado y busque errores de entrega en los logs de errores. Para obtener más información, consulte Supervisión de Amazon Data Firehose mediante Registros de CloudWatch.
- Si ves un mensaje de error en el registro que dice «Se encontró una Firehose InternalServerError al llamar al servicio Amazon S3». La operación se reintentará; si el error persiste, póngase en contacto con S3 para solucionarlo.”, podría deberse al aumento significativo de las tasas de solicitud en una sola partición de S3. Puede optimizar los patrones de diseño de los prefijos de S3 para mitigar el problema. Para obtener más información, consulte Patrones de diseño de prácticas recomendadas: optimización del rendimiento de Amazon S3. Si esto no resuelve el problema, ponte en contacto con AWS Support para obtener más ayuda.
- Asegúrese de que el bucket de Amazon S3 especificado en el flujo de Firehose aún exista.
- Si la transformación de datos con Lambda está habilitada, asegúrese de que la función de Lambda especificada en el flujo de Firehose aún exista.

- Asegúrese de que el rol de IAM especificado en el flujo de Firehose tenga acceso al bucket de S3 y a la función de Lambda (si la transformación de datos está habilitada). Además, asegúrese de que la función de IAM tenga acceso al grupo de CloudWatch registros y a los flujos de registros para comprobar los registros de errores. Para obtener más información, consulte Concesión de acceso a Firehose a un destino de Amazon S3.
- Si usa la transformación de datos, asegúrese de que la función de Lambda nunca devuelva respuestas cuyo tamaño de carga sea superior a 6 MB. Para obtener más información, consulte Amazon Data FirehoseData Transformation.

Solución de problemas de Amazon Redshift

Compruebe lo siguiente si los datos no se entregan en el clúster aprovisionado de Amazon Redshift o el grupo de trabajo de Amazon Redshift sin servidor.

Los datos se entregan en el bucket de S3 antes de cargarse en Amazon Redshift. Si los datos no se han entregado en el bucket de S3, consulte Solución de problemas de Amazon S3.

- Compruebe la métrica `DeliveryToRedshift.Success` de Firehose para asegurarse de que ha intentado copiar los datos del bucket de S3 en el clúster aprovisionado de Amazon Redshift o el grupo de trabajo Amazon Redshift sin servidor. Para obtener más información, consulte Supervisión de Amazon Data Firehose con métricas de CloudWatch.
- Habilite el registro de errores si aún no está habilitado y busque errores de entrega en los logs de errores. Para obtener más información, consulte Supervisión de Amazon Data Firehose mediante Registros de CloudWatch.
- Consulte la tabla `STL_CONNECTION_LOG` de Amazon Redshift para ver si Firehose puede establecer conexiones correctas. En esta tabla, debería poder ver las conexiones y su estado por nombre de usuario. Para obtener más información, consulte `STL_CONNECTION_LOG` en la Guía para desarrolladores de bases de datos de Amazon Redshift.
- Si al consultar la tabla anterior observa que se están estableciendo conexiones, consulte la tabla `STL_LOAD_ERRORS` de Amazon Redshift para dar con el motivo del error de COPY. Para obtener más información, consulte `STL_LOAD_ERRORS` en la Guía para desarrolladores de bases de datos de Amazon Redshift.
- Asegúrese de que la configuración de Amazon Redshift en el flujo Firehose sea correcta y válida.
- Asegúrese de que el rol de IAM que se ha especificado en su flujo de Firehose pueda acceder al bucket de S3 desde el que Amazon Redshift copia los datos y también a la función de Lambda de

transformación de datos (si la transformación de datos está habilitada). Además, asegúrese de que la función de IAM tenga acceso a los grupos de CloudWatch registros y a los flujos de registros para comprobar los registros de errores. Para obtener más información, consulte Concesión a Firehose de acceso a un destino de Amazon Redshift .

- Si el clúster aprovisionado de Amazon Redshift o el grupo de trabajo Amazon Redshift sin servidor se encuentra en una nube privada virtual (VPC), asegúrese de que el clúster permita el acceso desde las direcciones IP de Firehose. Para obtener más información, consulte Concesión a Firehose de acceso a un destino de Amazon Redshift .
- Asegúrese de que el clúster aprovisionado de Amazon Redshift o el grupo de trabajo de Amazon Redshift sin servidor esté disponible públicamente.
- Si usa la transformación de datos, asegúrese de que la función de Lambda nunca devuelva respuestas cuyo tamaño de carga sea superior a 6 MB. Para obtener más información, consulte Amazon Data FirehoseData Transformation.

Solución de problemas de Amazon OpenSearch Service

Comprueba lo siguiente si los datos no se envían a tu dominio OpenSearch de servicio.

Se pueden crear copias de seguridad de los datos en su bucket de Amazon S3 simultáneamente. Si los datos no se han entregado al bucket de S3, consulte Solución de problemas de Amazon S3.

- Compruebe las métricas `IncomingBytes` y `IncomingRecords` de Firehose para asegurarse de que los datos se envían correctamente al flujo de Firehose. Para obtener más información, consulte Supervisión de Amazon Data Firehose con métricas de CloudWatch.
- Si la transformación de datos con Lambda está habilitada, compruebe la métrica `ExecuteProcessingSuccess` de Firehose para asegurarse de que ha intentado invocar la función de Lambda. Para obtener más información, consulte Supervisión de Amazon Data Firehose con métricas de CloudWatch.
- Compruebe la `DeliveryToAmazonOpenSearchService.Success` métrica Firehose para asegurarse de que Firehose ha intentado indexar los datos en el clúster de servicio. OpenSearch Para obtener más información, consulte Supervisión de Amazon Data Firehose con métricas de CloudWatch.
- Habilite el registro de errores si aún no está habilitado y busque errores de entrega en los logs de errores. Para obtener más información, consulte Supervisión de Amazon Data Firehose mediante Registros de CloudWatch.

- Asegúrese de que la configuración del OpenSearch servicio de su transmisión de Firehose sea precisa y válida.
- Si la transformación de datos con Lambda está habilitada, asegúrese de que la función de Lambda especificada en el flujo de Firehose aún exista. Además, asegúrate de que la función de IAM tenga acceso a los grupos de CloudWatch registros y a los flujos de registros para comprobar los registros de errores. Para obtener más información, consulte Concesión FirehoseAccess a un destino de OpenSearch servicio público.
- Asegúrese de que la función de IAM que se especifica en la transmisión de Firehose pueda acceder al OpenSearch clúster de servicios, al depósito de backup de S3 y a la función Lambda (si la transformación de datos está habilitada). Además, asegúrate de que la función de IAM tenga acceso a los grupos de registros y a los flujos de CloudWatch registros para comprobar los registros de errores. Para obtener más información, consulte Conceda a Firehose acceso a un destino de servicio público OpenSearch .
- Si usa la transformación de datos, asegúrese de que la función de Lambda nunca devuelva respuestas cuyo tamaño de carga sea superior a 6 MB. Para obtener más información, consulte Amazon Data FirehoseData Transformation.
- Amazon Data Firehose no admite actualmente la entrega de registros CloudWatch al OpenSearch destino de Amazon Service porque Amazon CloudWatch combina varios eventos de registro en un registro de Firehose y OpenSearch Amazon Service no puede aceptar varios eventos de registro en un registro. Como alternativa, puedes considerar usar un filtro de suscripción para Amazon OpenSearch Service in CloudWatch Logs.

Solución de problemas de Splunk

Realice las siguientes comprobaciones si los datos no se entregan a su punto de enlace de Splunk.

- Si la plataforma Splunk está en una VPC, asegúrese de que Firehose tenga acceso a ella. Para obtener más información, consulte Acceso a Splunk en VPC.
- Si utilizas un balanceador de AWS cargas, asegúrate de que sea un Classic Load Balancer o un Application Load Balancer. Además, habilite las sesiones persistentes basadas en la duración con la caducidad de las cookies deshabilitada para el equilibrador de carga clásico y la caducidad establecida en el máximo (7 días) para el equilibrador de carga de aplicación. Para obtener información sobre cómo hacerlo, consulte Persistencia de la sesión basada en la duración para el equilibrador de carga clásico o el equilibrador de carga de aplicación.

- Revise los requisitos de la plataforma Splunk. El complemento de Splunk para Firehose requiere la versión 6.6.X o posterior de la plataforma Splunk. Para obtener más información, consulte Splunk Add-on for Amazon Kinesis Firehose.
- Si tiene un proxy (Elastic Load Balancing u otro) entre Firehose y el nodo HTTP Event Collector (HEC), habilite las sesiones fijas para admitir los reconocimientos de HEC (). ACKs
- Asegúrese de utilizar un token de HEC válido.
- Asegúrese de que el token de HEC esté habilitado.
- Compruebe que los datos que está enviando a Splunk tienen el formato correcto. Para obtener más información, consulte Format events for HTTP Event Collector.
- Asegúrese de que el token de HEC y el evento de entrada estén configurados con un índice válido.
- Cuando una carga en Splunk falla debido a un error del servidor desde el nodo del HEC, la solicitud vuelve a enviarse automáticamente. Si fallan todos los reintentos, se crea una copia de seguridad de los datos en Amazon S3. Compruebe si los datos aparecen en Amazon S3; esa es una indicación de este tipo de error.
- Asegúrese de que ha activado la confirmación de indexadores en el token de HEC.
- Aumente el valor de `HECAcknowledgmentTimeoutInSeconds` en la configuración de destino de Splunk de su flujo de Firehose.
- Aumente el valor de `DurationInSeconds` en `RetryOptions` en la configuración de destino de Splunk de su flujo de Firehose.
- Compruebe el estado del HEC.
- Si usa la transformación de datos, asegúrese de que la función de Lambda nunca devuelva respuestas cuyo tamaño de carga sea superior a 6 MB. Para obtener más información, consulte Transformación de datos de Amazon Data Firehose.
- Asegúrese de que el parámetro Splunk denominado `ackIdleCleanup` se establece en `true`. Su valor predeterminado es false. Para establecer este parámetro en `true`, haga lo siguiente:
  - Para una implementación de Splunk Cloud administrada, envíe un caso utilizando el portal de soporte de Splunk. En este caso, pida al soporte de Splunk que habilite el recopilador de eventos HTTP, establezca `ackIdleCleanup` en `true` en `inputs.conf` y cree o modifique un balanceador de carga para usarlo con este complemento.
  - Para efectuar una implementación distribuida de Splunk Enterprise, establezca el parámetro `ackIdleCleanup` en true en el archivo `inputs.conf`. Para los usuarios de *nix, este archivo se encuentra en `$SPLUNK_HOME/etc/apps/splunk_httpinput/local/`. Para los usuarios de Windows, se encuentra en `%SPLUNK_HOME%\etc\apps\splunk_httpinput\local\`.

  - Para efectuar una implementación de una sola instancia de Splunk Enterprise, establezca el parámetro `ackIdleCleanup` en `true` en el archivo `inputs.conf`. Para los usuarios de *nix, este archivo se encuentra en `$SPLUNK_HOME/etc/apps/splunk_httpinput/local/`. Para los usuarios de Windows, se encuentra en `%SPLUNK_HOME%\etc\apps\splunk_httpinput\local\`.
- Asegúrese de que el rol de IAM especificado en su flujo de Firehose tenga acceso al bucket de copias de seguridad de S3 y a la función de Lambda para la transformación de datos (si esta está habilitada). Además, asegúrate de que el rol de IAM tenga acceso al grupo de registros y a los flujos de CloudWatch registros para comprobar los registros de errores. Para obtener más información, consulte Concesión FirehoseAccess a un destino de Splunk.
- Para volver a enviar a Splunk los datos que se enviaron al bucket de errores de S3 (copia de seguridad de S3), siga los pasos que se mencionan en la documentación de Splunk.
- Consulte Troubleshoot the Splunk Add-on for Amazon Kinesis Firehose.

Solución de problemas de Snowflake

En esta sección, se describen los pasos comunes de solución de problemas al utilizar Snowflake como destino.

Error de creación del flujo de Firehose

Si se produce un error al crear una transmisión de Firehose para una transmisión que entrega datos a un clúster de Snowflake PrivateLink habilitado, esto indica que Firehose no puede acceder al VPCE-ID. Esto puede deberse a una de las siguientes razones:

- El VPCE-ID es incorrecto. Confirme que no haya errores tipográficos.
- Firehose no admite la versión preliminar de Snowflake sin región. URLs Proporcione la URL mediante el localizador de cuentas de Snowflake. Para obtener más información, consulte la documentación de Snowflake.
- Confirma que el arroyo Firehose se haya creado en la misma AWS región que la región de los copos de nieve.
- Si el problema persiste, ponte en contacto con el servicio de asistencia. AWS

Error de entrega

Realice las siguientes comprobaciones si los datos no se entregan a su tabla de Snowflake. Si falla la entrega de datos en Snowflake, estos se enviarán al bucket de errores de S3 junto con un código de error y un mensaje de error correspondientes a la carga útil. Los siguientes son algunas de las situaciones de error más comunes. Para ver la lista completa de códigos de error, consulte Errores de entrega de datos de Snowflake.

- Código de error: Snowflake. DefaultRoleMissing: Indica que la función Snowflake no está configurada al crear la transmisión de Firehose. Si el rol de Snowflake no está configurado, asegúrese de establecer un rol predeterminado para el usuario de Snowflake especificado.
- Código de error: Snowflake. ExtraColumns: Indica que se ha rechazado la inserción en Snowflake debido a la existencia de columnas adicionales en la carga útil de entrada. No se deben especificar las columnas que no están presentes en la tabla. Tenga en cuenta que los nombres de las columnas de Snowflake distinguen entre mayúsculas y minúsculas. Si la entrega no se realiza correctamente debido a este error a pesar de que la columna esté presente en la tabla, asegúrese de que las mayúsculas y minúsculas del nombre de la columna en la carga útil de entrada coincidan con el nombre de la columna indicado en la definición de la tabla.
- Código de error: Snowflake. MissingColumns: Indica que la inserción en Snowflake se ha rechazado porque faltan columnas en la carga útil de entrada. Asegúrese de que los valores estén especificados para todas las columnas que no admiten valores NULL.
- Código de error: Snowflake. InvalidInput: Esto puede ocurrir si Firehose no ha podido analizar la carga útil de entrada proporcionada en un formato JSON válido. Asegúrese de que la carga útil json esté bien formada y no tenga comillas dobles, comillas, caracteres de escape adicionales, etc. Actualmente, Firehose solo admite un elemento JSON como carga útil de registro; no se admiten matrices JSON.
- Código de error: Snowflake. InvalidValue: Indica que la entrega ha fallado debido a un tipo de datos incorrecto en la carga útil de entrada. Asegúrese de que los valores JSON especificados en la carga útil de entrada se ajusten al tipo de datos declarado en la definición de la tabla de Snowflake.
- Código de error: Snowflake. InvalidTableType: Indica que el tipo de tabla configurado en la transmisión Firehose no es compatible. Consulte las limitaciones (en la sección Limitaciones) de la transmisión por secuencias de snowpipe para conocer las tablas, las columnas y los tipos de datos compatibles.

> **Note**
>
> Por cualquier motivo, si la definición de la tabla o los permisos del rol se cambian en el destino de Snowflake después de crear el flujo de Firehose, Firehose puede tardar varios minutos en detectar esos cambios. Si ve errores de entrega debido a esto, intente eliminar y volver a crear el flujo de Firehose.

Solución de problemas de accesibilidad a los puntos de conexión de Firehose

Si se agota el tiempo de espera de la API de Firehose, realice los siguientes pasos para probar la accesibilidad a los puntos de conexión:

- Compruebe si las solicitudes de API se realizan desde un host de una VPC. Todo el tráfico de una VPC requiere la configuración de un punto de conexión de VPC de Firehose. Para obtener más información, consulte Uso de Firehose con. AWS PrivateLink
- Si el tráfico proviene de una red pública o una VPC con el punto de conexión de VPC de Firehose configurado en una subred determinada, ejecute los siguientes comandos desde el host para comprobar la conectividad de la red. El punto de conexión de Firehose se encuentra en Puntos de conexión y cuotas de Firehose.
  - Utilice herramientas, como traceroute o tcping, para comprobar si la configuración de la red es correcta. Si eso no funciona, compruebe la configuración de la red:

    Por ejemplo:

    ```
    traceroute firehose.us-east-2.amazonaws.com
    ```

    o

    ```
    tcping firehose.us-east-2.amazonaws.com 443
    ```

  - Si parece que la configuración de la red es correcta y se produce un error en el siguiente comando, compruebe si Amazon CA (Autoridad de certificación) está en la cadena de confianza.

    Por ejemplo:

```
curl firehose.us-east-2.amazonaws.com
```

Si los comandos anteriores se ejecutan correctamente, vuelva a probar la API para comprobar si devuelve una respuesta.

Solución de problemas de puntos de conexión HTTP

En esta sección se describen los pasos de solución de problemas habituales cuando Amazon Data Firehose entrega datos a destinos de puntos de enlace HTTP genéricos y a destinos de socios, como Datadog, Dynatrace, LogicMonitor MongoDB, New Relic, Splunk o Sumo Logic. A efectos de esta sección, todos los destinos aplicables se denominan puntos de conexión HTTP. Asegúrese de que el rol de IAM especificado en su flujo de Firehose tenga acceso al bucket de copias de seguridad de S3 y a la función de Lambda para la transformación de datos (si esta está habilitada). Además, asegúrese de que el rol de IAM tenga acceso a los grupos de registros y a las secuencias de registros para comprobar los registros de errores. CloudWatch Para obtener más información, consulte Concesión de acceso de Firehose a un destino de punto de conexión HTTP.

> **Note**
>
> La información de esta sección no se aplica a los siguientes destinos: Splunk, OpenSearch Service, S3 y Redshift.

CloudWatch Registros

Se recomienda encarecidamente activar el CloudWatch registro para. Los registros solo se publican cuando hay errores en la entrega en su destino.

Excepciones 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"
}
```

Las excepciones de destino indican que Firehose puede establecer una conexión con su punto de conexión y realizar una solicitud HTTP, pero no ha recibido un código de respuesta 200. Las respuestas 2xx que no sean 200 también generarán una excepción de destino. Amazon Data Firehose registra en Logs el código de respuesta y una carga útil de respuesta truncada recibida desde el punto de enlace configurado. CloudWatch Como Amazon Data Firehose registra el código de respuesta y la carga útil sin modificarlos ni interpretarlos, corresponde al punto de conexión indicar el motivo exacto por el que rechazó la solicitud de entrega HTTP de Amazon Data Firehose. A continuación se indican las recomendaciones de solución de problemas más comunes para estas excepciones:

- 400: indica que está enviando una solicitud incorrecta debido a una configuración incorrecta de Amazon Data Firehose. Asegúrese de tener la URL, los atributos comunes, la codificación del contenido, la clave de acceso y las sugerencias de almacenamiento en búfer correctos para su destino. Consulte la documentación específica del destino sobre la configuración requerida.
- 401: indica que la clave de acceso que configuró para el flujo de Firehose es incorrecta o falta.
- 403: indica que la clave de acceso que configuró para el flujo de Firehose no tiene permisos para entregar los datos en el punto de conexión configurado.
- 413: indica que la carga útil de la solicitud que Amazon Data Firehose envía al punto de conexión es demasiado grande para que este pueda gestionarla. Intente reducir la sugerencia de almacenamiento en búfer al tamaño recomendado para su destino.
- 429: indica que Amazon Data Firehose envía solicitudes a una velocidad superior a la que puede gestionar el destino. Afina la sugerencia de almacenamiento en búfer aumentando el tiempo and/or de almacenamiento en búfer y aumentando el tamaño del búfer (pero siempre dentro del límite de tu destino).
- 5xx: indica que hay un problema con el destino. El servicio Amazon Data Firehose sigue funcionando correctamente.

> **Important**
>
> Importante: Si bien estas son las recomendaciones habituales para la solución de problemas, es posible que los puntos de conexión específicos tengan diferentes motivos para proporcionar los códigos de respuesta, por lo que primero se deben seguir las recomendaciones específicas de los puntos de conexión.

Respuesta no válida

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"
}
```

Las excepciones de respuestas no válidas indican que Amazon Data Firehose recibió una respuesta no válida del punto de conexión de destino. La respuesta debe cumplir con las especificaciones de las respuestas; de lo contrario, Amazon Data Firehose considerará que el intento de entrega ha sido un error y volverá a entregar los mismos datos hasta que se supere la duración del reintento configurada. Amazon Data Firehose trata las respuestas que no siguen las especificaciones de la respuesta como errores, incluso si la respuesta tiene un estado 200. Si está desarrollando un punto de conexión compatible con Amazon Data Firehose, siga las especificaciones de las respuestas para garantizar que los datos se entreguen correctamente.

A continuación, se muestran algunos de los tipos comunes de respuestas no válidas y cómo solucionarlos:

- JSON no válido o campos inesperados: indica que la respuesta no se puede deserializar correctamente como JSON o que tiene campos inesperados. Asegúrese de que la respuesta no tenga el contenido codificado.
- Falta RequestId: indica que la respuesta no contiene un RequestID.
- RequestId no coincide: indica que el RequestID de la respuesta no coincide con el RequestID saliente.
- Falta la marca de tiempo: indica que la respuesta no contiene ningún campo de marca de tiempo. El campo de marca de tiempo debe ser un número y no una cadena.
- Falta el encabezado Content-Type: indica que la respuesta no contiene un encabezado “content-type: application/json”. No se acepta ningún otro content-type.

> **Important**
>
> Importante: Amazon Data Firehose solo puede entregar los datos en puntos de conexión que cumplan con las especificaciones de las solicitudes y respuestas de Firehose.
> Si está configurando su destino para un servicio de terceros, asegúrese de utilizar el punto de conexión correcto compatible con Amazon Data Firehose, que probablemente sea diferente del punto de conexión de ingesta público. Por ejemplo, el punto de conexión Amazon Data Firehose de Datadog es `https://aws-kinesis-http-intake.logs.datadoghq.com/`, mientras que su punto de conexión público es `https://api.datadoghq.com/`.

Otros errores habituales

A continuación se indican los códigos de error y las definiciones adicionales.

- Código de error:. HttpEndpoint RequestTimeout - Indica que el punto final tardó más de 3 minutos en responder. Si es el propietario del destino, reduzca el tiempo de respuesta del punto de conexión de destino. Si no es el propietario del destino, póngase en contacto con el propietario y pregúntele si se puede hacer algo para reducir el tiempo de respuesta (como reducir la sugerencia de almacenamiento en búfer para que se procesen menos datos por solicitud).
- Código de error: HttpEndpoint. ResponseTooLarge - Indica que la respuesta es demasiado grande. La respuesta debe ser inferior a 1 MiB, incluidos los encabezados.
- Código de error: HttpEndpoint. ConnectionFailed - Indica que no se ha podido establecer una conexión con el punto final configurado. Esto puede deberse a un error tipográfico en la URL

configurada, a que Amazon Data Firehose no puede acceder al punto de conexión o a que el punto de conexión tarda demasiado en responder a la solicitud de conexión.

- Código de error: HttpEndpoint. ConnectionReset - Indica que se estableció una conexión pero el punto final la restableció o cerró prematuramente.
- Código de error:. HttpEndpoint SSLHandshakeError: indica que un protocolo de enlace SSL no se pudo completar correctamente con el punto final configurado.

Solución de problemas de MSK como origen

En esta sección se describen los pasos comunes de solución de problemas al utilizar MSK como origen.

> **Note**
>
> Para solucionar problemas de procesamiento, transformación o entrega de S3, consulte las secciones anteriores.

Error de creación de conductos

Compruebe lo siguiente si el conducto con MSK como origen no se crea correctamente:

- Compruebe que el clúster de MSK de origen se encuentre en estado activo.
- Si utiliza la conectividad privada, asegúrese de que el enlace privado en el clúster esté activado.

  Si utiliza la conectividad pública, asegúrese de que el acceso público en el clúster esté activado.
- Si utiliza la conectividad privada, asegúrese de agregar una política basada en recursos que permita a Firehose crear un enlace privado. Consulte también: Permisos entre cuentas de MSK.
- Asegúrese de que el rol de la configuración de origen tenga permiso para ingerir datos del tema del clúster.
- Asegúrese de que los grupos de seguridad de la VPC permitan el tráfico de entrada en los puertos que utilizan los servidores de arranque del clúster.

Conducto suspendido

Compruebe lo siguiente si el conducto se encuentra en estado SUSPENDIDO:

- Compruebe que el clúster de MSK de origen se encuentre en estado activo.
- Compruebe que el tema de origen existe. En caso de que el tema se haya eliminado y vuelto a crear, tendrá que eliminar y volver a crear también el flujo de Firehose.

Conducto contrapresurizado

El valor de DataReadFromSource .Backpressured será 1 si se supera BytesPerSecondLimit cada partición o si el flujo normal de entrega es lento o se detiene.

- Si está acertando BytesPerSecondLimit , compruebe la métrica de DataReadFromSource .Bytes y solicite un aumento del límite.
- Compruebe los CloudWatch registros, las métricas de destino, las métricas de transformación de datos y las métricas de conversión de formato para identificar los cuellos de botella.

Actualización incorrecta de los datos

La actualización de los datos parece incorrecta.

- Firehose calcula la actualización de los datos en función de la marca de tiempo del registro consumido. Para garantizar que esta marca de tiempo se registre correctamente cuando el registro del productor se conserva en los registros del agente de Kafka, defina la configuración del tipo de marca de tiempo del tema de Kafka para que sea `message.timestamp.type=LogAppendTime`.

Problemas de conexión de clústeres de MSK

El siguiente procedimiento explica cómo se puede validar la conectividad con los clústeres de MSK. Para obtener más información sobre cómo configurar un cliente de Amazon MSK, consulte Empezar a utilizar Amazon MSK en la Guía para desarrolladores de Amazon Managed Streaming para Apache Kafka.

Validación de la conectividad con los clústeres de MSK

1. Cree una instancia de AL2 Amazon EC2 basada en Unix (preferiblemente). Si solo tienes habilitada la conectividad de VPC en tu clúster, asegúrate de que la EC2 instancia se ejecute en la misma VPC. Utilice SSH en la instancia una vez que esté disponible. Para obtener más información, consulta este tutorial en la Guía del EC2 usuario de Amazon.

2. Instale Java con el administrador de paquete Yum con el siguiente comando. Para obtener más información, consulte las instrucciones de instalación en la Guía del usuario de Amazon Corretto 8.

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

3. Instale el cliente de AWS ejecutando el siguiente comando:

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

4. Descargue la versión 2.6* de Apache Kafka del cliente ejecutando el siguiente comando.

```
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. Vaya al directorio `kafka_2.12-2.6.2/libs` y ejecute el siguiente comando para descargar el archivo JAR de IAM de 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. Cree el archivo `client.properties` en la carpeta bin de Kafka.

7. Sustituya `awsRoleArn` por el ARN del rol que utilizó en `SourceConfiguration` de Firehose y verifique la ubicación del certificado. Permita que su usuario AWS cliente asuma el rol`awsRoleArn`. AWS el usuario cliente intentará asumir el rol que especificó aquí.

```
[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. Ejecute el siguiente comando de Kafka para enumerar temas. Si su conexión es pública, utilice los servidores de arranque de puntos de conexión públicos. Si su conexión es privada, utilice los servidores de arranque de puntos de conexión privados.

```
bin/kafka-topics.sh --list --bootstrap-server <bootstrap servers> --command-config
 bin/client.properties
```

Si la solicitud tiene éxito, debería ver un resultado similar al del siguiente ejemplo.

```
[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. Si tiene problemas al ejecutar el script anterior, compruebe que los servidores de arranque que proporcionó estén accesibles en el puerto especificado. Para ello, puede descargar y utilizar telnet o una utilidad similar, como se muestra en el siguiente comando.

```
sudo yum install telnet
telnet <bootstrap servers><port>
```

Si la solicitud se realiza correctamente, obtendrá el siguiente resultado. Esto significa que puede conectarse al clúster de MSK dentro de la VPC local y que los servidores de arranque funcionan correctamente en el puerto especificado.

```
Connected to ..
```

10. Si la solicitud no se realiza correctamente, compruebe las reglas de entrada del grupo de seguridad de la VPC. Por ejemplo, podría utilizar las propiedades siguientes en la regla de entrada.

    ```
    Type: All traffic
    Port: Port used by the bootstrap server (e.g. 14001)
    Source: 0.0.0.0/0
    ```

    Vuelva a intentar la conexión telnet como se muestra en el paso anterior. Si sigue sin poder conectarse o la conexión de Firehose sigue fallando, comuníquese con el servicio de asistencia de AWS.

Cuota de Amazon Data Firehose

En esta sección, se describen las cuotas actuales (anteriormente se denominaban límites) de Amazon Data Firehose. Cada una de las cuotas se aplica a una sola región, a no ser que se especifique otra cosa.

La consola Service Quotas es una ubicación central donde puede ver y administrar las cuotas de AWS los servicios y solicitar un aumento de cuota para muchos de los recursos que utiliza. Utilice la información sobre cuotas que le proporcionamos para administrar su AWS infraestructura. Planifique la solicitud del aumento de las cuotas antes del momento en que lo necesite.

Para obtener más información, consulte Puntos de conexión y cuotas de Amazon Data Firehose en Referencia general de Amazon Web Services.

En la siguiente sección, se muestra que Amazon Data Firehose tiene la siguiente cuota.

- Con Amazon MSK como fuente de la transmisión Firehose, cada transmisión Firehose tiene una cuota predeterminada del 10% de rendimiento de lectura por partición y un MB/sec tamaño de registro máximo de 10 MB.
- Con Amazon MSK como fuente de la transmisión Firehose, hay un tamaño de registro máximo de 6 MB si AWS Lambda está habilitada y un tamaño de registro máximo de 10 MB si Lambda está deshabilitada. AWS Lambda limita su registro entrante a 6 MB y Amazon Data Firehose reenvía los registros de más de 6 MB a un bucket de error de S3. Si Lambda está deshabilitado, Firehose limita su registro de entrada a 10 MB. Si Amazon Data Firehose recibe un tamaño de registro de Amazon MSK superior a 10 MB, Amazon Data Firehose entrega este registro en el bucket de errores de S3 y emite métricas de CloudWatch a su cuenta. Para obtener más información sobre los límites de AWS Lambda, consulte Cuotas de Lambda.
- Cuando se habilita el particionamiento dinámico en un flujo de Firehose, hay una cuota predeterminada de 500 particiones activas que se pueden crear para ese flujo de Firehose. El recuento de particiones activas es el número total de particiones activas en el búfer de entrega. Por ejemplo, si la consulta de particionamiento dinámico crea 3 particiones por segundo y tiene una configuración de sugerencias de búfer que activa la entrega cada 60 segundos, tendrá un promedio de 180 particiones activas. Una vez que los datos se entregan en una partición, dicha partición deja de estar activa. Si necesita más particiones, puede crear más flujos de Firehose y distribuir las particiones activas entre ellos.
- Cuando se habilita el particionamiento dinámico en un flujo de Firehose, se admite un rendimiento máximo de 1 GB por segundo para cada partición activa.

- Cada cuenta tendrá la siguiente cuota para el número de flujos de Firehose por región:
  - Este de EE. UU. (Norte de Virginia), Este de EE. UU. (Ohio), Oeste de EE. UU. (Oregón), Europa (Irlanda), Asia-Pacífico (Tokio): 5000 flujos de Firehose
  - Europa (Fráncfort), Europa (Londres), Asia Pacífico (Singapur), Asia Pacífico (Sídney), Asia Pacífico (Seúl), Asia Pacífico (Bombay) AWS GovCloud , (EEUU-Oeste), Canadá (Oeste), Canadá (Centro): 2000 arroyos Firehose
  - Europa (París), Europa (Milán), Europa (Estocolmo), Asia Pacífico (Hong Kong), Asia Pacífico (Osaka), Sudamérica (São Paulo), China (Ningxia), China (Pekín), Oriente Medio (Baréin), (EEUU-Este), África AWS GovCloud (Ciudad del Cabo): 500 arroyos Firehose
  - Europa (Zúrich), Europa (España), Asia-Pacífico (Hyderabad), Asia-Pacífico (Yakarta), Asia-Pacífico (Melbourne), Medio Oriente (Emiratos Árabes Unidos), Israel (Tel Aviv), Oeste de Canadá (Calgary), Canadá (centro), Asia-Pacífico (Malasia), Asia-Pacífico (Tailandia), México (centro): 100 flujos de Firehose
  - Si supera este número, una llamada a CreateDeliveryStream genera una excepción `LimitExceededException`. Para aumentar esta cuota, puede utilizar Service Quotas si está disponible en su región. Para obtener más información acerca de cómo usar Service Quotas, consulte Requesting a Quota Increase.
- Cuando Direct PUT se configura como fuente de datos, cada transmisión de Firehose proporciona la siguiente cuota combinada de solicitudes PutRecordy PutRecordBatchcuotas:
  - Para EE. UU. Este (Norte de Virginia), EE. UU. Oeste (Oregón) y Europa (Irlanda): records/second, 2,000 requests/second, and 5 MiB/second 500 000.
  - Para otros Regiones de AWS: 100 000records/second, 1,000 requests/second, and 1 MiB/second.

  Si un flujo Direct PUT sufre una limitación debido a volúmenes más altos de ingesta de datos que superan la capacidad de rendimiento de un flujo de Firehose, Amazon Data Firehose aumentará automáticamente el límite de rendimiento de la transmisión hasta que se contenga la limitación. Según el aumento del rendimiento y la limitación, Firehose podrá tardar más en aumentar el rendimiento de un flujo hasta los niveles deseados. Por este motivo, continúe reintentando los registros de ingesta de datos fallidos. Si espera que el volumen de datos aumente en cantidades grandes y repentinas, o si su nuevo flujo necesita un rendimiento superior al límite del predeterminado, solicite aumentar el límite de rendimiento.

  Existen tres escalas de cuotas proporcionales para las cuotas. Por ejemplo, si aumentas la cuota de rendimiento en EE.UU. Este (Norte de Virginia), EE.UU. Oeste (Oregón) o Europa (Irlanda) a

10. MiB/second, the other two quota increase to 4,000 requests/second and 1,000,000 records/second

> **Note**
>
> - No utilice límites y cuotas a nivel de recursos como una forma de controlar el uso del servicio.
> - Cuando Kinesis Data Streams está configurado como origen de datos, esta cuota no se aplica y Amazon Data Firehose escala o reduce verticalmente sin límite.
> - Si la cuota incrementada es muy superior al tráfico en ejecución, la entrega a los destinos se produce en lotes pequeños. Esto resulta poco eficiente y puede provocar costos superiores en los servicios de destino. Asegúrese de incrementar la cuota actual al nivel necesario para satisfacer solo el tráfico en ejecución, e increméntela más si el tráfico aumenta.
> - Los registros de datos más pequeños pueden generar costos más altos. Los precios de ingesta de Firehose se basan en el número de registros de datos que envíe al servicio, multiplicado por el tamaño de cada registro redondeado al alza a los 5 KB (5120 bytes) más cercanos. Por lo tanto, para el mismo volumen de datos de entrada (bytes), si hay un número mayor de registros de entrada, el costo incurrido será mayor. Por ejemplo, si el volumen total de datos de entrada es de 5 MiB, enviar 5 MiB de datos de más de 5000 registros cuesta más que enviar la misma cantidad de datos con 1000 registros. Para obtener más información, consulte Amazon Data Firehose en la calculadora de AWS.

- Cada Firehose Stream almacena los registros de datos durante un máximo de 24 horas en caso de que el destino de entrega no esté disponible y la fuente sí lo esté. DirectPut Si el origen es Kinesis Data Streams (KDS) y el destino no está disponible, los datos se conservarán en función de la configuración de KDS.
- El tamaño máximo de un registro enviado a Amazon Data Firehose antes de codificarse en base64 es de 1000 KiB.
- La operación PutRecordBatch admite por cada llamada el valor más pequeño de entre 500 registros o 4 MiB. Esta cuota no se puede cambiar.
- Cada una de las siguientes operaciones puede proporcionar hasta cinco invocaciones por segundo, que es un límite máximo.
  - `CreateDeliveryStream`

  - DeleteDeliveryStream
  - DescribeDeliveryStream
  - ListDeliveryStreams
  - UpdateDestination
  - TagDeliveryStream
  - UntagDeliveryStream
  - ListTagsForDeliveryStream
  - StartDeliveryStreamEncryption
  - StopDeliveryStreamEncryption
- Las sugerencias de intervalo del búfer oscilan entre 60 y 900 segundos.
- Para la entrega desde Amazon Data Firehose en Amazon Redshift, solo se admiten clústeres de Amazon Redshift de acceso público.
- El intervalo de duración de los reintentos es de 0 a 7200 segundos para Amazon Redshift y Service Delivery. OpenSearch
- Cuando el destino es Amazon S3, Amazon Redshift o OpenSearch Service, Amazon Data Firehose permite hasta 5 invocaciones Lambda pendientes por fragmento. En Splunk, la cuota es de 10 invocaciones de Lambda pendientes por partición.
- Puede utilizar una CMK de tipo `CUSTOMER_MANAGED_CMK` para cifrar hasta 500 flujos de Firehose.

Historial de documentos

En la siguiente tabla, se describen los cambios importantes que se han hecho en la documentación de Amazon Data Firehose.

Cambio	Descripción	Fecha de modificación
Eliminación de la base de datos como origen (vista previa pública)	Se eliminó la base de datos como origen (vista previa pública).	24 de septiembre de 2025
Se agregó compatibilidad con la jerarquía de catálogos múltiples de Glue	Esto simplifica la integración de Firehose con las tablas de Amazon S3 sin necesidad de enlaces de recursos entre el catálogo de datos predeterminado y `S3TablesCatalog` . Consulte Configuración de un flujo de Firehose para tablas de Amazon S3.	14 de mayo de 2025
Se agregó la base de datos como origen (vista previa pública)	Ahora puede replicar los cambios de bases de datos en las tablas de Apache Iceberg en Amazon S3.	15 de noviembre de 2024
Versión de disponibilidad general (GA) para las tablas de Apache Iceberg agregadas como destino	Puede crear un flujo de Firehose con las tablas de Apache Iceberg como destino. Consulte Entrega de datos a tablas de Apache Iceberg.	30 de septiembre de 2024
Se añadieron ejemplos de tipos de datos	Se añadieron ejemplos de tipos de datos compatibl es con las tablas de Apache Iceberg. Consulte Comprensión de los tipos de datos compatibles.	22 de agosto de 2024

Cambio	Descripción	Fecha de modificación
Lanzamiento de nueva Región	Amazon Data Firehose ya está disponible en la región de Asia-Pacífico (Malasia). Consulte Cuota de Amazon Data Firehose.	22 de agosto de 2024
Se añadieron las tablas de Apache Iceberg como destino (versión preliminar pública)	Puede crear un flujo de Firehose con las tablas de Apache Iceberg como destino. Consulte Entrega de datos a tablas de Apache Iceberg.	25 de julio de 2024
Sugerencias de almacenamiento en búfer para Snowflake	Snowflake ahora admite sugerencias de almacenamiento en búfer. Consulte the section called “Configuración de los ajustes de destino de Snowflake”.	25 de julio de 2024
Snowflake como destino en nuevas regiones	Snowflake ahora está disponible como destino en Asia-Pacífico (Singapur), Asia Pacífico (Seúl), y Asia Pacífico (Sídney). Consulte the section called “Configuración de los ajustes de destino de Snowflake”.	25 de julio de 2024
Secciones reestructuradas de la guía del usuario	Navegación simplificada para las secciones de la guía del usuario. Consulte Enviar datos a un flujo de Firehose y Errores de solución de problemas.	5 de julio de 2024
Amazon Data Firehose se integra con AWS Secrets Manager	Ahora puede acceder a sus secretos y automatizar la rotación de credenciales de forma segura con Secrets Manager. Consulte the section called “Autenticación con AWS Secrets Manager”.	6 de junio de 2024
Se agregó compatibilidad para la ingesta de registros para Dynatrace	Ahora puede enviar registros y eventos a Dynatrace para su posterior análisis. Consulte the section called “Configuración de los ajustes de destino de Dynatrace”.	18 de abril de 2024

Cambio	Descripción	Fecha de modificación
Versión de disponibilidad general (GA) para Snowflake como destino	Snowflake ahora está disponible de forma general como destino. Consulte the section called “Configuración de los ajustes de destino de Snowflake”.	17 de abril de 2024
Amazon Kinesis Data Firehose ahora se conoce como Amazon Data Firehose	Amazon Kinesis Data Firehose ha cambiado su nombre a Amazon Data Firehose. Consulte ¿Qué es Amazon Data Firehose?	9 de febrero de 2024
Se agregó Snowflake como destino (versión preliminar pública)	Es posible crear un flujo de Firehose con Snowflake como destino. Consulte the section called “Configuración de los ajustes de destino de Snowflake”.	19 de enero de 2024
Se agregó la descompresión automática de Registros de CloudWatch	Puede activar la descompresión de flujos nuevos o existentes para enviar datos descomprimidos de Registros de CloudWatch a los destinos de Firehose. Consulte the section called “Enviar los registros de CloudWatch a Firehose”.	15 de diciembre de 2023
Se agregó Splunk Observability Cloud como destino	Es posible crear un flujo de Firehose con Splunk Observability Cloud como destino. Consulte the section called “Configuración de los ajustes de destino de Splunk Observability Cloud”.	3 de octubre de 2023
Se agregó Amazon Managed Streaming para Apache Kafka como origen de datos	Ahora puede configurar Amazon MSK para enviar información a un flujo de Firehose. Consulte the section called “Configuración de los ajustes de origen para Amazon MSK”.	26 de septiembre de 2023

Cambio	Descripción	Fecha de modificación
Se agregó compatibilidad con el tipo DocumentID para el destino OpenSearch Service	Si el destino de su flujo de Firehose es OpenSearch Service, el tipo DocumentID indica el método para configurar el ID de documento. Los métodos admitidos son el ID de documento generado por Firehose y el ID de documento generado por OpenSearch Service. Consulte the section called “Configuración de los ajustes de destino”.	10 de mayo de 2023
Se agregó compatibilidad con el particionamiento dinámico	Se agregó compatibilidad con el particionamiento dinámico continuo de los datos de streaming en Amazon Data Firehose. Consulte Partición de datos de streaming.	31 de agosto de 2021
Se ha añadido un tema sobre los prefijos personalizados	Se agregó un tema sobre las expresiones que puede utilizar para crear un prefijo personalizado para los datos que se entregan en Amazon S3. Consulte the section called “Comprensión de los prefijos personalizados para los objetos de Amazon S3”.	20 de diciembre de 2018
Se agregó un nuevo tutorial de Amazon Data Firehose	Se agregó un tutorial en el que se explica cómo enviar registros de flujos de Amazon VPC a Splunk a través de Amazon Data Firehose. Consulte Ingesta de registros de flujo de VPC en Splunk mediante Amazon Data Firehose.	30 de octubre de 2018
Se han añadido cuatro regiones nuevas de Amazon Data Firehose	Se han añadido París, Mumbai, São Paulo y Londres. Para obtener más información, consulte Cuota de Amazon Data Firehose.	27 de junio de 2018
Se han añadido dos regiones nuevas de Amazon Data Firehose	Se han añadido Seúl y Montreal. Para obtener más información, consulte Cuota de Amazon Data Firehose.	13 de junio de 2018

Cambio	Descripción	Fecha de modificación
Nueva característica de Kinesis Streams como origen	Se ha añadido Kinesis Streams como un posible origen para los registros de un flujo de Firehose. Para obtener más información, consulte Elija el origen y el destino del flujo de Firehose.	18 de agosto de 2017
Actualización de la documentación de la consola	El asistente de creación del flujo de Firehose se ha actualizado. Para obtener más información, consulte Tutorial: Crear un flujo de Firehose desde la consola.	19 de julio de 2017
Nueva transformación de datos	Puede configurar Amazon Data Firehose para transformar los datos antes de su entrega. Para obtener más información, consulte Transformación de los datos de origen en Amazon Data Firehose.	19 de diciembre de 2016
Nuevo reintento de COPY de Amazon Redshift	Es posible configurar Amazon Data Firehose para reintentar un comando COPY en su clúster de Amazon Redshift si se produce un error. Para obtener más información, consulte Tutorial: Crear un flujo de Firehose desde la consola, Comprensión de la entrega de datos en Amazon Data Firehose y Cuota de Amazon Data Firehose.	18 de mayo de 2016
Nuevo destino de Amazon Data Firehose: Amazon OpenSearch Service	Es posible crear un flujo de Firehose con Amazon OpenSearch Service como destino. Para obtener más información, consulte Tutorial: Crear un flujo de Firehose desde la consola, Comprensión de la entrega de datos en Amazon Data Firehose y Conceda a Firehose acceso a un destino de servicio público OpenSearch .	19 de abril de 2016

Cambio	Descripción	Fecha de modificación
Nuevas métricas de CloudWatch y características de solución de problemas mejoradas	Actualización de Supervisión de Amazon Data Firehose y Solución de problemas en Amazon Data Firehose.	19 de abril de 2016
Nuevo agente de Kinesis mejorado	Se ha actualizado Configurar el agente de Kinesis para enviar datos.	11 de abril de 2016
Nuevos agentes de Kinesis	Se ha agregado Configurar el agente de Kinesis para enviar datos.	2 de octubre de 2015
Versión inicial	Versión inicial de la Guía para desarrolladores de Amazon Data Firehose.	4 de octubre de 2015