aws

开发人员指南

Amazon Data Firehose

Amazon Data Firehose: 开发人员指南

Table of Contents

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。

什么是 Amazon Data Firehose？

Amazon Data Firehose 是一项完全托管的服务，用于向亚马逊简单存储服务 (Amazon S3)、亚马逊 Redshift、亚马逊服务、亚马逊无服务器、Splunk、Apache Iceberg Tables 以及支持的第三方服务提供商拥有的任何自定义 HTTP 终端节点或 HTTP 终端节点（包括 Datadog、Dyn LogicMonitor atrace、MongoDB）等目的地提供实时流式数据、New Relic、Coralogix 和 Elastic。 OpenSearch OpenSearch 在使用 Amazon Data Firehose 时，您无需编写应用程序或管理资源。您可以配置数据生产者，将数据发送到 Amazon Data Firehose，后者会自动将数据传输到您指定的目的地。您还可以配置 Amazon Data Firehose 在传输之前转换数据。

有关 AWS 大数据解决方案的更多信息，请参阅上的 Big Data AWS。有关 AWS 流数据解决方案的更多信息，请参阅什么是流数据？。

了解关键概念

开始使用 Amazon Data Firehose 时，您可以从理解以下概念中受益。

Firehose 流

Amazon Data Firehose 的基础实体。您可以通过创建 Firehose 流，然后向其发送数据来使用 Amazon Data Firehose。有关更多信息，请参阅教程：从控制台创建 Firehose 流和向 Firehose 流发送数据。

记录

数据创建器发送到 Firehose 流的相关数据。记录最大可达 1000 KB。

数据创建器

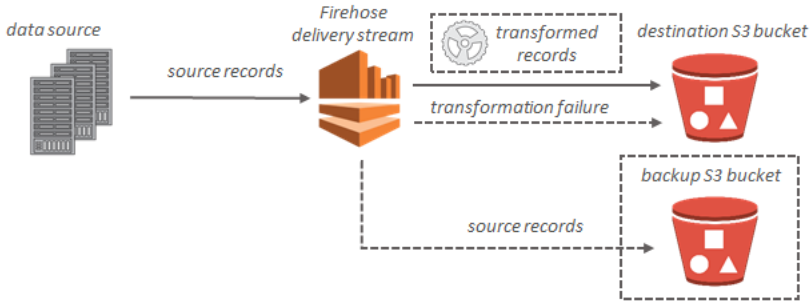
创建器将记录发送到 Firehose 流。例如，发送日志数据到 Firehose 流的 Web 服务器是数据创建器。您还可以配置 Firehose 流，以自动从现有 Kinesis 数据流读取数据，并将其加载到目的地。有关更多信息，请参阅 向 Firehose 流发送数据。

缓冲区大小和缓冲间隔

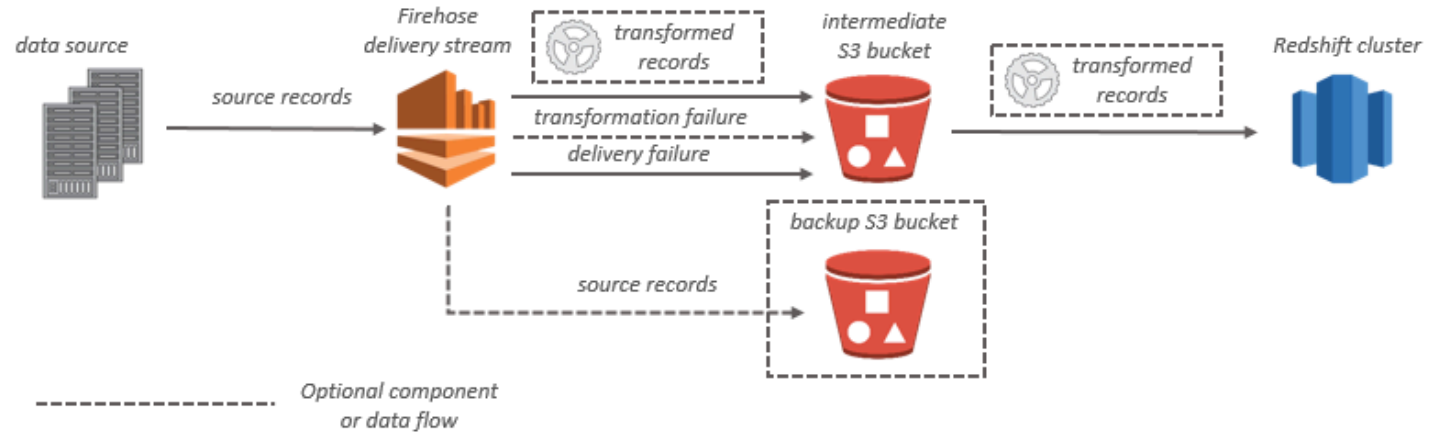
Amazon Data Firehose 会将传入的流数据缓冲到一定大小或一段时间，然后再将其传送到目的地。Buffer Size已进入 MBs 并以秒Buffer Interval为单位。

了解 Amazon Data Firehose 中的数据流

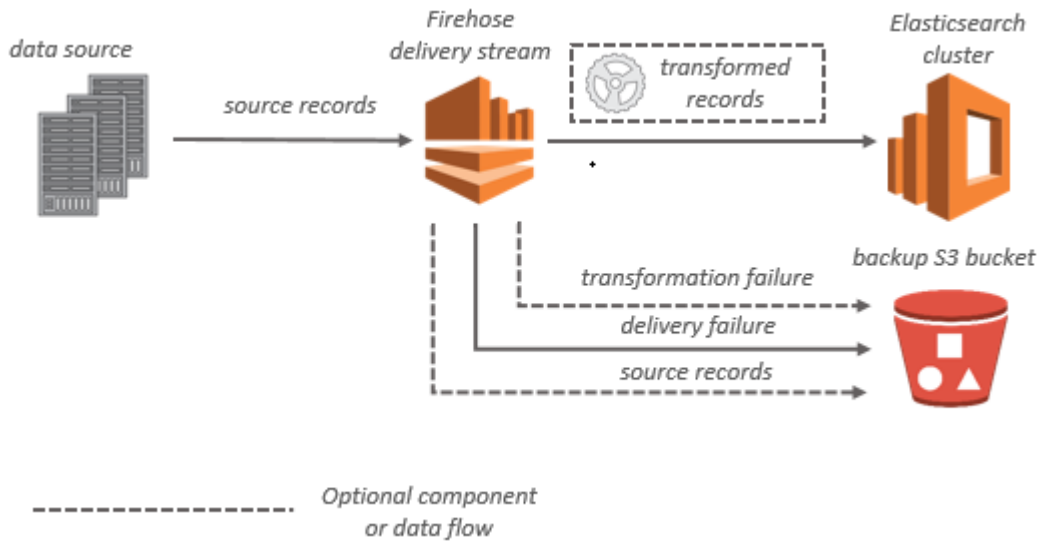
对于 Amazon S3 目标，流数据将传输到您的 S3 存储桶。如果启用了数据转换，您可以选择将源数据备份到另一个 Amazon S3 存储桶。



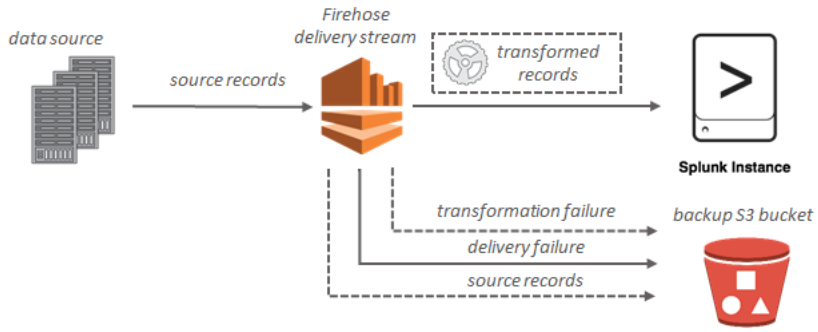
对于 Amazon Redshift 目标，流数据将传输到您的 S3 存储桶。然后，Amazon Data Firehose 会发出 Amazon Redshift COPY 命令，将数据从 S3 存储桶加载到 Amazon Redshift 集群。如果启用了数据转换，您可以选择将源数据备份到另一个 Amazon S3 存储桶。



对于 OpenSearch 服务目标，流数据将传输到您的 OpenSearch服务集群，并且可以选择将其同时备份到您的 S3 存储桶。



对于 Splunk 目标，流数据将传输到 Splunk，并且可以选择将流数据同时备份到 S3 存储桶中。



将 Firehose 与 SDK 配合使用 AWS

AWS 软件开发套件 (SDKs) 可用于许多流行的编程语言。每个软件开发工具包都提供 API、代码示例和文档，使开发人员能够更轻松地以其首选语言构建应用程序。

SDK 文档	代码示例
适用于 C++ 的 AWS SDK	适用于 C++ 的 AWS SDK 代码示例
AWS CLI	AWS CLI 代码示例
适用于 Go 的 AWS SDK	适用于 Go 的 AWS SDK 代码示例
适用于 Java 的 AWS SDK	适用于 Java 的 AWS SDK 代码示例

SDK 文档	代码示例
适用于 JavaScript 的 AWS SDK	适用于 JavaScript 的 AWS SDK 代码示例
适用于 Kotlin 的 AWS SDK	适用于 Kotlin 的 AWS SDK 代码示例
适用于 .NET 的 AWS SDK	适用于 .NET 的 AWS SDK 代码示例
适用于 PHP 的 AWS SDK	适用于 PHP 的 AWS SDK 代码示例
AWS Tools for PowerShell	AWS Tools for PowerShell 代码示例
适用于 Python (Boto3) 的 AWS SDK	适用于 Python (Boto3) 的 AWS SDK 代码示例
适用于 Ruby 的 AWS SDK	适用于 Ruby 的 AWS SDK 代码示例
适用于 Rust 的 AWS SDK	适用于 Rust 的 AWS SDK 代码示例
适用于 SAP ABAP 的 AWS SDK	适用于 SAP ABAP 的 AWS SDK 代码示例
适用于 Swift 的 AWS SDK	适用于 Swift 的 AWS SDK 代码示例

示例可用性

找不到所需的内容？ 通过使用此页面底部的提供反馈链接请求代码示例。

完成设置 Amazon Data Firehose 的先决条件

首次使用 Amazon Data Firehose 之前，请先完成以下任务。

任务

- 报名参加 AWS
- （可选）下载库和工具

报名参加 AWS

当您注册亚马逊网络服务 (AWS) 时，您的 AWS 账户将自动注册所有服务 AWS，包括亚马逊 Data Firehose。您只需为使用的服务付费。

如果您已经有一个 AWS 帐户，请跳到下一个任务。如果您还没有 AWS 账户，请按照以下步骤创建。

注册 AWS 账户

1. 打开https://portal.aws.amazon.com/billing/注册。
2. 按照屏幕上的说明操作。

   在注册时，将接到电话或收到短信，要求使用电话键盘输入一个验证码。

   当您注册时 AWS 账户，就会创建AWS 账户根用户一个。根用户有权访问该账户中的所有 AWS 服务 和资源。作为最佳安全实践，请为用户分配管理访问权限，并且只使用根用户来执行需要根用户访问权限的任务。

（可选）下载库和工具

以下库和工具可帮助您以编程方式以及从命令行使用 Amazon Data Firehose：

- Firehose API 操作 是 Amazon Data Firehose 支持的一组基本操作。
- f o AWS SDKs r Go、Java、.NET、Node.js、Python 和 Ruby 包括亚马逊 Data Firehose 支持和示例。

  如果您的版本 适用于 Java 的 AWS SDK 不包括 Amazon Data Firehose 的示例，您也可以从中下载最新的 AWS 软件开发工具包。GitHub

- AWS Command Line Interface 支持 Amazon Data Firehose。 AWS CLI 使您可以从命令行控制多项 AWS 服务，并通过脚本自动执行这些服务。

教程：从控制台创建 Firehose 流

您可以使用 AWS 管理控制台 或 AWS SDK 创建到您所选目的地的 Firehose 直播。

Firehose 直播创建后，您可以随时使用亚马逊数据 Firehose 控制台或更新其配置。UpdateDestination在更新配置时，Firehose 流将保持 `Active` 状态，您可以继续发送数据。更新后的配置通常在数分钟内生效。更新配置后，Firehose 流的版本号将增加一个值 1。该增加值反映在传输的 Amazon S3 对象名称中。有关更多信息，请参阅 配置 Amazon S3 对象名称格式。

执行以下主题中的步骤来创建 Firehose 流。

主题

为 Firehose 流选择来源和目的地

1. 在 https://console.aws.amazon.com/firehose/ 中打开 Firehose 控制台。
2. 选择创建 Firehose 流。
3. 在创建 Firehose 流页面上，从以下选项中选择一个 Firehose 流的来源。
    - Direct PUT：选择此选项可创建 Firehose 流，供生产者应用程序直接写入。以下是与 Amazon Data Firehose 中的 Direct PUT 集成的 AWS 服务、代理以及开源服务的列表。此列表并非详尽无遗，可能还有其他服务可用于将数据直接发送到 Firehose。
        - AWS SDK
        - AWS Lambda
        - AWS CloudWatch Logs
        - AWS CloudWatch Events
        - AWS Cloud Metric Streams

- AWS IoT
- AWS Eventbridge
- Amazon Simple Email Service
- Amazon SNS
- AWS WAF web ACL logs
- Amazon API Gateway - 访问日志
- Amazon Pinpoint
- Amazon MSK 代理日志
- Amazon Route 53 Resolver 查询日志
- AWS Network Firewall 警报日志
- AWS Network Firewall 流日志
- Amazon ElastiCache Redis SLOWLOG
- Kinesis Agent（linux）
- Kinesis Tap（windows）
- Flientbit
- Fluentd
- Apache Nifi
- Snowflake

- Amazon Kinesis Data Streams：选择此选项，以配置使用 Kinesis 数据流作为数据来源的 Firehose 流。然后，您可以使用 Firehose 从现有 Kinesis 数据流轻松读取数据，并将其加载到目的地。有关使用 Kinesis Data Streams 作为数据来源的更多信息，请参阅使用 Kinesis Data Streams 将数据发送到 Firehose。
- Amazon MSK：选择此选项，以配置使用 Amazon MSK 作为数据来源的 Firehose 流。然后，您可以使用 Firehose，从现有 Amazon MSK 集群轻松读取数据，并将其加载到指定的 S3 存储桶。有关更多信息，请参阅使用 Amazon MSK 将数据发送到 Firehose 流。

4. 从 Firehose 支持的以下目的地之一中为您的 Firehose 流选择一个目的地。

- Amazon OpenSearch Service
- Amazon OpenSearch 无服务器
- Amazon Redshift
- Amazon S3
- Apache Iceberg 表

- Coralogix
- Datadog
- Dynatrace
- 弹性
- HTTP 端点
- Honeycomb
- Logic Monitor
- Logz.io
- MongoDB Cloud
- New Relic
- Splunk
- Splunk Observability Cloud
- Sumo Logic
- Snowflake

5. 对于 Firehose 流名称，您可以使用控制台为您生成的名称，也可以添加您选择的 Firehose 流。

配置源设置

您可以根据您选择的来源配置源设置，以便从控制台向 Firehose 流发送信息。您可以将 Amazon MSK 和 Amazon Kinesis Data Streams 的源设置配置为来源。没有可用于 Direct PUT 的源设置作为源。

为 Amazon MSK 配置源设置

当您选择 Amazon MSK 将信息发送到 Firehose 流时，可以在 MSK 预置和 MSK 无服务器集群之间选择。然后，您可以使用 Firehose 从特定的 Amazon MSK 集群和主题轻松读取数据，并将其加载到指定的 S3 目的地。

在本页面的源设置部分，为以下字段提供值。

Amazon MSK 集群连接

根据您的集群配置，选择私有引导代理（推荐）或公有引导代理选项。引导代理是 Apache Kafka 客户端用来连接集群的起点。公有引导代理用于从 AWS 外部公开访问，而私有引导代理用于从 AWS 内部访问。有关 Amazon MSK 的更多信息，请参阅 Amazon Managed Streaming for Apache Kafka。

要通过私有引导代理连接到预置或无服务器 Amazon MSK 集群，该集群必须满足以下所有要求。

- 集群必须处于活动状态。
- 集群必须将 IAM 作为其访问控制方法之一。
- 必须为 IAM 访问控制方法启用多 VPC 私有连接。
- 您必须向此集群添加基于资源的策略，该策略为 Firehose 服务主体授予调用 Amazon MSK `CreateVpcConnection` API 操作的权限。

要通过公有引导代理连接到预置 Amazon MSK 集群，该集群必须满足以下所有要求。

- 集群必须处于活动状态。
- 集群必须将 IAM 作为其访问控制方法之一。
- 集群必须可公开访问。

MSK 集群账户

您可以选择 Amazon MSK 集群所在的账户。这可能是以下各项之一。

- 当前账户：使您能够从当前 AWS 账户的 MSK 集群中摄取数据。为此，您必须指定 Firehose 流将从中读取数据的 Amazon MSK 集群 ARN。
- 跨账户：使您能够从另一个 AWS 账户的 MSK 集群中摄取数据。有关更多信息，请参阅 Cross-account 从亚马逊 MSK 发货。

主题

指定您需要 Firehose 流从中摄取数据的 Apache Kafka 主题。Firehose 流创建完成后，您将无法更新此主题。

> **Note**
>
> Firehose 自动解压缩 Apache Kafka 消息。

配置 Amazon Kinesis Data Streams 的源设置

配置 Amazon Kinesis Data Streams 的源设置以将信息如下所示发送到 Firehose 流。

> **Important**
>
> 如果使用 Kinesis Producer Library（KPL）将数据写入 Kinesis 数据流，则可以使用聚合来合并写入该 Kinesis 数据流的记录。如果随后将该数据流用作 Firehose 流的来源，Amazon Data

Firehose 将在记录传送到目的地之前解聚。如果您配置 Firehose 流来转换数据，则 Amazon Data Firehose 在将记录传输到 AWS Lambda 之前，会先解聚记录。有关更多信息，请参阅使用 Kinesis 创建者库开发 Amazon Kinesis Data Streams 创建者和聚合。

在源设置下，在 Kinesis 数据流列表中选择现有流，或者输入格式为 `arn:aws:kinesis:[Region]:[AccountId]:stream/[StreamName]` 的数据流 ARN。

如果您没有现有的数据流，则请选择创建，以从 Amazon Kinesis 控制台创建一个新的数据流。您可能需要一个对 Kinesis 流具有必要权限的 IAM 角色。有关更多信息，请参阅 ???。创建新流后，选择“刷新”图标，以更新 Kinesis 流列表。如果您有大量的流，可使用 Filter by name 筛选列表。

> **Note**
>
> 将 Kinesis 数据流配置为 Firehose 流的源时，Amazon Data Firehose `PutRecord` 和 `PutRecordBatch` 操作将被禁用。在这种情况下，要将数据添加到 Firehose 流，请使用 Kinesis Data Streams `PutRecord` 和 `PutRecords` 操作。

Amazon Data Firehose 开始从 Kinesis 流的 `LATEST` 位置读取数据。有关 Kinesis Data Streams 位置的更多信息，请参阅 GetShardIterator。

Amazon Data Firehose 每秒为每个分片调用一次 Kinesis Data Streams GetRecords 操作。但是，启用完整备份后，Firehose 会每秒对每个分片调用 Kinesis Data Streams `GetRecords` 操作两次，一次用于主传输目的地，另一次用于完整备份。

可从同一个 Kinesis 流中读取多个 Firehose 流。其他 Kinesis 应用程序（使用者）也可以从同一个流中读取。来自任何 Firehose 流或其他消费端应用程序的所有调用都会计入该分区的总体限制。为了避免受限，请小心计划您的应用程序。有关 Kinesis Data Streams 限制的更多信息，请参阅 Amazon Kinesis Streams 限制。

继续执行下一步，以配置记录转换和格式转换。

（可选）配置记录转换和格式转换

配置 Amazon Data Firehose 以转换记录数据。

如果您选择 Amazon MSK 作为 Firehose 流的来源。

在 “使用 AWS Lambda 转换源记录” 部分中，为以下字段提供值。

1. 数据转换

   要创建不转换传入数据的 Firehose 流，请不要选中启用数据转换复选框。

   要为 Firehose 指定 Lambda 函数，以便在传输传入数据之前调用并使用该函数转换传入数据，请选中启用数据转换复选框。您可以使用其中一个 Lambda 蓝图配置新的 Lambda 函数，也可以选择现有的 Lambda 函数。Lambda 函数必须包含 Firehose 所需的状态模型。有关更多信息，请参阅 在 Amazon Data Firehose 中转换源数据。

2. 在 Convert record format (转换记录格式) 部分，为以下字段提供值：

   Record format conversion (记录格式转换)

   要创建不转换传入数据记录格式的 Firehose 流，请选择禁用。

   要转换传入记录的格式，选择 Enabled (启用)，然后指定所需的输出格式。您需要指定一个 AWS Glue 表，其中包含您希望 Firehose 用来转换记录格式的架构。有关更多信息，请参阅转换输入数据格式。

   有关如何使用设置记录格式转换的示例 CloudFormation，请参阅AWS::KinesisFirehose::DeliveryStream。

如果您选择 Amazon Kinesis Data Streams 或 Direct PUT 作为 Firehose 流的来源

在源设置部分中，提供以下字段。

1. 在转换记录下，选择以下选项之一：

   a. 如果您的目标是 Amazon S3 或 Splunk，请在 “解压缩源记录 Amazon CloudWatch 日志” 部分，选择 “开启解压缩”。

   b. 在 “使用 AWS Lambda 转换源记录” 部分中，为以下字段提供值：

      数据转换

      要创建不转换传入数据的 Firehose 流，请不要选中启用数据转换复选框。

要为 Amazon Data Firehose 指定 Lambda 函数，以便在传输传入数据之前调用并使用该函数转换传入数据，请选中启用数据转换复选框。您可以使用其中一个 Lambda 蓝图配置新的 Lambda 函数，也可以选择现有的 Lambda 函数。Lambda 函数必须包含 Amazon Data Firehose 所需的状态模型。有关更多信息，请参阅 在 Amazon Data Firehose 中转换源数据。

2. 在 Convert record format (转换记录格式) 部分，为以下字段提供值：

   Record format conversion (记录格式转换)

   要创建不转换传入数据记录格式的 Firehose 流，请选择禁用。

   要转换传入记录的格式，选择 Enabled (启用)，然后指定所需的输出格式。您需要指定一个 AWS Glue 表，其中包含您希望 Amazon Data Firehose 用来转换记录格式的架构。有关更多信息，请参阅 转换输入数据格式。

   有关如何使用设置记录格式转换的示例 CloudFormation，请参阅AWS::KinesisFirehose::DeliveryStream。

配置目的地设置

本节介绍了您必须根据所选目的地设置为 Firehose 流配置的设置。

主题

- 为 Logz.io 配置目的地设置
- 为 MongoDB Atlas 配置目的地设置
- 为 New Relic 配置目的地设置
- 为 Snowflake 配置目的地设置
- 为 Splunk 配置目的地设置
- 为 Splunk Observability Cloud 配置目的地设置
- 为 Sumo Logic 配置目的地设置
- 为 Elastic 配置目的地设置

为 Amazon S3 配置目的地设置

您必须指定以下设置，才能使用 Amazon S3 作为 Firehose 流的目的地。

- 输入以下字段的值。

  S3 存储桶

  请选择一个您拥有的用于接收流数据的 S3 存储桶。您可以创建一个新 S3 存储桶或选择现有的 S3 存储桶。

  新行分隔符

  您可以配置 Firehose 流，在传输到 Amazon S3 的对象的记录之间添加新行分隔符。为此，请选择启用。若不在传输到 Amazon S3 的对象中的记录之间添加新行分隔符，请选择禁用。如果您计划使用 Athena 来查询包含聚合记录的 S3 对象，则请启用此选项。

  动态分区

  选择启用以启用和配置动态分区。

  多记录解聚

  多记录解聚是解析 Firehose 流中的记录，并根据有效的 JSON 或指定的换行符分隔记录的过程。

  如果您将多个事件、日志或记录聚合到单个 PutRecord 和 PutRecordBatch API 调用中，则仍然可以启用和配置动态分区。对于聚合数据，启用动态分区时，Amazon Data Firehose 会解析记录，并在每次 API 调用中查找多个有效的 JSON 对象。将 Firehose 流配置为使用 Kinesis Data Stream 作为源时，您还可以使用 Kinesis Producer Library（KPL）中的内置聚合。数据

分区功能在数据解聚后执行。因此，每次 API 调用中的每条记录都可以传输到不同的 Amazon S3 前缀。您还可以利用 Lambda 函数集成，在数据分区功能之前执行任何其他解聚或转换。

> **Important**
>
> 如果数据是聚合的，则只有在执行数据解聚后才能应用动态分区。因此，如果您对聚合数据启用动态分区，则必须选择启用才能启用多记录解聚。

Firehose 流按以下顺序执行处理步骤：KPL（protobuf）解聚、JSON 或分隔符解聚、Lambda 处理、数据分区、数据格式转换和 Amazon S3 传输。

多记录解聚类型

如果您启用了多记录解聚，则必须指定 Firehose 解聚数据的方法。使用下拉菜单选择 JSON 或分隔。

内联解析

内联解析是一种受支持的机制，可以对发往 Amazon S3 的数据进行动态分区。要使用内联解析对数据进行动态分区，则必须指定要用作分区键的数据记录参数，并为每个指定的分区键提供一个值。选择启用以启用和配置内联解析。

> **Important**
>
> 如果您在上述步骤中指定了 AWS Lambda 函数来转换源记录，则可以使用此函数对绑定到 S3 的数据进行动态分区，并且仍然可以通过内联解析创建分区密钥。通过动态分区，您可以使用内联解析或 Lambd AWS a 函数来创建分区密钥。或者，您可以同时使用内联解析和 Lamb AWS da 函数来创建分区密钥。

动态分区键

您可以使用键和值字段指定用作动态分区键的数据记录参数，并使用 jq 查询生成动态分区键值。Firehose 仅支持 jq 1.6。您最多可以指定 50 个动态分区键。您必须为动态分区键值输入有效的 jq 表达式，才能成功为 Firehose 流配置动态分区。

S3 存储桶前缀

启用和配置动态分区后，必须指定 S3 存储桶前缀，Amazon Data Firehose 要向该存储桶传输分区数据。

为了正确配置动态分区，S3 存储桶前缀的数量必须与指定分区键的数量相同。

您可以使用内联解析或指定的 Lambd AWS a 函数对源数据进行分区。如果您指定了 AWS Lambda 函数来为源数据创建分区密钥，则必须使用以下格式手动键入 S3 存储桶前缀值：“Lambda: keyID” partitionKeyFrom。如果您使用内联解析为源数据指定分区密钥，则可以使用以下格式手动键入 S3 存储桶预览值：“partitionKeyFromquery: keyID”，也可以选择应用动态分区密钥按钮使用动态分区 key/value 对自动生成 S3 存储桶前缀。在使用内联解析或 Lambda AWS 对数据进行分区时，您还可以在您的 S3 存储桶前缀中使用以下表达式形式：! {namespace: value}，其中命名空间可以是 Query partitionKeyFrom 或 Lambda。partitionKeyFrom

S3 存储桶和 S3 错误输出前缀时区

在 Amazon S3 对象的自定义前缀中选择要用作日期和时间的时区。默认情况下，Firehose 会添加以 UTC 为单位的时间前缀。如果您想使用不同的时区，则可以更改 S3 前缀中使用的时区。

缓冲提示

Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目标位置的建议缓冲区大小因服务提供商而异。

S3 压缩

选择 GZIP、Snappy、Zip 或 Hadoop 兼容的 Snappy 数据压缩，或者不压缩数据。Snappy、Zip 和 Hadoop 兼容的 Snappy 压缩，不适用于以 Amazon Redshift 作为目的地的 Firehose 流。

S3 文件扩展名格式（可选）

为传输到 Amazon S3 目标存储桶的对象指定文件扩展名格式。如果启用此功能，则指定的文件扩展名将覆盖数据格式转换或 S3 压缩功能（例如 .parquet 或 .gz）附加的默认文件扩展名。在将此功能与数据格式转换或 S3 压缩配合使用时，请确保您配置了正确的文件扩展名。文件扩展名必须以句点（.）开头，并且可以包含允许的字符：0-9a-z!-_.*‘()。文件扩展名不能超过 128 个字符。

S3 加密

Firehose 支持使用 AWS Key Management Service (SSE-KMS) 的 Amazon S3 服务器端加密，用于加密亚马逊 S3 中交付的数据。您可以选择使用目标 S3 存储桶中指定的默认加密类型，也可以选择使用您拥有的 AWS KMS 密钥列表中的密钥进行加密。如果您使用密钥加密数

据，则可以使用默认 AWS 托管 AWS KMS 密钥 (aws/s3) 或客户托管密钥。有关更多信息，请参阅使用 AWS KMS 托管密钥的服务器端加密 (SSE-KMS) 保护数据。

配置 Apache Iceberg 表的目的地设置

除AWS 区域中国地区、亚太地区（台北）、亚太地区（马来西亚）、亚太地区（新西兰）和墨西哥（中部）外 AWS GovCloud (US) Regions，Firehose 支持将 Apache Iceberg Tables 作为目的地。

有关将 Apache Iceberg 表作为目的地的更多信息，请参阅使用 Amazon Data Firehose 将数据传输到 Apache Iceberg 表。

为 Amazon Redshift 配置目的地设置

本节介绍使用 Amazon Redshift 作为 Firehose 流目的地的设置。

根据您是否拥有 Amazon Redshift 预置集群或 Amazon Redshift Serverless 工作组，选择以下任一过程。

- Amazon Redshift 预置集群
- 为 Amazon Redshift Serverless 工作组配置目的地设置

> **Note**
>
> Firehose 无法写入使用增强型 VPC 路由的 Amazon Redshift 集群。

Amazon Redshift 预置集群

本节介绍使用 Amazon Redshift 预置集群作为 Firehose 流目的地的设置。

- 输入以下字段的值：

  集群

  Amazon Redshift 集群，S3 存储桶数据将复制到该集群。将 Amazon Redshift 集群配置为可公开访问，并取消阻止 Amazon Data Firehose IP 地址。有关更多信息，请参阅 授予 Firehose 对 Amazon Redshift 目的地的访问权限。

身份验证

您可以选择 username/password 直接输入，也可以从中检索密钥 AWS Secrets Manager 以访问 Amazon Redshift 集群。

- 用户名

  指定有权访问 Amazon Redshift 集群的 Amazon Redshift 用户。该用户必须具有 Amazon Redshift `INSERT` 权限才能将数据从 S3 存储桶复制到 Amazon Redshift 集群。

- 密码

  指定有权访问集群的用户的密码。

- 密钥

  从中 AWS Secrets Manager 选择一个包含 Amazon Redshift 集群凭证的密钥。如果在下拉列表中未看到您的密钥，则请在 AWS Secrets Manager 中为您的 Amazon Redshift 凭证创建一个。有关更多信息，请参阅 使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

数据库

Amazon Redshift 数据库，数据将复制到该数据库。

表

Amazon Redshift 表，数据将复制到该表。

列

（可选）表的特定列，数据将复制到该特定列。如果 Amazon S3 对象中定义的列数少于 Amazon Redshift 表中的列数，请使用此选项。

中间 S3 目标

Firehose 会先将数据传输到 S3 存储桶，然后发出 Amazon Redshift COPY 命令将数据加载到 Amazon Redshift 集群。请指定一个您拥有的用于接收流数据的 S3 存储桶。创建新的 S3 存储桶或选择您当前拥有的存储桶。

将数据加载到 Amazon Redshift 集群后，Firehose 不会将数据从 S3 存储桶中删除。您可以使用生命周期配置管理 S3 存储桶中的数据。有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的对象生命周期管理。

中间 S3 存储桶前缀

（可选）要对 Amazon S3 对象使用默认前缀，请将此选项留空。Firehose 会自动对传输的 Amazon S3 对象使用“YYYY/MM/dd/HH”UTC 时间格式的前缀。您可以将此前缀添加到开头。有关更多信息，请参阅 配置 Amazon S3 对象名称格式。

COPY options

您可以在 Amazon Redshift COPY 命令中指定的参数。您可以根据自己的配置情况酌情使用这些参数。例如，如果启用了 Amazon S3 数据压缩，则必须使用 GZIP “”。 如果您的 S3 存储桶与您的 Amazon Redshift 集群不在同一个 AWS 区域，则必须填写 “REGION”。有关更多信息，请参阅《Amazon Redshift 数据库开发人员指南》中的 COPY。

COPY command

Amazon Redshift COPY 命令。有关更多信息，请参阅《Amazon Redshift 数据库开发人员指南》中的 COPY。

Retry duration

如果将数据 COPY 到 Amazon Redshift 集群失败，Firehose 重试的持续时间（0-7200 秒）。Firehose 每 5 分钟重试一次，直到重试持续时间结束。如果将重试持续时间设置为 0（零）秒，Firehose 在 COPY 命令失败时不会重试。

缓冲提示

Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目标位置的建议缓冲区大小因服务提供商而异。

S3 压缩

选择 GZIP、Snappy、Zip 或 Hadoop 兼容的 Snappy 数据压缩，或者不压缩数据。Snappy、Zip 和 Hadoop 兼容的 Snappy 压缩，不适用于以 Amazon Redshift 作为目的地的 Firehose 流。

S3 文件扩展名格式（可选）

S3 文件扩展名格式（可选）：为传输到 Amazon S3 目标存储桶的对象指定文件扩展名格式。如果启用此功能，则指定的文件扩展名将覆盖数据格式转换或 S3 压缩功能（例如 .parquet 或 .gz）附加的默认文件扩展名。在将此功能与数据格式转换或 S3 压缩配合使用时，请确保您配置了正确的文件扩展名。文件扩展名必须以句点（.）开头，并且可以包含允许的字符：0-9a-z!-_.*‘()。文件扩展名不能超过 128 个字符。

S3 加密

Firehose 支持使用 AWS Key Management Service (SSE-KMS) 的 Amazon S3 服务器端加密，用于加密亚马逊 S3 中交付的数据。您可以选择使用目标 S3 存储桶中指定的默认加密类型，也可以选择使用您拥有的 AWS KMS 密钥列表中的密钥进行加密。如果您使用密钥加密数据，则可以使用默认 AWS 托管 AWS KMS 密钥 (aws/s3) 或客户托管密钥。有关更多信息，请参阅使用 AWS KMS 托管密钥的服务器端加密 (SSE-KMS) 保护数据。

为 Amazon Redshift Serverless 工作组配置目的地设置

本节介绍使用 Amazon Redshift Serverless 工作组作为 Firehose 流目的地的设置。

- 输入以下字段的值：

  Workgroup name (工作组名称)

  Amazon Redshift Serverless 工作组，S3 存储桶数据将复制到该工作组。将 Amazon Redshift Serverless 工作组配置为可公开访问，并取消阻止 Firehose IP 地址。有关更多信息，请参阅连接到 Amazon Redshift Serverless 中的“连接到可公开访问的 Amazon Redshift Serverless 实例”部分，以及“授予 Firehose 对 Amazon Redshift 目的地的访问权限”。

  身份验证

  您可以选择 username/password 直接输入，也可以从中检索密钥 AWS Secrets Manager 以访问 Amazon Redshift Serverless 工作组。

  - 用户名

    指定有权访问 Amazon Redshift Serverless 工作组的 Amazon Redshift 用户。该用户必须具有 Amazon Redshift `INSERT` 权限才能将数据从 S3 存储桶复制到 Amazon Redshift Serverless 工作组。

  - 密码

    指定有权访问 Amazon Redshift Serverless 工作组的用户的密码。

  - 密钥

    从中 AWS Secrets Manager 选择一个包含 Amazon Redshift 无服务器工作组凭证的密钥。如果在下拉列表中未看到您的密钥，则请在 AWS Secrets Manager 中为您的 Amazon

Redshift 凭证创建一个。有关更多信息，请参阅 使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

数据库

Amazon Redshift 数据库，数据将复制到该数据库。

表

Amazon Redshift 表，数据将复制到该表。

列

（可选）表的特定列，数据将复制到该特定列。如果 Amazon S3 对象中定义的列数少于 Amazon Redshift 表中的列数，请使用此选项。

中间 S3 目标

Amazon Data Firehose 会先将数据传输到 S3 存储桶，然后发出 Amazon Redshift COPY 命令，将数据加载到 Amazon Redshift Serverless 工作组。请指定一个您拥有的用于接收流数据的 S3 存储桶。创建新的 S3 存储桶或选择您当前拥有的存储桶。

将数据加载到 Amazon Redshift Serverless 工作组后，Firehose 不会从 S3 存储桶中删除数据。您可以使用生命周期配置管理 S3 存储桶中的数据。有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的对象生命周期管理。

中间 S3 存储桶前缀

（可选）要对 Amazon S3 对象使用默认前缀，请将此选项留空。Firehose 会自动对传输的 Amazon S3 对象使用“YYYY/MM/dd/HH”UTC 时间格式的前缀。您可以将此前缀添加到开头。有关更多信息，请参阅 配置 Amazon S3 对象名称格式。

COPY options

您可以在 Amazon Redshift COPY 命令中指定的参数。您可以根据自己的配置情况酌情使用这些参数。例如，如果启用了 Amazon S3 数据压缩，则必须使用 `GZIP` “”。 如果您的 S3 存储桶与您的 Amazon Redshift Serverless 工作组不在同一个 AWS 区域，则必须填写 “`REGION`”。有关更多信息，请参阅《Amazon Redshift 数据库开发人员指南》中的 COPY。

COPY command

Amazon Redshift COPY 命令。有关更多信息，请参阅《Amazon Redshift 数据库开发人员指南》中的 COPY。

Retry duration

如果将数据 COPY 到 Amazon Redshift Serverless 工作组失败，Firehose 重试的持续时间（0-7200 秒）。Firehose 每 5 分钟重试一次，直到重试持续时间结束。如果将重试持续时间设置为 0（零）秒，Firehose 在 COPY 命令失败时不会重试。

缓冲提示

Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目标位置的建议缓冲区大小因服务提供商而异。

S3 压缩

选择 GZIP、Snappy、Zip 或 Hadoop 兼容的 Snappy 数据压缩，或者不压缩数据。Snappy、Zip 和 Hadoop 兼容的 Snappy 压缩，不适用于以 Amazon Redshift 作为目的地的 Firehose 流。

S3 文件扩展名格式（可选）

S3 文件扩展名格式（可选）：为传输到 Amazon S3 目标存储桶的对象指定文件扩展名格式。如果启用此功能，则指定的文件扩展名将覆盖数据格式转换或 S3 压缩功能（例如 .parquet 或 .gz）附加的默认文件扩展名。在将此功能与数据格式转换或 S3 压缩配合使用时，请确保您配置了正确的文件扩展名。文件扩展名必须以句点（.）开头，并且可以包含允许的字符：0-9a-z!-_.*‘()。文件扩展名不能超过 128 个字符。

S3 加密

Firehose 支持使用 AWS Key Management Service (SSE-KMS) 的 Amazon S3 服务器端加密，用于加密亚马逊 S3 中交付的数据。您可以选择使用目标 S3 存储桶中指定的默认加密类型，也可以选择使用您拥有的 AWS KMS 密钥列表中的密钥进行加密。如果您使用密钥加密数据，则可以使用默认 AWS 托管 AWS KMS 密钥 (aws/s3) 或客户托管密钥。有关更多信息，请参阅使用 AWS KMS 托管密钥的服务器端加密 (SSE-KMS) 保护数据。

配置 OpenSearch 服务的目标设置

Firehose 支持 Elasticsearch 版本 1.5、2.3、5.1、5.3、5.5、5.6 以及所有的 6.*、7.* 和 8.* 版本。Firehose 支持亚马逊 OpenSearch 服务 2.x 和 3.x。

本节介绍在目的地使用 OpenSearch 服务的选项。

- 输入以下字段的值：

OpenSearch 服务域

您的数据传输到的 OpenSearch 服务域。

索引

将数据索引到 OpenSearch 服务集群时使用的 OpenSearch 服务索引名称。

Index rotation

选择是否轮换 OpenSearch 服务索引以及轮换频率。如果启用了索引轮换，Amazon Data Firehose 会将相应的时间戳附加到指定的索引名称，并进行轮换。有关更多信息，请参阅 为 OpenSearch 服务配置索引轮换。

Type

将数据索引到 OpenSearch 服务集群时使用的 OpenSearch 服务类型名称。对于 Elasticsearch 7. OpenSearch x 和 1.x，每个索引只能有一个类型。如果您尝试为已具有其他类型的现有索引指定新类型，Firehose 会在运行时返回错误。

对于 Elasticsearch 7.x，请将此字段留空。

Retry duration

Firehose 在索引请求失败时重试的持续时间。 OpenSearch 对于重试持续时间，您可以设置介于 0-7200 秒之间的任何值。默认重试持续时间是 300 秒。Firehose 将采用指数回退重试多次，直到重试持续时间到期为止。

重试持续时间到期后，Firehose 会将数据传输到死信队列（DLQ），这是配置的 S3 错误存储桶。对于传送到 DLQ 的数据，您必须将数据从配置的 S3 错误存储桶重新传回目的地。 OpenSearch

如果您想因 OpenSearch 集群停机或维护而阻止 Firehose stream 向 DLQ 传输数据，则可以将重试持续时间配置为更高的值（以秒为单位）。您可以联系 AWS 支持人员，将上述重试持续时间值增加到 7200 秒。

DocumentID 类型

指示设置文档 ID 的方法。支持的方法有 Firehose 生成的文档 ID 和 OpenSearch 服务生成的文档 ID。未设置文档 ID 值时，Firehose 生成的文档 ID 是默认选项。 OpenSearch 推荐使用服务生成的文档 ID，因为它支持写入密集型操作，包括日志分析和可观察性，在 OpenSearch 服务域中消耗更少的 CPU 资源，从而提高性能。

目标 VPC 连接

如果您的 OpenSearch 服务域位于私有 VPC 中，请使用此部分指定该 VPC。还要指定您希望 Amazon Data Firehose 在向您的服务域发送数据时使用的子网和子组。 OpenSearch 您可以使用与 OpenSearch 服务域相同的安全组。如果您指定不同的安全组，请确保它们允许 OpenSearch 服务域安全组的出站 HTTPS 流量。此外，请确保 OpenSearch 服务域的安全组允许来自您在配置 Firehose 直播时指定的安全组的 HTTPS 流量。如果您对 Firehose 直播和 OpenSearch 服务域使用同一个安全组，请确保安全组的入站规则允许 HTTPS 流量。有关安全组规则的更多信息，请参阅 Amazon VPC 文档中的安全组规则。

> **Important**
>
> 在私有 VPC 中指定将数据传输到目的地的子网时，请确保所选子网中有足够数量的免费 IP 地址。如果指定子网中没有可用的免费 IP 地址，Firehose 将无法在私有 VPC 中创建或添加 ENIs 数据传输，并且传输将降级或失败。

缓冲区提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目标位置的建议缓冲区大小因服务提供商而异。

为 OpenSearch 无服务器配置目标设置

本节介绍在目的地使用 OpenSearch 无服务器的选项。

- 输入以下字段的值：

  OpenSearch 无服务器集合

  数据传输到的一组 OpenSearch 无服务器索引的终端节点。

  索引

  将数据索引到 OpenSearch 无服务器集合时使用的 OpenSearch 无服务器索引名称。

目标 VPC 连接

如果您的 OpenSearch 无服务器集合位于私有 VPC 中，请使用此部分指定该 VPC。还要指定您希望 Amazon Data Firehose 在向您的无服务器集合发送数据时使用的子网和子组。OpenSearch

> **Important**
>
> 在私有 VPC 中指定将数据传输到目的地的子网时，请确保所选子网中有足够数量的免费 IP 地址。如果指定子网中没有可用的免费 IP 地址，Firehose 将无法在私有 VPC 中创建或添加 ENIs 数据传输，并且传输将降级或失败。

Retry duration

如果向 Serverless 发出的索引请求失败，Firehose 重试的持续时间。 OpenSearch 对于重试持续时间，您可以设置介于 0-7200 秒之间的任何值。默认重试持续时间是 300 秒。Firehose 将采用指数回退重试多次，直到重试持续时间到期为止。

重试持续时间到期后，Firehose 会将数据传输到死信队列（DLQ），这是配置的 S3 错误存储桶。对于传输到 DLQ 的数据，您必须将数据从配置的 S3 错误存储桶重新驱动回 OpenSearch 无服务器目标。

如果由于 OpenSearch 无服务器集群的停机或维护而想要阻止 Firehose stream 向 DLQ 传输数据，则可以将重试持续时间配置为更高的值（以秒为单位）。您可以联系 AWS 支持人员，将上述重试持续时间值增加到 7200 秒。

缓冲区提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目标位置的建议缓冲区大小因服务提供商而异。

配置 HTTP 端点的目的地设置

本节介绍了使用 HTTP 端点作为目标的选项。

> **Important**
>
> 如果您选择 HTTP 端点作为目标，请查看并按照 了解 HTTP 端点传输请求和响应规范 中的说明进行操作。

- 提供下列字段的值：

  HTTP 端点名称（可选）

  为 HTTP 端点指定一个用户友好的名称。例如 `My HTTP Endpoint Destination`。

  HTTP 端点 URL

  按以下格式指定 HTTP 端点的 URL：`https://xyz.httpendpoint.com`。URL 必须是 HTTPS URL。

  身份验证

  您可以选择直接输入访问密钥，也可以从中检索密钥 AWS Secrets Manager 以访问 HTTP 端点。

  - （可选）访问密钥

    如果您需要获取访问密钥，以便从 Firehose 向其端点传输数据，请联系端点所有者。

  - 密钥

    从中选择一个包 AWS Secrets Manager 含 HTTP 端点访问密钥的密钥。如果您在下拉列表中看不到您的密钥，请在中 AWS Secrets Manager 为访问密钥创建一个密钥。有关更多信息，请参阅 使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

  内容编码

  Amazon Data Firehose 使用内容编码来压缩请求的正文，然后再将请求发送到目的地。对请求的 enable/disable 内容编码选择 GZIP 或“禁用”。

  Retry duration

  指定 Amazon Data Firehose 重试将数据发送到所选 HTTP 端点的时长。

  发送数据后，Amazon Data Firehose 先等待 HTTP 端点的确认。如果出现错误或未在确认超时期限内收到确认，Amazon Data Firehose 将启动重试持续时间计数器。它将不断重试，直到

重试持续时间到期为止。重试持续时间到期后，Amazon Data Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

每次 Amazon Data Firehose 向 HTTP 端点发送数据（初始尝试或重试）时，都会重新启动确认超时计数器并等待 HTTP 端点的确认。

即使重试持续时间已过，Amazon Data Firehose 仍会等待确认，直到收到确认或达到确认超时期限。如果确认超时，Amazon Data Firehose 会确定重试计数器是否还有剩余时间。如果有剩余时间，它将再次重试并重复该逻辑，直到收到确认或确定重试时间已到期。

如果不希望 Amazon Data Firehose 重试发送数据，请将此值设置为 0。

参数（可选）

Amazon Data Firehose 在每次 HTTP 调用中都包含这些键值对。这些参数可以帮助您识别和组织目的地。

缓冲提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目标位置的建议缓冲区大小因服务提供商而异。

> **Important**
>
> 对于 HTTP 终端节点目标，如果您在 “ CloudWatch 日志” 中看到来自目标端点的 413 个响应代码，请降低 Firehose 流上的缓冲提示大小，然后重试。

为 Datadog 配置目的地设置

本节介绍了使用 Datadog 作为目标的选项。有关 Datadog 的更多信息，请参阅 https://docs.datadoghq.com/integrations/ amazon_web_services/。

- 提供下列字段的值。

  HTTP 端点 URL

  从下拉菜单中的以下选项之一选择要从中发送数据的位置。

  - Datadog 日志- US1
  - Datadog 日志- US3

- Datadog 日志- US5
- Datadog 日志- AP1
- Datadog 日志 - EU
- Datadog 日志 - GOV
- Datadog 指标 - US
- Datadog 指标- US5
- Datadog 指标- AP1
- Datadog 指标 - EU
- Datadog 配置- US1
- Datadog 配置- US3
- Datadog 配置- US5
- Datadog 配置- AP1
- Datadog 配置 - EU
- Datadog 配置 - US GOV

身份验证

你可以选择直接输入 API 密钥，也可以从中检索密钥 AWS Secrets Manager 来访问 Datadog。

- API 密钥

  联系 Datadog 以获取您从 Firehose 向此端点传输数据所需的 API 密钥。

- 密钥

  从中选择一个包 AWS Secrets Manager 含 Datadog API 密钥的密钥。如果在下拉列表中未看到您的密钥，则请在 AWS Secrets Manager中创建一个。有关更多信息，请参阅 使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

内容编码

Amazon Data Firehose 使用内容编码来压缩请求的正文，然后再将请求发送到目的地。对请求的 enable/disable 内容编码选择 GZIP 或 “禁用”。

Retry duration

指定 Amazon Data Firehose 重试将数据发送到所选 HTTP 端点的时长。

发送数据后，Amazon Data Firehose 先等待 HTTP 端点的确认。如果出现错误或未在确认超时期限内收到确认，Amazon Data Firehose 将启动重试持续时间计数器。它将不断重试，直到重试持续时间到期为止。重试持续时间到期后，Amazon Data Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

每次 Amazon Data Firehose 向 HTTP 端点发送数据（初始尝试或重试）时，都会重新启动确认超时计数器并等待 HTTP 端点的确认。

即使重试持续时间已过，Amazon Data Firehose 仍会等待确认，直到收到确认或达到确认超时期限。如果确认超时，Amazon Data Firehose 会确定重试计数器是否还有剩余时间。如果有剩余时间，它将再次重试并重复该逻辑，直到收到确认或确定重试时间已到期。

如果不希望 Amazon Data Firehose 重试发送数据，请将此值设置为 0。

参数（可选）

Amazon Data Firehose 在每次 HTTP 调用中都包含这些键值对。这些参数可以帮助您识别和组织目的地。

缓冲提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目标位置的建议缓冲区大小因服务提供商而异。

为 Honeycomb 配置目的地设置

本节介绍了使用 Honeycomb 作为目标的选项。有关 Honeycomb 的更多信息，请参阅 https://docs.honeycomb. io/getting-data-in/metrics/aws-cloudwatch-metrics/。

- 提供下列字段的值：

  Honeycomb Kinesis 端点

  使用以下格式指定 HTTP 终端节点的网址： https://api.honeycomb.io/1/kinesis_events/{{dataset}}

  身份验证

  您可以选择直接输入 API 密钥，也可以从中检索密钥 AWS Secrets Manager 以访问 Honeycomb。

- API 密钥

  联系 Honeycomb 以获取您从 Firehose 向此端点传输数据所需的 API 密钥。

- 密钥

  从中选择一个包含 Hon AWS Secrets Manager eycomb API 密钥的密钥。如果在下拉列表中未看到您的密钥，则请在 AWS Secrets Manager中创建一个。有关更多信息，请参阅 使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

内容编码

Amazon Data Firehose 使用内容编码来压缩请求的正文，然后再将请求发送到目的地。选择 GZIP 以启用请求的内容编码。这是 Honeycomb 目标的推荐选项。

Retry duration

指定 Amazon Data Firehose 重试将数据发送到所选 HTTP 端点的时长。

发送数据后，Amazon Data Firehose 先等待 HTTP 端点的确认。如果出现错误或未在确认超时期限内收到确认，Amazon Data Firehose 将启动重试持续时间计数器。它将不断重试，直到重试持续时间到期为止。重试持续时间到期后，Amazon Data Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

每次 Amazon Data Firehose 向 HTTP 端点发送数据（初始尝试或重试）时，都会重新启动确认超时计数器并等待 HTTP 端点的确认。

即使重试持续时间已过，Amazon Data Firehose 仍会等待确认，直到收到确认或达到确认超时期限。如果确认超时，Amazon Data Firehose 会确定重试计数器是否还有剩余时间。如果有剩余时间，它将再次重试并重复该逻辑，直到收到确认或确定重试时间已到期。

如果不希望 Amazon Data Firehose 重试发送数据，请将此值设置为 0。

参数（可选）

Amazon Data Firehose 在每次 HTTP 调用中都包含这些键值对。这些参数可以帮助您识别和组织目的地。

缓冲提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目标位置的建议缓冲区大小因服务提供商而异。

为 Coralogix 配置目的地设置

本节介绍了使用 Coralogix 作为目标的选项。有关 Coralogix 的更多信息，请参阅 Get Started with Coralogix。

- 提供下列字段的值：

  HTTP 端点 URL

  从下拉菜单的以下选项中选择 HTTP 端点 URL：

  - Coralogix - US
  - Coralogix - SINGAPORE
  - Coralogix - IRELAND
  - Coralogix - INDIA
  - Coralogix - STOCKHOLM

  身份验证

  您可以选择直接输入私钥，也可以从中检索密钥 AWS Secrets Manager 以访问 Coralogix。

  - 私有密钥

    联系 Coralogix 以获取您从 Firehose 向此端点传输数据所需的私有密钥。

  - 密钥

    从中选择一个包 AWS Secrets Manager 含 Coralogix 私钥的密钥。如果在下拉列表中未看到您的密钥，则请在 AWS Secrets Manager中创建一个。有关更多信息，请参阅 使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

  内容编码

  Amazon Data Firehose 使用内容编码来压缩请求的正文，然后再将请求发送到目的地。选择 GZIP 以启用请求的内容编码。这是 Coralogix 目标的推荐选项。

  Retry duration

  指定 Amazon Data Firehose 重试将数据发送到所选 HTTP 端点的时长。

  发送数据后，Amazon Data Firehose 先等待 HTTP 端点的确认。如果出现错误或未在确认超时期限内收到确认，Amazon Data Firehose 将启动重试持续时间计数器。它将不断重试，直到

重试持续时间到期为止。重试持续时间到期后，Amazon Data Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

每次 Amazon Data Firehose 向 HTTP 端点发送数据（初始尝试或重试）时，都会重新启动确认超时计数器并等待 HTTP 端点的确认。

即使重试持续时间已过，Amazon Data Firehose 仍会等待确认，直到收到确认或达到确认超时期限。如果确认超时，Amazon Data Firehose 会确定重试计数器是否还有剩余时间。如果有剩余时间，它将再次重试并重复该逻辑，直到收到确认或确定重试时间已到期。

如果不希望 Amazon Data Firehose 重试发送数据，请将此值设置为 0。

参数（可选）

Amazon Data Firehose 在每次 HTTP 调用中都包含这些键值对。这些参数可以帮助您识别和组织目的地。

- applicationName：运行 Data Firehose 的环境
- subsystemName：Data Firehose 集成的名称
- computerName：正在使用的 Firehose 流的名称

缓冲提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目的地的建议缓冲区大小因服务提供商而异。

为 Dynatrace 配置目的地设置

本节介绍了使用 Dynatrace 作为目标的选项。有关更多信息，请参阅 https://www.dynatrace.com/support/help/technology-support/cloud-platforms/amazon-web-services/integrations/cloudwatch-metric-streams/。

- 选择选项，以将 Dynatrace 作为 Firehose 流的目的地。

  摄取类型

  选择您要在 Dynatrace 中提供指标还是日志（默认），以供进一步分析和处理。

  HTTP 端点 URL

  从下拉菜单中选择 HTTP 端点 URL（Dynatrace US、Dynatrace EU 或 Dynatrace Global）。

身份验证

您可以选择直接输入 API 令牌，也可以从中检索密钥 AWS Secrets Manager 以访问 Dynatrace。

- API 令牌

  生成您从 Firehose 向此端点传输数据所需的 Dynatrace API 令牌。有关更多信息，请参阅 Dynatrace API - Tokens and authentication。

- 密钥

  从中 AWS Secrets Manager 选择一个包含 Dynatrace API 令牌的密钥。如果在下拉列表中未看到您的密钥，则请在 AWS Secrets Manager中创建一个。有关更多信息，请参阅 使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

API URL

提供 Dynatrace 环境的 API URL。

内容编码

选择是否要启用内容编码来压缩请求的正文。Amazon Data Firehose 使用内容编码来压缩请求的正文，然后再将请求发送到目的地。启用后，以 GZIP 格式压缩内容。

Retry duration

指定 Firehose 重试将数据发送到所选 HTTP 端点的时长。

发送数据后，Firehose 先等待 HTTP 端点的确认。如果出现错误或在确认超时期限内没有收到确认，Firehose 将启动重试持续时间计数器。它将不断重试，直到重试持续时间到期为止。重试持续时间到期后，Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

每次 Firehose 向 HTTP 端点发送数据（初始尝试期间或重试后）时，都会重新启动确认超时计数器并等待 HTTP 端点的确认。

即使重试持续时间到期，Firehose 仍会等待确认，直到它收到确认或到达确认超时期限。如果确认超时，Firehose 会确定重试计数器中是否还有剩余时间。如果有剩余时间，它将再次重试并重复该逻辑，直到收到确认或确定重试时间已到期。

如果不希望 Firehose 重试发送数据，请将此值设置为 0。

参数（可选）

Amazon Data Firehose 在每次 HTTP 调用中都包含这些键值对。这些参数可以帮助您识别和组织目的地。

缓冲提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。缓冲区提示包括流的缓冲区大小和时间间隔。目的地的建议缓冲区大小因服务提供商而异。

配置目标设置 LogicMonitor

本部分介绍将 LogicMonitor 用于您的目标的选项。有关更多信息，请参阅 https://www.logicmonitor.com。

- 提供下列字段的值：

  HTTP 端点 URL

  按以下格式指定 HTTP 端点的 URL。

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

  身份验证

  您可以选择直接输入 API 密钥，也可以从中检索密钥 AWS Secrets Manager 进行访问 LogicMonitor。

  - API 密钥

    请联系 LogicMonitor 以获取允许从 Firehose 向该端点传输数据所需的 API 密钥。

  - 密钥

    从中选择一个包 AWS Secrets Manager 含 API 密钥的密钥 LogicMonitor。如果在下拉列表中未看到您的密钥，则请在 AWS Secrets Manager中创建一个。有关更多信息，请参阅 使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

  内容编码

  Amazon Data Firehose 使用内容编码来压缩请求的正文，然后再将请求发送到目的地。对请求的 enable/disable 内容编码选择 GZIP 或“禁用”。

Retry duration

指定 Amazon Data Firehose 重试将数据发送到所选 HTTP 端点的时长。

发送数据后，Amazon Data Firehose 先等待 HTTP 端点的确认。如果出现错误或未在确认超时期限内收到确认，Amazon Data Firehose 将启动重试持续时间计数器。它将不断重试，直到重试持续时间到期为止。重试持续时间到期后，Amazon Data Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

每次 Amazon Data Firehose 向 HTTP 端点发送数据（初始尝试或重试）时，都会重新启动确认超时计数器并等待 HTTP 端点的确认。

即使重试持续时间已过，Amazon Data Firehose 仍会等待确认，直到收到确认或达到确认超时期限。如果确认超时，Amazon Data Firehose 会确定重试计数器是否还有剩余时间。如果有剩余时间，它将再次重试并重复该逻辑，直到收到确认或确定重试时间已到期。

如果不希望 Amazon Data Firehose 重试发送数据，请将此值设置为 0。

参数（可选）

Amazon Data Firehose 在每次 HTTP 调用中都包含这些键值对。这些参数可以帮助您识别和组织目的地。

缓冲提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目标位置的建议缓冲区大小因服务提供商而异。

为 Logz.io 配置目的地设置

本节介绍了使用 Logz.io 作为目标的选项。欲了解更多信息，请参阅 https://logz.io/。

> **Note**
>
> 在欧洲地区（米兰）区域，不支持将 Logz.io 作为 Amazon Data Firehose 目的地。

- 提供下列字段的值：

HTTP 端点 URL

按以下格式指定 HTTP 端点的 URL。URL 必须是 HTTPS URL。

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

例如

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

身份验证

您可以选择直接输入发货令牌，也可以从中检索密钥 AWS Secrets Manager 以访问 Logz.io。

- 送达令牌

  联系 Logz.io 以获取您从 Firehose 向此端点传输数据所需的送达令牌。

- 密钥

  从中 AWS Secrets Manager 选择一个包含 Logz.io 发货令牌的密钥。如果在下拉列表中未看到您的密钥，则请在 AWS Secrets Manager中创建一个。有关更多信息，请参阅 使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

Retry duration

指定 Amazon Data Firehose 重试向 Logz.io 发送数据的时长。

发送数据后，Amazon Data Firehose 先等待 HTTP 端点的确认。如果出现错误或未在确认超时期限内收到确认，Amazon Data Firehose 将启动重试持续时间计数器。它将不断重试，直到重试持续时间到期为止。重试持续时间到期后，Amazon Data Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

每次 Amazon Data Firehose 向 HTTP 端点发送数据（初始尝试或重试）时，都会重新启动确认超时计数器并等待 HTTP 端点的确认。

即使重试持续时间已过，Amazon Data Firehose 仍会等待确认，直到收到确认或达到确认超时期限。如果确认超时，Amazon Data Firehose 会确定重试计数器是否还有剩余时间。如果有剩余时间，它将再次重试并重复该逻辑，直到收到确认或确定重试时间已到期。

如果不希望 Amazon Data Firehose 重试发送数据，请将此值设置为 0。

参数（可选）

Amazon Data Firehose 在每次 HTTP 调用中都包含这些键值对。这些参数可以帮助您识别和组织目的地。

缓冲提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目标位置的建议缓冲区大小因服务提供商而异。

为 MongoDB Atlas 配置目的地设置

本节介绍了使用 MongoDB Atlas 作为目的地的选项。有关更多信息，请参阅 Amazon Web Services 上的 MongoDB Atlas。

- 提供下列字段的值：

  API Gateway URL

  按以下格式指定 HTTP 端点的 URL。

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

  URL 必须是 HTTPS URL。

  身份验证

  你可以选择直接输入 API 密钥，也可以从中检索密钥 AWS Secrets Manager 来访问 MongoDB Atlas。

  - API 密钥

    按照 Amazon Web Services 上的 MongoDB Atlas 中的说明获取您从 Firehose 向此端点传输数据所需的 `APIKeyValue`。

  - 密钥

    从中 AWS Secrets Manager 选择一个包含 API Gateway 的 API 密钥的密钥，该密钥由 Lambda 与 MongoDB Atlas 交互提供支持。如果在下拉列表中未看到您的密钥，则请在 AWS Secrets Manager中创建一个。有关更多信息，请参阅 使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

内容编码

Amazon Data Firehose 使用内容编码来压缩请求的正文，然后再将请求发送到目的地。对请求的 enable/disable 内容编码选择 GZIP 或 “禁用”。

Retry duration

指定 Amazon Data Firehose 重试将数据发送到所选第三方提供商的时长。

发送数据后，Amazon Data Firehose 先等待 HTTP 端点的确认。如果出现错误或未在确认超时期限内收到确认，Amazon Data Firehose 将启动重试持续时间计数器。它将不断重试，直到重试持续时间到期为止。重试持续时间到期后，Amazon Data Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

每次 Amazon Data Firehose 向 HTTP 端点发送数据（初始尝试或重试）时，都会重新启动确认超时计数器并等待 HTTP 端点的确认。

即使重试持续时间已过，Amazon Data Firehose 仍会等待确认，直到收到确认或达到确认超时期限。如果确认超时，Amazon Data Firehose 会确定重试计数器是否还有剩余时间。如果有剩余时间，它将再次重试并重复该逻辑，直到收到确认或确定重试时间已到期。

如果不希望 Amazon Data Firehose 重试发送数据，请将此值设置为 0。

缓冲提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目标位置的建议缓冲区大小因服务提供商而异。

参数（可选）

Amazon Data Firehose 在每次 HTTP 调用中都包含这些键值对。这些参数可以帮助您识别和组织目的地。

为 New Relic 配置目的地设置

本节介绍了使用 New Relic 作为目标的选项。有关更多信息，请参阅 https://newrelic.com。

- 提供下列字段的值：

HTTP 端点 URL

从下拉列表的以下选项中选择 HTTP 端点 URL。

- New Relic 日志 - US
- New Relic 指标 - US
- New Relic 指标 - EU

身份验证

你可以选择直接输入 API 密钥，也可以从中检索密钥 AWS Secrets Manager 来访问 New Relic。

- API 密钥

  在 New Relic One 账户设置中输入许可证密钥（40 个字符的十六进制字符串）。您需要此 API 密钥从 Firehose 向此端点传输数据。

- 密钥

  从中选择一个包含 N AWS Secrets Manager ew Relic 的 API 密钥的密钥。如果在下拉列表中未看到您的密钥，则请在 AWS Secrets Manager中创建一个。有关更多信息，请参阅 使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

内容编码

Amazon Data Firehose 使用内容编码来压缩请求的正文，然后再将请求发送到目的地。对请求的 enable/disable 内容编码选择 GZIP 或 “禁用”。

Retry duration

指定 Amazon Data Firehose 重试将数据发送到 New Relic HTTP 端点的时长。

发送数据后，Amazon Data Firehose 先等待 HTTP 端点的确认。如果出现错误或未在确认超时期限内收到确认，Amazon Data Firehose 将启动重试持续时间计数器。它将不断重试，直到重试持续时间到期为止。重试持续时间到期后，Amazon Data Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

每次 Amazon Data Firehose 向 HTTP 端点发送数据（初始尝试或重试）时，都会重新启动确认超时计数器并等待 HTTP 端点的确认。

即使重试持续时间已过，Amazon Data Firehose 仍会等待确认，直到收到确认或达到确认超时期限。如果确认超时，Amazon Data Firehose 会确定重试计数器是否还有剩余时间。如果有剩余时间，它将再次重试并重复该逻辑，直到收到确认或确定重试时间已到期。

如果不希望 Amazon Data Firehose 重试发送数据，请将此值设置为 0。

参数（可选）

Amazon Data Firehose 在每次 HTTP 调用中都包含这些键值对。这些参数可以帮助您识别和组织目的地。

缓冲提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目标位置的建议缓冲区大小因服务提供商而异。

为 Snowflake 配置目的地设置

本节介绍将 Snowflake 用于您的目的地的选项。

> **Note**
>
> Firehose 与 Snowflake 的集成已在美国东部（弗吉尼亚北部）、美国西部（俄勒冈）、欧洲（爱尔兰）、美国东部（俄亥俄州）、亚太地区（东京）、欧洲（法兰克福）、亚太地区（新加坡）、亚太地区（首尔）和亚太地区（悉尼）、亚太地区（孟买）、欧洲（伦敦）、南美洲（圣保罗）、加拿大（中部）、欧洲（中部）、欧洲（悉尼）、亚太地区（孟买）、欧洲（伦敦）、南美洲（圣保罗）、加拿大（中部）、欧洲（中部）、欧洲（中部）巴黎）、亚太地区（大阪）、欧洲（斯德哥尔摩）、亚太地区（雅加达）。 AWS 区域

连接设置

- 提供下列字段的值：

  Snowflake 账户 URL

  指定 Snowflake 账户 URL。例如：`xy12345.us-east-1.aws.snowflakecomputing.com`。有关如何确定您的账户 URL 的信息，请参阅 Snowflake 文档。请注意，您不能指定端口号，而协议（https://）是可选的。

身份验证

您可以选择手动输入用户登录名、私钥和密码，也可以从 AWS Secrets Manager 中检索密钥以访问 Snowflake。

- 用户登录

  指定要用于加载数据的 Snowflake 用户。请确保用户有权将数据插入到 Snowflake 表中。

- 私有密钥

  指定私有密钥，以便使用 Snowflake 以 PKCS8 格式进行身份验证。此外，不要将 PEM 页眉和页脚作为私有密钥的一部分。如果密钥被分成多行，则请删除换行符。以下为私有密钥必须呈现的形式的示例。

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

  移除 KEY_CONTENT 中的空格然后将其提供给 Firehose。不需要换行符 header/footer 或换行符。

- Passphrase (密码)

  指定密码来解密已加密的私有密钥。如果私有密钥未加密，则可以将此字段留空。有关信息，请参阅 Using Key Pair Authentication & Key Rotation。

- 密钥

  从中 AWS Secrets Manager 选择一个包含 Snowflake 凭据的密钥。如果在下拉列表中未看到您的密钥，则请在 AWS Secrets Manager中创建一个。有关更多信息，请参阅 使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

角色配置

使用默认的 Snowflake 角色：如果选择此选项，则 Firehose 不会将任何角色传递给 Snowflake。假设默认角色用于加载数据。请确保默认角色具有将数据插入到 Snowflake 表中的权限。

使用自定义 Snowflake 角色：输入将数据加载到 Snowflake 表时由 Firehose 担任的非默认 Snowflake 角色。

Snowflake 连接

选项为私有或公有。

私有 VPCE ID（可选）

Firehose 与 Snowflake 私有连接的 VPCE ID。身份证格式为 com.amazonaws.vpce。 [区域] .vpce-svc-。*[id]*有关更多信息，请参阅 AWS PrivateLink & Snowflake。

> **Note**
>
> 如果您的 Snowflake 集群已启用私有链接，则请使用基于 `AwsVpceIds` 的网络策略来允许 Amazon Data Firehose 数据。Firehose 不需要您在 Snowflake 账户中配置基于 IP 的网络策略。启用基于 IP 的网络策略可能会干扰 Firehose 连接。如果您遇到需要基于 IP 的策略的边缘案例，则请通过提交支持工单与 Firehose 团队联系。有关您可以使用的 VPCE IDs 列表，请参阅。访问 VPC 中的 Snowflake

数据库配置

- 您必须指定以下设置，才能使用 Snowflake 作为 Firehose 流的目的地。
  - Snowflake 数据库：Snowflake 中的所有数据都保存在数据库中。
  - Snowflake 架构：每个数据库由一个或多个架构组成，这些架构是数据库对象（例如表和视图）的逻辑分组
  - Snowflake 表：Snowflake 中的所有数据都存储在数据库表中，逻辑结构为列和行的集合。

您 Snowflake 表的数据加载选项

- 使用 JSON 键作为列名
- 使用 VARIANT 列
  - 内容列名：在表中指定列名，其中必须加载原始数据。
  - 元数据列名（可选）：在表中指定列名，其中必须加载元数据信息。启用此字段后，您将在基于源类型的 Snowflake 表中看到以下列。

    Direct PUT 用作来源

```
{
```

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

Kinesis Data Stream 用作来源

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

Retry duration

Snowflake 服务问题导致打开通道或向 Snowflake 传输内容失败时，Firehose 重试的持续时间（0–7200 秒）。Firehose 以指数回退方式重试，直到重试持续时间结束。如果将重试持续时间设置为 0（零）秒，则 Firehose 在 Snowflake 失败时不会重试，而是将数据路由至 Amazon S3 错误存储桶。

缓冲区提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目标位置的建议缓冲区大小因服务提供商而异。有关更多信息，请参阅 配置缓冲提示。

为 Splunk 配置目的地设置

本节介绍使用将 Splunk 用于目的地的选项。

> **Note**
>
> Firehose 向配置了经典负载均衡器或应用程序负载均衡器的 Splunk 集群传输数据。

- 提供下列字段的值：

Splunk cluster endpoint

要确定端点，请参阅 Splunk 文档中的配置 Amazon Data Firehose 以将数据发送到 Splunk 平台。

Splunk endpoint type

在大多数情况下，请选择 `Raw endpoint`。选择`Event endpoint`是否使用按事件类型将数据发送 AWS Lambda 到不同的索引来预处理数据。有关要使用的端点的信息，请参阅 Splunk 文档中的配置 Amazon Data Firehose 以将数据发送到 Splunk 平台。

身份验证

您可以选择直接输入身份验证令牌，也可以从中检索密钥 AWS Secrets Manager 以访问 Splunk。

- 身份验证令牌

  要设置可从 Amazon Data Firehose 接收数据的 Splunk 端点，请参阅 Splunk 文档中的适用于 Amazon Data Firehose 的 Splunk 插件的安装和配置概述。保存在为该 Firehose 流设置端点时从 Splunk 获取的令牌，并在此处添加该令牌。

- 密钥

  从中选择一个包 AWS Secrets Manager 含 Splunk 身份验证令牌的密钥。如果在下拉列表中未看到您的密钥，则请在 AWS Secrets Manager中创建一个。有关更多信息，请参阅 使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

HEC acknowledgement timeout

指定 Amazon Data Firehose 等待来自 Splunk 的索引确认的时长。如果 Splunk 在超时之前未发送确认，Amazon Data Firehose 会将这种情况视为数据传输失败。然后，Amazon Data Firehose 会重试或将数据备份到 Amazon S3 存储桶，具体取决于您设置的重试持续时间值。

Retry duration

指定 Amazon Data Firehose 重试向 Splunk 发送数据的时长。

发送数据后，Amazon Data Firehose 先等待 Splunk 的确认。如果出现错误或未在确认超时期限内收到确认，Amazon Data Firehose 将启动重试持续时间计数器。它将不断重试，直到重试持续时间到期为止。重试持续时间到期后，Amazon Data Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

每次 Amazon Data Firehose 向 Splunk 发送数据（无论是初次尝试还是重试）时，都会重新启动确认超时计数器，并等待 Splunk 的确认。

即使重试持续时间已过，Amazon Data Firehose 仍会等待确认，直到收到确认或达到确认超时期限。如果确认超时，Amazon Data Firehose 会确定重试计数器是否还有剩余时间。如果有剩余时间，它将再次重试并重复该逻辑，直到收到确认或确定重试时间已到期。

如果不希望 Amazon Data Firehose 重试发送数据，请将此值设置为 0。

缓冲提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目的地的建议缓冲区大小因服务提供商而异。

为 Splunk Observability Cloud 配置目的地设置

本节介绍了使用 Splunk Observability Cloud 作为目标的选项。有关更多信息，请参阅 https://docs.splunk.com/observability/en/gdi/get-data-in/connect/aws/aws-apiconfig.html#--ap connect-to-aws-using i。the-splunk-observability-cloud

- 提供下列字段的值：

  云摄取端点 URL

  您可以在 Splunk Observability 控制台的“配置文件”>“组织”>“实时数据摄取端点”中，找到 Splunk Observability Cloud 的实时数据摄取 URL。

  身份验证

  您可以选择直接输入访问令牌，也可以从中检索密钥 AWS Secrets Manager 以访问 Splunk Observability Cloud。

  - 访问令牌

    从 Splunk Observability 控制台的设置下的访问令牌中，复制具有 INGEST 授权范围的 Splunk Observability 访问令牌。

  - 密钥

从中选择一个包 AWS Secrets Manager 含 Splunk 可观察性云访问令牌的密钥。如果在下拉列表中未看到您的密钥，则请在 AWS Secrets Manager中创建一个。有关更多信息，请参阅使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

内容编码

Amazon Data Firehose 使用内容编码来压缩请求的正文，然后再将请求发送到目的地。对请求的 enable/disable 内容编码选择 GZIP 或“禁用”。

Retry duration

指定 Amazon Data Firehose 重试将数据发送到所选 HTTP 端点的时长。

发送数据后，Amazon Data Firehose 先等待 HTTP 端点的确认。如果出现错误或未在确认超时期限内收到确认，Amazon Data Firehose 将启动重试持续时间计数器。它将不断重试，直到重试持续时间到期为止。重试持续时间到期后，Amazon Data Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

每次 Amazon Data Firehose 向 HTTP 端点发送数据（初始尝试或重试）时，都会重新启动确认超时计数器并等待 HTTP 端点的确认。

即使重试持续时间已过，Amazon Data Firehose 仍会等待确认，直到收到确认或达到确认超时期限。如果确认超时，Amazon Data Firehose 会确定重试计数器是否还有剩余时间。如果有剩余时间，它将再次重试并重复该逻辑，直到收到确认或确定重试时间已到期。

如果不希望 Amazon Data Firehose 重试发送数据，请将此值设置为 0。

参数（可选）

Amazon Data Firehose 在每次 HTTP 调用中都包含这些键值对。这些参数可以帮助您识别和组织目的地。

缓冲提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。目标位置的建议缓冲区大小因服务提供商而异。

为 Sumo Logic 配置目的地设置

本节介绍了使用 Sumo Logic 作为目标的选项。有关更多信息，请参阅 https://www.sumologic.com。

- 提供下列字段的值：

  HTTP 端点 URL

  按以下格式指定 HTTP 端点的 URL：`https://deployment name.sumologic.net/receiver/v1/kinesis/dataType/access token`。URL 必须是 HTTPS URL。

  内容编码

  Amazon Data Firehose 使用内容编码来压缩请求的正文，然后再将请求发送到目的地。对请求的 enable/disable 内容编码选择 GZIP 或 “禁用”。

  Retry duration

  指定 Amazon Data Firehose 重试向 Sumo Logic 发送数据的时长。

  发送数据后，Amazon Data Firehose 先等待 HTTP 端点的确认。如果出现错误或未在确认超时期限内收到确认，Amazon Data Firehose 将启动重试持续时间计数器。它将不断重试，直到重试持续时间到期为止。重试持续时间到期后，Amazon Data Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

  每次 Amazon Data Firehose 向 HTTP 端点发送数据（初始尝试或重试）时，都会重新启动确认超时计数器并等待 HTTP 端点的确认。

  即使重试持续时间已过，Amazon Data Firehose 仍会等待确认，直到收到确认或达到确认超时期限。如果确认超时，Amazon Data Firehose 会确定重试计数器是否还有剩余时间。如果有剩余时间，它将再次重试并重复该逻辑，直到收到确认或确定重试时间已到期。

  如果不希望 Amazon Data Firehose 重试发送数据，请将此值设置为 0。

  参数（可选）

  Amazon Data Firehose 在每次 HTTP 调用中都包含这些键值对。这些参数可以帮助您识别和组织目的地。

  缓冲提示

  Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。Elastic 目标的建议缓冲区大小因服务提供商而异。

为 Elastic 配置目的地设置

本节介绍了使用 Elastic 作为目标的选项。

- 提供下列字段的值：

  Elastic 端点 URL

  按以下格式指定 HTTP 端点的 URL：`https://<cluster-id>.es.<region>.aws.elastic-cloud.com`。URL 必须是 HTTPS URL。

  身份验证

  您可以选择直接输入 API 密钥，也可以从中检索密钥 AWS Secrets Manager 以访问 Elastic。

  - API 密钥

    联系 Elastic 以获取从 Firehose 向其服务传输数据所需的 API 密钥。

  - 密钥

    从中 AWS Secrets Manager 选择一个包含 Elastic API 密钥的密钥。如果在下拉列表中未看到您的密钥，则请在 AWS Secrets Manager中创建一个。有关更多信息，请参阅 使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中。

  内容编码

  Amazon Data Firehose 使用内容编码来压缩请求的正文，然后再将请求发送到目的地。选择 GZIP（这是默认选择的内容）或禁用您的请求的 enable/disable 内容编码。

  Retry duration

  指定 Amazon Data Firehose 重试向 Elastic 发送数据的时长。

  发送数据后，Amazon Data Firehose 先等待 HTTP 端点的确认。如果出现错误或未在确认超时期限内收到确认，Amazon Data Firehose 将启动重试持续时间计数器。它将不断重试，直到重试持续时间到期为止。重试持续时间到期后，Amazon Data Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

  每次 Amazon Data Firehose 向 HTTP 端点发送数据（初始尝试或重试）时，都会重新启动确认超时计数器并等待 HTTP 端点的确认。

即使重试持续时间已过，Amazon Data Firehose 仍会等待确认，直到收到确认或达到确认超时期限。如果确认超时，Amazon Data Firehose 会确定重试计数器是否还有剩余时间。如果有剩余时间，它将再次重试并重复该逻辑，直到收到确认或确定重试时间已到期。

如果不希望 Amazon Data Firehose 重试发送数据，请将此值设置为 0。

参数（可选）

Amazon Data Firehose 在每次 HTTP 调用中都包含这些键值对。这些参数可以帮助您识别和组织目的地。

缓冲提示

Amazon Data Firehose 在将传入数据传输到指定的目的地之前会进行缓冲。弹性目标的建议缓冲区大小为 1 MiB。

配置备份设置

Amazon Data Firehose 使用 Amazon S3 备份所有数据或仅备份失败的数据，并尝试将其传输到您选择的目的地。

> **Important**
>
> - 只有当 Firehose 流的源是 Direct PUT 或 Kinesis Data Streams 时，才支持备份设置。
> - 零缓冲功能仅适用于应用程序目的地，不适用于 Amazon S3 备份目的地。

如果您做出以下选择之一，则可以为 Firehose 流指定 S3 备份设置。

- 如果您将 Amazon S3 设置为 Firehose 流的目的地，并选择指定 Lamb AWS da 函数来转换数据记录，或者您选择转换您的 Firehose 流的数据记录格式。
- 如果您将 Amazon Redshift 设置为 Firehose 流的目的地，并选择指定一个 Lambda 函数来转换数据 AWS 记录。
- 如果你将以下任何服务设置为 Firehose 直播的目的地：亚马逊 OpenSearch 服务、Datadog、Dynatrace、HTTP Endpoint、 LogicMonitor MongoDB Cloud、New Relic、Splunk 或 Sumo Logic、Snowflake、Apache Iceberg Tables。

以下是 Firehose 流的备份设置。

- Amazon S3 中的源记录备份：如果您选择的目标是 S3 或 Amazon Redshift，则此设置指示您是要启用源数据备份还是将其禁用。如果将任何其他支持的服务（S3 或 Amazon Redshift 除外）设置为您选择的目标，则此设置指示您是要备份所有源数据还是仅备份失败的数据。
- S3 备份存储桶：该存储桶为 Amazon Data Firehose 备份数据的 S3 存储桶。
- S3 备份存储桶前缀：这是 Amazon Data Firehose 备份数据的前缀。
- S3 备份存储桶错误输出前缀：所有失败的数据都备份在此 S3 存储桶错误输出前缀中。
- 备份的缓冲提示、压缩和加密：Amazon Data Firehose 使用 Amazon S3 备份所有数据或仅备份失败的数据，并尝试将其传输到您选择的目的地。Amazon Data Firehose 在将传入数据传输（备份）到 Amazon S3 之前对其进行缓冲。您可以选择 1—128 的缓冲区大小 MiBs 和 60—900 秒的缓冲间隔。先满足的条件会触发向 Amazon S3 进行数据传输的操作。如果您启用数据转换，缓冲区间隔是指从 Amazon Data Firehose 接收转换数据的时间到数据传输到 Amazon S3 的时间。如果数据传输到目的地的速度落后于数据写入到 Firehose 流的速度，Amazon Data Firehose 会动态增加缓冲区大小以跟上速度。此操作有助于确保所有数据都传输到目标。
- S3 压缩：选择 GZIP、Snappy、Zip 或 Hadoop 兼容的 Snappy 数据压缩，或者不压缩数据。Snappy、Zip 和 Hadoop 兼容的 Snappy 压缩，不适用于以 Amazon Redshift 作为目的地的 Firehose 流。
- S3 文件扩展名格式（可选）：为传输到 Amazon S3 目标存储桶的对象指定文件扩展名格式。如果启用此功能，则指定的文件扩展名将覆盖数据格式转换或 S3 压缩功能（例如 .parquet 或 .gz）附加的默认文件扩展名。在将此功能与数据格式转换或 S3 压缩配合使用时，请确保您配置了正确的文件扩展名。文件扩展名必须以句点（.）开头，并且可以包含允许的字符：0-9a-z!-_.*‘()。文件扩展名不能超过 128 个字符。
- Firehose 支持使用 AWS Key Management Service (SSE-KMS) 的 Amazon S3 服务器端加密，用于加密亚马逊 S3 中交付的数据。您可以选择使用目标 S3 存储桶中指定的默认加密类型，也可以选择使用您拥有的 AWS KMS 密钥列表中的密钥进行加密。如果您使用密钥加密数据，则可以使用默认 AWS 托管 AWS KMS 密钥 (aws/s3) 或客户托管密钥。有关更多信息，请参阅使用 AWS KMS 托管密钥的服务器端加密 (SSE-KMS) 保护数据。

配置缓冲提示

Amazon Data Firehose 将传入的流数据缓冲到一定大小（缓冲大小），或缓冲一定时间（缓冲时间间隔）后再将其传输到指定的目的地。如果您想向 Amazon S3 传输大小最佳的文件并提高数据处理应用程序的性能，或者要调整 Firehose 的传输速率以匹配目标速度，则需要使用缓冲提示。

您可以在创建新的 Firehose 流时配置缓冲大小和缓冲区时间间隔，或者更新现有 Firehose 流的缓冲大小和缓冲时间间隔。缓冲大小以秒为单位测量 MBs ，缓冲间隔以秒为单位。但是，如果您为其中之一

指定值，您还必须为另一个提供值。满足的第一个缓冲条件将触发 Firehose 传输数据。如果未配置缓冲值，则使用默认值。

您可以通过 AWS 管理控制台、 AWS Command Line Interface或配置 Firehose 缓冲提示。 AWS SDKs对于现有流，您可以使用控制台中的编辑选项或使用 API 重新配置缓冲提示，使其具有适合您用例的值。UpdateDestination对于新直播，您可以使用控制台或 CreateDeliveryStreamAPI 将缓冲提示配置为创建新直播的一部分。要调整缓冲区大小，请在CreateDeliveryStream或 UpdateDestinationAPI 的目标特定`DestinationConfiguration`参数`IntervalInSeconds`中设置`SizeInMBs`和。

> **Note**
>
> - 缓冲区提示应用于分片或分区级别，而动态分区缓冲区提示则应用于流或主题级别。
> - 为了满足低延迟的实时使用案例，您可以使用零缓冲时间间隔提示。当您将缓冲时间间隔配置为零秒时，Firehose 不会缓冲数据，而是在几秒钟内传输数据。在将缓冲提示更改为较低的值之前，请咨询供应商，了解有关其目的地的 Firehose 建议缓冲提示。
> - 零缓冲功能仅适用于应用程序目的地，不适用于 Amazon S3 备份目的地。
> - 零缓冲功能不适用于动态分区。
> - 当您将缓冲区时间间隔配置为小于 60 秒以提供更低的延迟时，Firehose 会对 S3 目的地使用分段上传。由于 S3 目的地的分段上传，如果您选择的缓冲区时间间隔小于 60 秒，则会看到 S3 PUT API 成本会有所增加。

有关目的地特定缓冲提示范围和默认值，请参阅下表：

目标位置	缓冲大小（以 MB 为单位，括号中为默认值）	缓冲时间间隔（以秒为单位，括号中为默认值）
Amazon S3	1-128 (5)	0-900 (300)
Apache Iceberg 表	1-128 (5)	0-900 (300)
Amazon Redshift	1-128 (5)	0-900 (300)
OpenSearch 无服务器	1-100 (5)	0-900 (300)

目标位置	缓冲大小（以 MB 为单位，括号中为默认值）	缓冲时间间隔（以秒为单位，括号中为默认值）
OpenSearch	1-100 (5)	0-900 (300)
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)
弹性	1	0-900 (60)
Honeycomb	1-64 (15)	0-900 (60)
HTTP 端点	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)

配置高级设置

下一节包含 Firehose 流的高级设置的详细信息。

- 服务器端加密——Amazon Data Firehose 支持 AWS 使用密钥管理服务AWS (KMS) 的 Amazon S3 服务器端加密，用于加密亚马逊 S3 中交付的数据。有关更多信息，请参阅使用 KMS AWS 托管密钥的服务器端加密 (SSE-KMS) 保护数据。
- 错误日志记录：Amazon Data Firehose 将记录处理和传输相关的错误。此外，启用数据转换后，它可以记录 Lambda 调用并将数据传输错误发送到日志。 CloudWatch 有关更多信息，请参阅 使用日志监控亚马逊数据 Firehose CloudWatch。

  > **Important**
  >
  > 虽然是可选的，但强烈建议在创建 Firehose 流过程中启用 Amazon Data Firehose 错误日志记录。这种做法可确保在记录处理或传输失败的情况下，您可以访问错误详细信息。

- 权限：Amazon Data Firehose 使用 IAM 角色来获取 Firehose 流所需的所有权限。您可以选择创建自动分配所需权限的新角色，也可以选择为 Amazon Data Firehose 创建的现有角色。该角色用于向 Firehose 授予对各种服务的访问权限，包括您的 S3 存储桶、 AWS KMS 密钥（如果启用了数据加密）和 Lambda 函数（如果启用了数据转换）。控制台可创建带占位符的角色。有关更多信息，请参阅什么是 IAM？。

  > **Note**
  >
  > IAM 角色（包括占位符）根据您在创建 Firehose 流时选择的配置创建。如果您对 Firehose 流源或目的地进行任何更改，则必须手动更新 IAM 角色。

- 标签-您可以添加标签来组织 AWS 资源、跟踪成本和控制访问权限。

  如果您在 `CreateDeliveryStream` 操作中执行了标签，则 Amazon Data Firehose 会对 `firehose:TagDeliveryStream` 操作执行额外的授权，以验证用户是否具备创建标签的权限。如果您不提供此权限，则创建带有 IAM 资源标签的新 Firehose 流的请求将失败，并显示 `AccessDeniedException`，如下所示。

  ```
  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.
  ```

  以下示例演示了允许用户创建 Firehose 流并应用标签的策略。

选择备份和高级设置后，查看您的选择，然后选择创建 Firehose 流。

新的 Firehose 流在正在创建状态下需要一些时间才能使用。当您的 Firehose 流处于活动状态后，就可以从生产者向其发送数据。

使用示例数据测试 Firehose 流

您可以使用 AWS 管理控制台 来提取模拟股票行情数据。该控制台在您的浏览器中运行脚本，以将示例记录放入您的 Firehose 流中。这能让您测试 Firehose 流的配置，而无需生成测试数据。

下面是模拟数据的一个示例：

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

请注意，当您的 Firehose 流传输数据时，会产生标准 Amazon Data Firehose 费用，但在生成数据时不会产生任何费用。要停止产生这些费用，您可以随时从控制台停止示例数据流。

先决条件

在您开始之前，请先创建 Firehose 流。有关更多信息，请参阅 教程：从控制台创建 Firehose 流。

使用 Amazon S3 进行测试

执行以下步骤，将 Amazon Simple Storage Service（Amazon S3）作为目的地测试 Firehose 流。

要使用 Amazon S3 测试 Firehose 流

1. 打开 Firehose 控制台，网址为。https://console.aws.amazon.com/firehose/
2. 选择活跃的 Firehose 流。在开始发送数据之前，Firehose 流必须处于活动状态。
3. 在使用演示数据进行测试下，选择开始发送演示数据即可生成示例股票代码数据。
4. 按照屏幕上的说明验证是否正在将数据传输到 S3 存储桶。请注意，根据存储桶的缓冲配置，可能需要几分钟时间，新对象才能显示在您的存储桶中。
5. 在测试完成后，选择停止发送演示数据即可停止产生使用费用。

使用 Amazon Redshift 进行测试

执行以下步骤，将 Amazon Redshift 作为目的地以测试 Firehose 流。

要使用 Amazon Redshift 测试 Firehose 流

1. Firehose 流要求 Amazon Redshift 集群中存在一个表。通过 SQL 接口连接到 Amazon Redshift，并运行以下语句以创建一个接受示例数据的表。

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

2. 打开 Firehose 控制台，网址为。https://console.aws.amazon.com/firehose/
3. 选择活跃的 Firehose 流。在开始发送数据之前，Firehose 流必须处于活动状态。
4. 编辑 Firehose 流的目的地详细信息以指向新创建的 `firehose_test_table` 表。
5. 在使用演示数据进行测试下，选择开始发送演示数据即可生成示例股票代码数据。
6. 按照屏幕上的说明验证是否正在将数据传输到您的表。请注意，根据缓冲配置，可能需要几分钟时间，新行才能显示在您的表中。
7. 在测试完成后，选择停止发送演示数据即可停止产生使用费用。
8. 编辑 Firehose 流的目的地详细信息以指向另一个表。
9. （可选）删除 `firehose_test_table` 表。

使用 OpenSearch 服务进行测试

使用以下步骤以亚马逊 OpenSearch 服务为目标来测试您的 Firehose 直播。

使用服务测试 Firehose 直播 OpenSearch

1. 打开 Firehose 控制台，网址为。https://console.aws.amazon.com/firehose/
2. 选择活跃的 Firehose 流。在开始发送数据之前，Firehose 流必须处于活动状态。
3. 在使用演示数据进行测试下，选择开始发送演示数据即可生成示例股票代码数据。
4. 按照屏幕上的说明验证数据是否已传送到您的 OpenSearch 服务域。有关更多信息，请参阅《亚马逊 OpenSearch 服务开发者指南》中的在 OpenSearch 服务域中搜索文档。
5. 在测试完成后，选择停止发送演示数据即可停止产生使用费用。

使用 Splunk 进行测试

执行以下步骤，以便将 Splunk 作为目的地以测试 Firehose 流。

要使用 Splunk 测试 Firehose 流

1. 打开 Firehose 控制台，网址为。https://console.aws.amazon.com/firehose/
2. 选择活跃的 Firehose 流。在开始发送数据之前，Firehose 流必须处于活动状态。
3. 在使用演示数据进行测试下，选择开始发送演示数据即可生成示例股票代码数据。
4. 检查是否正在将数据传输到 Splunk 索引。Splunk 中的示例搜索词为 `sourcetype="aws:firehose:json"` 和 `index="name-of-your-splunk-index"`。有关如何在 Splunk 中搜索事件的更多信息，请参阅 Splunk 文档中的搜索手册。

   如果测试数据未出现在您的 Splunk 索引中，请检查您的 AmazonS3 存储桶中是否有失败事件。另请参阅数据未传输到 Splunk。
5. 在完成测试时，请选择 Stop sending demo data 以停止产生使用费用。

使用 Apache Iceberg 表进行测试

执行以下步骤，将 Apache Iceberg 表作为目的地以测试 Firehose 流。

要使用 Apache Iceberg 表测试 Firehose 流

1. 打开 Firehose 控制台，网址为。https://console.aws.amazon.com/firehose/
2. 选择活跃的 Firehose 流。在开始发送数据之前，Firehose 流必须处于活动状态。
3. 在使用演示数据进行测试下，选择开始发送演示数据即可生成示例股票代码数据。
4. 按照屏幕上的说明验证是否正在将数据传输到您的 Apache Iceberg 表。请注意，根据存储桶的缓冲配置，可能需要几分钟时间，新对象才能显示在您的存储桶中。
5. 如果测试数据未出现在您的 Apache Iceberg 表中，则请检查您的 AmazonS3 存储桶中是否有失败事件。
6. 在完成测试时，请选择 Stop sending demo data 以停止产生使用费用。

向 Firehose 流发送数据

本节介绍如何使用不同的数据源将数据发送到您的 Firehose 流。如果您不了解 Amazon Data Firehose，请花点时间熟悉 什么是 Amazon Data Firehose？ 中介绍的概念和术语。

> **Note**
>
> 某些 AWS 服务只能向位于同一区域的 Firehose 直播发送消息和事件。如果在为亚马逊 CloudWatch 日志、 CloudWatch 事件或配置目标时，您的 Firehose 直播未作为选项出现 AWS IoT，请验证您的 Firehose 直播是否与其他服务位于同一区域。有关每个区域的服务端点的信息，请参阅 Amazon Data Firehose 端点。

您可以将数据从以下数据来源发送到 Firehose 流。

主题

配置 Kinesis 代理以发送数据

Amazon Kinesis 代理是独立的 Java 软件应用程序，可用作参考实施，以显示如何收集数据并将其发送到 Firehose。此代理持续监控一组文件，并将新数据发送到您的 Firehose 流。代理显示您如何处理文件轮换、检查点操作并在失败时重试。它向您显示如何以可靠、及时且简单的方法传输您的数据。它还显示了如何发布 CloudWatch指标以更好地监控流媒体过程并对其进行故障排除。要了解更多信息，请访问 awslabs/ amazon-kinesis-agent。

默认情况下，会基于换行符 ('\n') 分析每个文件中的记录。但是，也可以将代理配置为分析多行记录（请参阅 指定代理配置设置）。

您可以在基于 Linux 的服务器环境（如 Web 服务器、日志服务器和数据库服务器）上安装此代理。在安装代理后，通过指定要监控的文件和数据的 Firehose 流来配置代理。在配置好代理之后，代理将持续从这些文件中收集数据并以可靠的方式将数据发送到 Firehose 流。

先决条件

在开始使用 Kinesis 代理之前，请确保您满足以下先决条件。

- 您的操作系统必须是 Amazon Linux 或 Red Hat Enterprise Linux 7 或更高版本。
- 代理 2.0.0 或更高版本使用 JRE 1.8 或更高版本运行。代理 1.1.x 使用 JRE 1.7 或更高版本运行。
- 如果您使用 Amazon EC2 运行代理，请启动 EC2 实例。
- 您指定的 IAM 角色或 AWS 证书必须具有执行 Amazon Data Firehose PutRecordBatch操作的权限，代理才能将数据发送到您的 Firehose 流。如果您为代理启用 CloudWatch 监控，则还需要执行该 CloudWatch PutMetricData操作的权限。有关更多信息，请参阅使用 Amazon Data Firehose 控制访问权限监控 Kinesis 代理运行状况、和 Amazon 的身份验证和访问控制 CloudWatch。

管理 AWS 凭证

使用以下方法之一管理您的 AWS 证书：

- 创建自定义凭证提供程序。有关更多信息，请参阅 the section called “创建自定义凭证提供程序”。
- 当您启动您的 EC2 实例时指定该 IAM 角色。
- 在配置代理时指定 AWS 凭据（参见下方配置表`awsSecretAccessKey`中的`awsAccessKeyId`和条目the section called “指定代理配置设置”）。
- 编辑`/etc/sysconfig/aws-kinesis-agent`以指定您的 AWS 地区和 AWS 访问密钥。
- 如果您的 EC2 实例位于不同的 AWS 账户中，请创建一个 IAM 角色以提供对 Amazon Data Firehose 服务的访问权限。在配置代理时指定该角色（参见 Assume Learn 和 IdassumeRoleExternal）。使用前面的方法之一来指定另一个账户中有权担任此角色的用户的 AWS 证书。

创建自定义凭证提供程序

您可以创建自定义凭证提供程序，并在以下配置设置中为 Kinesis 代理提供其类名和 jar 路径：`userDefinedCredentialsProvider.classname` 和 `userDefinedCredentialsProvider.location`。有关这两个配置设置的说明，请参阅the section called “指定代理配置设置”。

要创建自定义凭证提供程序，请定义一个实现 `AWS CredentialsProvider` 接口的类，如下例所示。

```
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() {
    }
}
```

您的类必须有一个不带参数的构造函数。

AWS 定期调用刷新方法以获取更新的凭证。如果希望凭证提供程序在其整个生命周期内提供不同的凭证，请在此方法中包含用于刷新凭证的代码。或者，如果您需要提供静态（不更改）凭证的凭证提供程序，则可以将此方法保留为空。

下载并安装代理

首先，连接到您的实例。有关详细信息，请参阅《Amazon EC2 用户指南》中的连接到您的实例。如果您在连接时遇到问题，请参阅《Amazon EC2 用户指南》中的排查 Amazon EC2 Linux 实例的连接问题。

接下来，请使用以下方法之一安装代理。

- 要从 Amazon Linux 存储库设置代理

  此方法仅适用于 Amazon Linux 实例。使用以下命令：

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

  Agent v 2.0.0 或更高版本安装在装有 Amazon Linux 2 () AL2 操作系统的计算机上。此代理版本需要安装 Java 1.8 或更高版本。如果尚未安装所需的 Java 版本，代理安装程序将会安装。有关亚马逊 Linux 2 的更多信息，请参阅https://aws.amazon.com/amazon-linux-2/。

- 从 Amazon S3 存储库设置代理

  此方法从公开可用的存储库安装代理，因此适用于 Red Hat Enterprise Linux 以及 Amazon Linux 2 实例。使用以下命令下载并安装最新版本的代理 2.x.x：

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

  要安装特定版本的代理，请在命令中指定版本号。例如，以下命令将安装代理 2.0.1。

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

  如果您使用的是 Java 1.7，但不想升级，则可以下载与 Java 1.7 兼容的代理 1.x.x。例如，要下载代理 1.1.6，可使用以下命令：

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

  您可以使用以下命令下载最新代理

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

- 从 GitHub repo 中设置代理

  1. 首先，确保您已安装所需的 Java 版本，具体取决于代理版本。
  2. 从 awslabs amazon-kinesis-agent GitHub /存储库下载代理。
  3. 导航到下载目录并运行以下命令来安装代理：

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

- 在 Docker 容器中设置代理

  Kinesis 代理可以在容器中运行，也可以通过 amazonlinux 容器库运行。使用以下 Dockerfile，然后运行 `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"]
```

配置并启动代理

配置并启动代理

1. 打开并编辑配置文件（如果使用默认文件访问权限，则以超级用户的身份来执行）：`/etc/aws-kinesis/agent.json`

   在此配置文件中，指定代理从中收集数据的文件（`"filePattern"`）以及代理将数据发送到的 Firehose 流的名称（`"deliveryStream"`）。文件名是一种模式，并且代理能够识别文件轮换。每秒内您轮换使用文件或创建新文件的次数不能超过一次。代理使用文件创建时间戳来确定要跟踪并送入 Firehose 流中的文件。如果每秒创建新文件或轮换使用文件的次数超过一次，代理将无法正确区分这些文件。

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

   默认 AWS 区域为`us-east-1`。如果使用的是其他区域，请将 `firehose.endpoint` 设置添加到配置文件，为区域指定终端节点。有关更多信息，请参阅 指定代理配置设置。

2. 手动启动代理：

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

3. （可选）将代理配置为在系统启动时启动：

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

现在，代理作为系统服务在后台运行。它会持续监控指定的文件，并将数据发送到指定的 Firehose 流。代理活动记录在 /var/log/aws-kinesis-agent/aws-kinesis-agent.log 中。

指定代理配置设置

代理支持两个必需的配置设置，即 filePattern 和 deliveryStream，以及可用于其他功能的可选配置设置。您可以在 /etc/aws-kinesis/agent.json 中指定必需配置设置和可选配置设置。

只要您更改配置文件，就必须使用以下命令停止再启动代理：

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

或者，您也可以使用以下命令：

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

一般的设置配置如下。

配置设置	说明
assumeRoleARN	用户应承担的角色的 Amazon 资源名称 (ARN)。有关更多信息，请参阅 IAM 用户指南中的使用 IAM 角色委派跨 AWS 账户访问权限。
assumeRoleExternalId	确定谁可以担任该角色的可选标识符。有关更多信息，请参阅《IAM 用户指南》中的如何使用外部 ID。
awsAccessKeyId	AWS 覆盖默认凭证的访问密钥 ID。此设置优先于所有其他凭证提供程序。
awsSecretAccessKey	AWS 覆盖默认凭证的密钥。此设置优先于所有其他凭证提供程序。

配置设置	说明
cloudwatch.emitMetrics	CloudWatch 如果已设置，则允许代理向其发送指标 (true)。 默认：True
cloudwatch.endpoint	的区域终端节点 CloudWatch。 默认值：monitoring.us-east-1.amazonaws.com
firehose.endpoint	Amazon Data Firehose 的区域端点。 默认值：firehose.us-east-1.amazonaws.com
sts.endpoint	AWS 安全令牌服务的区域终端节点。 默认值：https://sts.amazonaws.com
userDefinedCredentialsProvider.classname	如果定义自定义凭证提供程序，请使用此设置提供其完全限定类名。不要在类名末尾包含 .class。
userDefinedCredentialsProvider.location	如果定义自定义凭证提供程序，请使用此设置指定包含自定义凭证提供程序的 jar 的绝对路径。代理还在以下位置查找 jar 文件：/usr/share/aws-kinesis-agent/lib/ 。

流配置设置如下。

配置设置	说明
aggregatedRecordSizeBytes	要使代理聚合记录，然后在一个操作中将这些记录放入 Firehose 流，请指定此设置。将此项设置为所需的大小，您希望聚合记录达到该大小后，代理将记录放入 Firehose 流。 默认值：0（不聚合）

配置设置	说明
dataProcessingOptions	在将每个被分析的记录发送到 Firehose 流之前应用于这些记录的处理选项的列表。这些处理选项按指定的顺序执行。有关更多信息，请参阅 使用代理对数据进行预处理。
deliveryStream	[必需] Firehose 流名称。
filePattern	[必需] 需要由代理监控的文件的 Glob。与此模式匹配的任何文件都会自动由代理挑选出来并进行监控。对于匹配此模式的所有文件，请向 aws-kinesis-agent-user 授予读取权限。对于包含这些文件的目录，请向 aws-kinesis-agent-user 授予读取和执行权限。 **Important** 代理将选择与此模式匹配的任何文件。为了确保代理不会选择意外的记录，请仔细选择此模式。
initialPosition	开始解析文件的初始位置。有效值为 START_OF_FILE 和 END_OF_FILE 。 默认值：END_OF_FILE
maxBufferAgeMillis	代理在将数据发送到 Firehose 流之前缓冲数据的最长时间（以毫秒计）。 值范围：1,000-900,000（1 秒到 15 分钟） 默认值：60,000（1 分钟）
maxBufferSizeBytes	代理在将数据发送到 Firehose 流之前缓冲的数据的最大大小（以字节计）。 值范围：1-4,194,304（4MB） 默认值：4194304 (4 MB)

配置设置	说明
maxBuffer SizeRecords	代理在将数据发送到 Firehose 流之前缓冲数据的最大记录数。 值范围：1-500 默认值：500
minTimeBe tweenFile PollsMillis	代理轮询和分析受监控文件中是否有新数据的时间间隔（以毫秒计）。 值范围：1 或更大值 默认值：100
multiLine StartPattern	用于标识记录开始的模式。记录由与模式匹配的行以及与模式不匹配的任何以下行组成。有效值为正则表达式。默认情况下，日志文件中的每一个新行都被解析为一条记录。
skipHeaderLines	代理从受监控文件开头跳过分析的行数。 值范围：0 或更大值 默认值：0（零）
truncated RecordTer minator	在记录大小超过 Amazon Data Firehose 记录大小限制时，代理用来截断已解析记录的字符串。(1000 KB) 默认值：'\n'（换行符）

配置多个文件目录和流

通过指定多个流程配置设置，您可以配置代理以监控多个文件目录并将数据发送到多个流。在以下配置示例中，代理监控两个文件目录，并将数据分别发送到 Kinesis 数据流和 Firehose 流。您可以为 Kinesis Data Streams 和 Amazon Data Firehose 指定不同的端点，这样您的数据流和 Firehose 流就不需要位于同一区域。

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

有关在 Amazon Kinesis Data Streams 中使用代理的更多详细信息，请参阅使用 Kinesis 代理写入 Amazon Kinesis Data Streams。

使用代理对数据进行预处理

代理可以预处理从受监控文件分析的记录，然后再将这些记录发送到 Firehose 流。通过将 `dataProcessingOptions` 配置设置添加到您的文件流可以启用此功能。可以添加一个或多个处理选项，这些选项将按指定的顺序执行。

代理支持以下处理选项。由于代理是开源的，您可以进一步开发和扩展其处理选项。您可以从 Kinesis 代理下载代理。

处理选项

`SINGLELINE`

通过移除换行符和首尾的空格，将多行记录转换为单行记录。

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

`CSVTOJSON`

将记录从分隔符分隔的格式转换为 JSON 格式。

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

```
}
```

customFieldNames

[必需] 在每个 JSON 键值对中用作键的字段名称。例如，如果您指定 ["f1", "f2"]，则记录“v1, v2”将转换为 {"f1":"v1","f2":"v2"}。

delimiter

在记录中用作分隔符的字符串。默认值为逗号（,）。

LOGTOJSON

将记录从日志格式转换为 JSON 格式。支持的日志格式为 Apache Common Log、Apache Combined Log、Apache Error Log 和 RFC3164 Syslog。

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

logFormat

[必需] 日志条目格式。以下是可能的值：

- COMMONAPACHELOG – Apache 通用日志格式。默认情况下每个日志条目都为以下模式：“%{host} %{ident} %{authuser} [%{datetime}] \"%{request}\" %{response} %{bytes}”。
- COMBINEDAPACHELOG – Apache 组合日志格式。默认情况下每个日志条目都为以下模式：“%{host} %{ident} %{authuser} [%{datetime}] \"%{request}\" %{response} %{bytes} %{referrer} %{agent}”。
- APACHEERRORLOG – Apache 错误日志格式。默认情况下每个日志条目都为以下模式：“[%{timestamp}] [%{module}:%{severity}] [pid %{processid}:tid %{threadid}] [client: %{client}] %{message}”。
- SYSLOG— RFC3164 系统日志格式。默认情况下每个日志条目都为以下模式：“%{timestamp} %{hostname} %{program}[%{processid}]: %{message}”。

matchPattern

覆盖指定的日志格式的默认模式。如果日志条目使用自定义格式，则可以使用该设置提取日志条目中的值。如果指定 matchPattern，还必须指定 customFieldNames。

`customFieldNames`

在每个 JSON 键值对中用作键的自定义字段名称。您可以使用此设置定义从 `matchPattern` 中提取的值的字段名称，或覆盖预定义日志格式的默认字段名称。

Example：LOGTOJSON 配置

下面是一个转换为 JSON 格式的 Apache 通用日志条目的 LOGTOJSON 配置示例：

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

转换前：

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

转换后：

```
{"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：包含自定义字段的 LOGTOJSON 配置

下面是 LOGTOJSON 配置的另一个示例：

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

使用此配置设置时，上一个示例中的同一个 Apache 通用日志条目将如下转换为 JSON 格式：

```
{"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：转换 Apache 通用日志条目

以下流配置将一个 Apache 通用日志条目转换为 JSON 格式的单行记录：

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

Example：转换多行记录

以下流配置分析第一行以“[SEQUENCE=”开头的多行记录。每个记录先转换为一个单行记录。然后，将基于制表分隔符从记录中提取值。提取的值映射到指定的 customFieldNames 值，从而形成 JSON 格式的单行记录。

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

```
}
```

Example：具有匹配模式的 LOGTOJSON 配置

下面是一个转换为 JSON 格式的 Apache 通用日志条目的 LOGTOJSON 配置示例，其中省略了最后一个字段 (bytes)：

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

转换前：

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

转换后：

```
{"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"}
```

使用常用的代理 CLI 命令

下表提供了一组使用 AWS Kinesis 代理的常见用例和相应的命令。

使用案例	命令
系统启动时自动启动代理	`sudo chkconfig aws-kinesis-agent on`
检查代理的状态	`sudo service aws-kinesis-agent status`
停止代理	`sudo service aws-kinesis-agent stop`

使用案例	命令
从此位置读取代理的日志文件	`/var/log/aws-kinesis-agent/aws-kinesis-agent.log`
卸载代理	`sudo yum remove aws-kinesis-agent`

排查从 Kinesis 代理发送时出现的问题

此表提供了使用 Amazon Kinesis 代理时所遇到常见问题的问题排查信息和解决方案。

问题	解决方案
为什么 Kinesis 代理无法在 Windows 上运行？	适用于 Windows 的 Kinesis 代理与适用于 Linux 平台的 Kinesis 代理是不同的软件。
为什么 Kinesis 代理速度变慢且/或 `RecordSendErrors` 增加？	这通常是由于 Kinesis 节流造成的。查看 Kinesis Data Streams 的 `WriteProvisionedThroughputExceeded` 指标或 Firehose 流的 `ThrottledRecords` 指标。这些指标从 0 开始的任何增加都表示需要增加流限制。有关更多信息，请参阅 Kinesis Data Stream limits 和 Firehose streams。 排除节流后，查看 Kinesis 代理是否配置为跟踪大量小文件。Kinesis 代理跟踪新文件时会有延迟，因此 Kinesis 代理应跟踪少量大文件。尝试将您的日志文件合并为大文件。
如何解决 `java.lang.OutOfMemoryError` 异常？	当 Kinesis 代理没有足够的内存来处理当前的工作负载时会发生此情况。尝试增加 `/usr/bin/start-aws-kinesis-agent` 中的 `JAVA_START_HEAP` 和 `JAVA_MAX_HEAP`，并重新启动代理。
如何解决 `IllegalStateException : connection pool shut down` 异常？	Kinesis 代理没有足够的连接来处理当前的工作负载。尝试在 `/etc/aws-kinesis/agent.json` 的常规代理配置设置中增加 `maxConnections` 和 `maxSendingThreads`。这些字段的默认值是可用运行时系统处理

问题	解决方案
	器的 12 倍。有关高级代理配置设置的更多信息，请参见 AgentConfiguration.java。
如何调试 Kinesis 代理的其他问题？	可在 `/etc/aws-kinesis/log4j.xml` 中启用 DEBUG 级别日志。
我应该如何配置 Kinesis 代理？	`maxBufferSizeBytes` 越小，Kinesis 代理发送数据的频率就越高。这可能是好事，因为这减少了记录的传输时间，但也增加了每秒对 Kinesis 的请求。
为什么 Kinesis 代理会发送重复记录？	这是由于文件跟踪中的错误配置造成的。确保每个 `fileFlow's filePattern` 只匹配一个文件。如果正在使用的 `logrotate` 模式处于 `copytruncate` 模式，也可能发生这种情况。尝试将模式更改为默认模式或创建模式以避免重复。有关处理重复记录的更多信息，请参阅处理重复记录。

使用 AWS SDK 发送数据

您可以使用 Amazon Data Firehose API 通过适用于 Java、.NET、Node.js、Python 或 Ruby 的 AWS SDK 将数据发送到 Firehose 流。如果您不了解 Amazon Data Firehose，请花点时间熟悉 什么是 Amazon Data Firehose？ 中介绍的概念和术语。有关更多信息，请参阅开始使用 Amazon Web Services 开发。

这些示例并非可直接用于生产的代码，因为它们不会检查所有可能的异常，或者不会考虑到所有可能的安全或性能问题。

Amazon Data Firehose API 提供了两种用于向你的 Firehose 直播发送数据的操作：和。PutRecordPutRecordBatch `PutRecord()`在一个呼叫中发送一条数据记录，并且`PutRecordBatch()`可以在一个呼叫中发送多条数据记录。

使用单次写入操作 PutRecord

写入数据时，只需要 Firehose 流名称和字节缓冲区（<=1000 KB）。由于 Amazon Data Firehose 在将文件加载到 Amazon S3 之前会批量处理多条记录，因此可能需要添加记录分隔符。要以一次一条记录的方式向 Firehose 流写入数据，请使用以下代码：

```
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);
```

有关更多代码上下文，请参阅 AWS SDK 中包含的示例代码。有关请求和响应语法的信息，请参阅 Firehose API Operations 中的相关主题。

使用 Batch 写入操作 PutRecordBatch

写入数据时，只需要 Firehose 流名称和记录列表。由于 Amazon Data Firehose 在将文件加载到 Amazon S3 之前会批量处理多条记录，因此可能需要添加记录分隔符。要以批量方式向 Firehose 流写入数据记录，请使用以下代码：

```
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();
```

有关更多代码上下文，请参阅 AWS SDK 中包含的示例代码。有关请求和响应语法的信息，请参阅 Firehose API Operations 中的相关主题。

向 Firehose 发送 CloudWatch 日志

CloudWatch 可以使用 CloudWatch 订阅过滤器将日志事件发送到 Firehose。有关更多信息，请参阅使用 Amazon Data Firehose 的订阅筛选条件。

CloudWatch 日志事件以压缩的 gzip 格式发送到 Firehose。如果您想将解压缩后的日志事件传送到 Firehose 目标，可以使用 Firehose 中的解压缩功能自动解压缩日志。 CloudWatch

> **Important**
>
> 目前，Firehose 不支持将 CloudWatch 日志传送到亚马逊 OpenSearch 服务目标，因为亚马逊 CloudWatch 将多个日志事件合并到一个 Firehose 记录中，而亚马逊 OpenSearch 服务无法在一条记录中接受多个日志事件。作为替代方案，您可以考虑在 CloudWatch 日志中使用亚马逊 OpenSearch 服务的订阅筛选条件。

解压缩 CloudWatch 日志

如果您使用 Firehose 传送 CloudWatch 日志，并希望将解压缩后的数据传输到 Firehose 直播目标，请使用 Firehose 数据格式转换（Parquet、ORC）或动态分区。您必须为 Firehose 流启用解压功能。

您可以使用 AWS 管理控制台、 AWS Command Line Interface 或 AWS SDKs启用解压功能。

> **Note**
>
> 如果您在直播上启用解压缩功能，请将该流专门用于 CloudWatch 日志订阅过滤器，而不是用于Vended Logs。如果您在用于同时采集日志和已售 CloudWatch 日志的流上启用解压缩功能，则向Firehose提取Vended Logs将失败。此解压缩功能仅适用于 CloudWatch 日志。

解压日志后提取消息 CloudWatch

启用解压缩时，您可以选择同时启用消息提取。使用消息提取时，Firehose 会从解压缩的 CloudWatch 日志记录中筛选出所有元数据，例如所有者、日志组、日志流和其他元数据，并仅提供消息字段内的内容。如果您要将数据传输到 Splunk 目的地，则必须开启消息提取功能，Splunk 才能解析数据。以下是解压缩后的输出示例，无论是否具有消息提取。

图 1：没有消息提取的解压缩后的输出示例：

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

图 2：有消息提取的解压缩后的输出示例：

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

通过控制台对新的 Firehose 流启用解压缩功能

要对新的 Firehose 直播启用解压功能，请使用 AWS 管理控制台

1. 登录 AWS 管理控制台 并在 /kinesis 上打开 Kinesis 控制台。https://console.aws.amazon.com
2. 在导航窗格中选择 Amazon Data Firehose。
3. 选择创建 Firehose 流。
4. 在选择源和目的地下

   源

   您的 Firehose 流的源。请选择下列源之一：

   - Direct PUT：选择此选项可创建 Firehose 流，供生产者应用程序直接写入。有关与 Firehose 中的 Direct PUT 集成的 AWS 服务、代理以及开源服务的列表，请参阅此部分。

- Kinesis 流：选择此选项，以配置使用 Kinesis 数据流作为数据来源的 Firehose 流。然后，您可以使用 Firehose 从现有 Kinesis 数据流轻松读取数据，并将其加载到目的地。有关更多信息，请参阅 Writing to Firehose Using Kinesis Data Streams

目标位置

Firehose 流的目的地。选择下列选项之一：

- Amazon S3
- Splunk

5. 在 Firehose 流名称下，输入您的流名称。
6. （可选）在转换记录下：
   - 在 “从 Amazon CloudWatch Logs 解压源记录” 部分中，选择开启解压缩。
   - 如果要在解压缩后使用消息提取，则请选择开启消息提取。

在现有 Firehose 流上启用解压缩功能

本节提供了关于在现有 Firehose 流中启用解压缩的说明。它涵盖了两种场景——禁用 Lambda 处理的流和已经启用 Lambda 处理的流。以下各节概述了每种情况的 step-by-step程序，包括创建或修改 Lambda 函数、更新 Firehose 设置以及监控 CloudWatch 指标，以确保成功实施内置 Firehose 解压缩功能。

禁用 Lambda 处理时启用解压缩

要在禁用 Lambda 处理的情况下对现有 Firehose 流启用解压缩，必须先启用 Lambda 处理。此条件仅对现有流有效。以下步骤说明如何对未启用 Lambda 处理的现有流启用解压缩。

1. 创建一个 Lambda 函数。您可以创建虚拟记录通道，也可以使用此蓝图创建新的 Lambda 函数。
2. 更新您当前的 Firehose 流以启用 Lambda 处理并使用您创建的 Lambda 函数进行处理。
3. 使用新的 Lambda 函数更新流后，请返回 Firehose 控制台并启用解压缩功能。
4. 禁用您在步骤 1 中启用的 Lambda 处理。现在，您可以删除已在步骤 1 中创建的函数。

启用 Lambda 处理时启用解压缩

如果您已经有带有 Lambda 函数的 Firehose 流，要执行解压缩，可以将其替换为 Firehose 解压缩功能。在继续操作之前，请查看您的 Lambda 函数代码，以确认它仅执行解压缩或消息提取。您的

Lambda 函数的输出应与图 1 或图 2 中显示的示例类似。如果输出看起来相似，则您可以使用以下步骤替换 Lambda 函数。

1. 将您当前的 Lambda 函数替换为此蓝图。新的蓝图 Lambda 函数会自动检测传入的数据是被压缩还是解压缩。它只有在输入数据被压缩时才会执行解压缩。
2. 使用内置的 Firehose 选项开启解压缩功能，以进行解压缩。
3. 如果您的 Firehose 直播尚未启用，请启用该 CloudWatch 指标。监控指标 `CloudWatchProcessorLambda_IncomingCompressedData` 并等待至该指标变为零。这将确认发送到您的 Lambda 函数的所有输入数据均已解压缩，并且不再需要 Lambda 函数。
4. 移除 Lambda 数据转换，因为您不再需要此项来解压缩您的流。

在 Firehose 流中禁用解压缩

要禁用对数据流的解压缩，请使用 AWS 管理控制台

1. 登录 AWS 管理控制台 并在 /kinesis 上打开 Kinesis 控制台。https://console.aws.amazon.com
2. 在导航窗格中选择 Amazon Data Firehose。
3. 选择您要编辑的 Firehose 流。
4. 在 Firehose 流详细信息页面上，选择配置选项卡。
5. 在转换记录部分，选择编辑。
6. 在 “从 Amazon CloudWatch Logs 解压源记录” 下，清除 “开启解压缩”，然后选择 “保存更改”。

对 Firehose 中的解压缩进行问题排查

下表显示了 Firehose 如何处理数据解压缩和处理期间的错误，包括将记录传输到 S3 错误存储桶、日志记录错误和发出指标。其中还解释了未经授权的数据放置操作返回的错误消息。

问题	解决方案
如果在解压缩过程中出现错误，源数据会发生什么？	如果 Amazon Data Firehose 无法解压缩记录，则记录将按原样（以压缩格式）传输到您在 Firehose 流创建期间指定的 S3 错误存储桶。除了记录外，传输的对象还包括错误代码和错误消息，这些对象将传输到名为 `decompres`

问题	解决方案
	`sion-failed` 的 S3 存储桶前缀。记录解压缩失败后，Firehose 将继续处理其他记录。
成功解压缩后，如果处理管道出现错误，源数据会发生什么？	如果 Amazon Data Firehose 在解压后的处理步骤（例如，动态分区和数据格式转换）中出错，则记录将以压缩格式传输到您在创建 Firehose 流时指定的 S3 错误存储桶。除了记录外，传输的对象还包括错误代码和错误消息。
如果出现错误或异常，要如何通知您？	如果在解压缩过程中出现错误或异常，如果您配置 CloudWatch 日志，Firehose 会将错误消息 CloudWatch 记录到日志中。此外，Firehose 还会向您可以 CloudWatch 监控的指标发送指标。您还可以选择根据 Firehose 发出的指标来创建警报。
当`put`操作不是来自 CloudWatch 日志时会发生什么？	如果客户不是`puts`来自 CloudWatch Logs，则会返回以下错误消息： `Put to Firehose failed for AccountId: <accountID>, FirehoseName: <firehosename> because the request is not originating from allowed source types.`
Firehose 为解压缩功能发出了哪些指标？	Firehose 发出每条记录的解压缩指标。您应该选择周期（1 分钟）、统计数据（总和）、日期范围，以获取 `DecompressedRecords` 失败或成功或 `DecompressedBytes` 失败或成功的次数。有关更多信息，请参阅 CloudWatch 日志解压缩指标。

向 Fire CloudWatch hose 发送事件

您可以通过 CloudWatch 向事件规则添加目标，将亚马逊配置为向 Firehose 流发送 CloudWatch 事件。

为向现有 Firehose 直播发送事件的事件规则创建目标 CloudWatch

1. 登录 AWS 管理控制台 并打开 CloudWatch 控制台，网址为https://console.aws.amazon.com/cloudwatch/。
2. 选择 Create rule（创建规则）。
3. 在步骤 1：创建规则页面上，对于目标，选择添加目标，然后选择 Firehose 流。
4. 选择现有的 Firehose 流。

有关创建 CloudWatch 事件规则的更多信息，请参阅 Amazon CloudWatch 事件入门。

配置 AWS IoT 为向 Firehose 发送数据

您可以通过添加操作 AWS IoT 将信息配置为向 Firehose 直播发送信息。

要创建向现有 Firehose 流发送事件的操作

1. 在 AWS IoT 控制台中创建规则时，在创建规则页面的设置一个或多个操作下，选择添加操作。
2. 选择将消息发送到 Amazon Kinesis Firehose 流。
3. 选择 Configure action。
4. 对于流名称，选择现有的 Firehose 流。
5. 对于 Separator，选择要在记录之间插入的分隔符字符。
6. 对于 IAM 角色，选择现有的 IAM 角色，或选择创建新角色。
7. 选择添加操作。

有关创建 AWS IoT 规则的更多信息，请参阅 AWS IoT 规则教程。

在 Amazon Data Firehose 中转换源数据

Amazon Data Firehose 可以调用 Lambda 函数来转换传入的源数据，并将转换后的数据传输到目的地。当您创建 Firehose 流时，可以启用 Amazon Data Firehose 数据转换。

了解数据转换流

当您启用 Firehose 数据转换时，Firehose 会缓冲传入的数据。缓冲大小提示的范围在 0.2 MB 到 3 MB 之间。除 Splunk 和 Snowflake 以外，所有目的地的默认 Lambda 缓冲大小提示均为 1 MB。对于 Splunk 和 Snowflake，默认缓冲区提示为 256 KB。Lambda 缓冲间隔提示为 0 秒和 900 秒之间。除 Snowflake 以外，所有目的地的默认 Lambda 缓冲间隔提示均为 60 秒。对于 Snowflake，默认的缓冲提示间隔为 30 秒。要调整缓冲区大小，请使用ProcessorParameter调用的`BufferSizeInMBs`和`IntervalInSeconds`设置CreateDeliveryStream或 UpdateDestinationAPI 的ProcessingConfiguration参数。然后，Firehose 使用同步调用模式与每个缓冲批次同步调用指定的 Lambda 函数。 AWS Lambda 转换后的数据将从 Lambda 发送到 Firehose。当达到指定的目的地缓冲大小或缓冲间隔时（以先发生者为准），Firehose 会将其发送到目的地。

> **Important**
>
> Lambda 同步调用模式对请求和响应的负载大小限制均为 6MB。确保用于向函数发送请求的缓冲大小小于或等于 6 MB，并且函数返回的响应也不超过 6 MB。

Lambda 调用持续时间

Amazon Data Firehose 支持长达 5 分钟的 Lambda 调用时间。如果您的 Lambda 函数需要超过 5 分钟才能完成，则会收到以下错误：Firehose 在调用 Lambda 时遇到超时错误。 AWS 支持的最大函数超时为 5 分钟。

有关在发生此类错误时 Amazon Data Firehose 执行的操作的信息，请参阅the section called “处理数据转换失败”。

数据转换所需的参数

来自 Lambda 的所有转换记录必须包含以下参数，否则 Amazon Data Firehose 将会拒绝并将其视为数据转换失败。

For Kinesis Data Streams and Direct PUT

Lambda 中所有转换后的记录都需要以下参数。

- `recordId`：在调用期间，记录 ID 从 Amazon Data Firehose 传递到 Lambda。转换后的记录必须包含相同记录 ID。原始记录的 ID 和转换记录的 ID 之间如果有不匹配，将被视为数据转换失败。
- `result`：记录的数据转换的状态。可能的值为：`Ok`（记录成功转换）、`Dropped`（处理逻辑故意丢弃记录）和 `ProcessingFailed`（记录无法转换）。如果记录的状态为 `Ok` 或 `Dropped`，Amazon Data Firehose 会视为处理成功。否则，Amazon Data Firehose 会视为处理失败。
- `data`：转换后的数据负载（使用 base64 编码之后）。

以下是 Lambda 结果输出示例：

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

For Amazon MSK

Lambda 中所有转换后的记录都需要以下参数。

- `recordId`：在调用期间，记录 ID 从 Firehose 传递到 Lambda。转换后的记录必须包含相同记录 ID。原始记录的 ID 和转换记录的 ID 之间如果有不匹配，将被视为数据转换失败。
- `result`：记录的数据转换的状态。可能的值为：`Ok`（记录成功转换）、`Dropped`（处理逻辑故意丢弃记录）和 `ProcessingFailed`（记录无法转换）。如果记录的状态为 `Ok` 或 `Dropped`，则 Firehose 会认为它已成功处理。否则，Firehose 会视为处理失败。
- `KafkaRecordValue`：转换后的数据负载（使用 base64 编码之后）。

以下是 Lambda 结果输出示例：

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

```
}
```

支持的 Lambda 蓝图

这些蓝图演示了如何创建和使用 Lamb AWS da 函数来转换您的 Amazon Data Firehose 数据流中的数据。

查看控制台中可用的蓝图 AWS Lambda

1. 登录 AWS 管理控制台 并打开 AWS Lambda 控制台，网址为https://console.aws.amazon.com/lambda/。
2. 选择创建函数，然后选择使用蓝图。
3. 在蓝图字段中，搜索关键字 `firehose` 以查找 Amazon Data Firehose Lambda 蓝图。

蓝图列表：

- 处理发送到 Amazon Data Firehose 流的记录（Node.js、Python）

  此蓝图展示了如何使用 Lambda AWS 处理您的 Firehose 数据流中的数据的基本示例。

  最新发行日期：2016 年 11 月。

  发行说明：无。
- 发送到 Firehose 的处理 CloudWatch 日志

  已弃用此蓝图。请勿使用此蓝图。当解压缩后的 CloudWatch 日志数据超过 6MB（Lambda 限制）时，可能会产生高额费用。有关处理发送到 Firehose 的 CloudWatch 日志的信息，请参阅使用日志写入 Firehose。 CloudWatch
- 将系统日志格式的 Amazon Data Firehose 流记录转换为 JSON（Node.js）

  此蓝图显示了如何将 RFC3164 Syslog 格式的输入记录转换为 JSON。

  最新发行日期：2016 年 11 月。

  发行说明：无。

要查看中可用的蓝图 AWS Serverless Application Repository

1. 转到 AWS Serverless Application Repository。
2. 选择浏览所有应用程序。
3. 在 应用程序 字段中，搜索关键字 `firehose`。

您也可以在不使用蓝图的情况下创建 Lambda 函数。请参阅 AWS Lambda 入门。

处理数据转换失败

如果您的 Lambda 函数调用因网络超时或达到 Lambda 调用限制而失败，Amazon Data Firehose 默认会重试调用 3 次。如果调用失败，Amazon Data Firehose 将跳过该批记录。跳过的记录会被视为未被成功处理的记录。您可以使用或 `UpdateDestination` API 指定或覆盖重试选项。CreateDeliveryStream对于此类故障，您可以将调用错误记录到 Amazon Lo CloudWatch gs 中。有关更多信息，请参阅 使用日志监控亚马逊数据 Firehose CloudWatch。

如果记录的数据转换的状态为 `ProcessingFailed`，则 Amazon Data Firehose 会将记录视为处理失败。对于此类故障，您可以通过 Lambda 函数向 Amazon CloudWatch 日志发送错误日志。有关更多信息，请参阅AWS Lambda 开发者指南 AWS Lambda中的访问 Amazon CloudWatch 日志。

如果数据转换失败，则未被成功处理的记录会传输到您在 `processing-failed` 文件夹中的 S3 存储桶。这些记录具有如下格式：

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

`attemptsMade`

尝试的调用请求数。

`arrivalTimestamp`

Amazon Data Firehose 收到记录的时间。

`errorCode`

Lambda 返回的 HTTP 错误代码。

`errorMessage`

Lambda 返回的错误消息。

`attemptEndingTimestamp`

Amazon Data Firehose 停止尝试 Lambda 调用的时间。

`rawData`

经过 base64 编码的记录数据。

`lambdaArn`

Lambda 函数的 Amazon 资源名称（ARN）。

备份源记录

Amazon Data Firehose 可将所有未转换的记录同时备份到 S3 存储桶，同时将转换后的记录传输到目的地。您可以在创建或更新 Firehose 流时，启用源记录备份。在启用源记录备份之后，便不能禁用它。

在 Amazon Data Firehose 中对流数据进行分区

动态分区使您能够使用数据中的键（例如 `customer_id` 或 `transaction_id`）对 Firehose 中的流数据进行连续分区，然后将按这些键分组的数据传输到相应的 Amazon Simple Storage Service（Amazon S3）前缀。这使得使用各种服务（例如亚马逊雅典娜、亚马逊EMR、Amazon Redshift Spectrum 和亚马逊）可以更轻松地对亚马逊S3中的流数据进行高性能、具有成本效益的分析。 QuickSight此外，在需要额外处理的 AWS 用例中，在动态分区的流数据传输到 Amazon S3 之后，Glue 可以执行更复杂的提取、转换和加载 (ETL) 任务。

对数据进行分区可以最大限度地减少扫描的数据量，优化性能，并降低在 Amazon S3 上进行分析查询的成本, 还可以提高对数据的精细访问。传统上，Firehose 流用于捕获数据并将其加载到 Amazon S3 中。要对 Amazon S3-based Analytics 的流数据集进行分区，您需要先在 Amazon S3 存储桶之间运行分区应用程序，然后才能将数据提供给分析，这可能会变得复杂或昂贵。

通过动态分区，Firehose 使用动态或静态定义的数据键连续对传输中的数据进行分组，并按键将数据传输到各个 Amazon S3 前缀。这样可以缩短几分钟或几小时的洞察时间，还可以降低成本并简化架构。

主题

在 Amazon Data Firehose 中启用动态分区

您可以通过 Amazon Data Firehose 管理控制台、CLI 或 API 为 Firehose 流配置动态分区。

> **Important**
>
> 只有在创建新的 Firehose 流时，才能启用动态分区。对于未启用动态分区的现有 Firehose 流，无法启用动态分区。

有关如何在创建新的 Firehose 流时通过 Firehose 管理控制台启用和配置动态分区的详细步骤，请参阅 Creating an Amazon Firehose stream。当您开始执行为 Firehose 流指定目的地的任务时，请确保按照 配置目的地设置 部分中的步骤操作，因为目前只有使用 Amazon S3 作为目的地的 Firehose 流才支持动态分区。

在活动 Firehose 流上启用动态分区后，您可以通过添加新的分区键，或删除或更新现有分区键和 S3 前缀表达式来更新配置。更新后，Firehose 开始使用新的键和新的 S3 前缀表达式。

> **Important**
>
> 在 Firehose 流上启用动态分区后，就无法在此 Firehose 流上禁用。

了解分区键

通过动态分区，您可以根据分区键对数据进行分区，从 S3 流数据创建目标数据集。分区键使您能够根据特定值筛选流数据。例如，如果您需要根据客户 ID 和国家/地区筛选数据，则可以将 `customer_id` 的数据字段指定为一个分区键，将 `country` 的数据字段指定为另一个分区键。然后，指定表达式（使用支持的格式）来定义动态分区数据记录要传输到的 S3 存储桶前缀。

您可以使用以下方法来创建分区键。

- 内联解析：此方法使用 Firehose 内置支持机制（jq 解析器），从 JSON 格式的数据记录中提取用于分区的键。目前，我们仅支持 `jq 1.6` 版本。
- AWS Lambda 函数 — 此方法使用指定的 AWS Lambda 函数提取并返回分区所需的数据字段。

> **Important**
>
> 启用动态分区时，必须至少配置其中一种方法来对数据进行分区。您可以配置其中一种方法来指定分区键，也可以同时配置这两种方法。

使用内联解析创建分区键

要将内联解析配置为流数据的动态分区方法，必须选择要用作分区键的数据记录参数，并为每个指定的分区键提供一个值。

以下示例数据记录显示如何通过内联解析为其定义分区键。请注意，应以 Base64 格式对数据进行编码。您也可以参考 CLI 示例。

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

例如，您可以选择根据 `customer_id` 参数或 `event_timestamp` 参数对数据进行分区。这意味着您希望使用每条记录中的 `customer_id` 参数或 `event_timestamp` 参数的值来确定要向其传输传递记录的 S3 前缀。您也可以选择嵌套参数，例如表达式为 `.type.device` 的 `device`。您的动态分区逻辑可能取决于多个参数。

为分区键选择数据参数后，将每个参数映射到有效的 jq 表达式。下表显示了参数到 jq 表达式的映射：

参数	jq 表达式
`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")

在运行时系统，Firehose 使用上面的右列根据每条记录中的数据来评估参数。

使用创建分区密钥 AWS Lambda 函数

对于压缩或加密的数据记录，或除 JSON 以外的任何文件格式的数据，您可以使用集成 AWS Lambda 函数和自己的自定义代码来解压、解密或转换记录，以便提取和返回分区所需的数据字段。这是对现有

转换 Lambda 函数的扩展，该函数现已随 Firehose 一起提供。您可以转换、解析和返回数据字段，然后使用相同的 Lambda 函数进行动态分区。

以下是一个用 Python 编写的 Firehose 流处理 Lambda 函数示例，该函数从输入到输出重放每一条读取记录，并从记录中提取分区键。

```
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
```

以下是一个用 Go 编写的 Firehose 流处理 Lambda 函数示例，该函数从输入到输出重放每一条读取记录，并从记录中提取分区键。

```
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)
}
```

使用 Amazon S3 存储桶前缀传输数据

在创建使用 Amazon S3 作为目的地的 Firehose 流时，您必须指定 Amazon S3 存储桶，Firehose 将在该存储桶中传输数据。您可以使用 Amazon S3 存储桶前缀来组织存储在 S3 存储桶中的数据。Amazon S3 存储桶前缀类似于目录，可让您将类似的对象分组在一起。

通过动态分区，您的分区数据将传输到指定的 Amazon S3 前缀。如果未启用动态分区，则可以选择为 Firehose 流指定 S3 存储桶前缀。但是，如果选择启用动态分区，则必须指定 S3 存储桶前缀，Firehose 要向该存储桶传输分区数据。

在启用动态分区的每个 Firehose 流中，S3 存储桶前缀值由基于该 Firehose 流的指定分区键的表达式组成。再次使用上面的数据记录示例，您可以构建以下 S3 前缀值，该值由基于上面定义的分区键的表达式组成：

```
"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 会在运行时系统估算上述表达式。将匹配相同计算的 S3 前缀表达式的记录分组到一个数据集中。然后，Firehose 将每个数据集传输到计算的 S3 前缀。向 S3 传输数据集的频率由 Firehose 流缓冲区设置决定。因此，本示例中的记录将传输到以下 S3 对象键：

```
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
```

对于动态分区，您必须在 S3 存储桶前缀中使用以下表达式格式：`!{namespace:value}`，其中命名空间可以是 `partitionKeyFromQuery` 或 `partitionKeyFromLambda`，也可以是两者。如果使用内联解析为源数据创建分区键，则必须指定一个 S3 存储桶前缀值，该值由以下格式指定的表达式组成：`"partitionKeyFromQuery:keyID"`。如果使用 AWS Lambda 函数为源数据创建分区键，则必须指定一个 S3 存储桶前缀值，该值由以下格式指定的表达式组成：`"partitionKeyFromLambda:keyID"`。

> **Note**
>
> 您也可以使用 Hive 样式格式指定 S3 存储桶前缀值，例如 customer_id=! {分区KeyFromQuery:customer _id}。

有关更多信息，请参阅 Creating an Amazon Firehose stream 和 Custom Prefixes for Amazon S3 Objects 中的“Choose Amazon S3 for Your Destination”。

向 Amazon S3 传输数据时添加新的行分隔符

您可以启用新行分隔符，在传输到 Amazon S3 的对象的记录之间添加新行分隔符。这对解析 Amazon S3 中的对象很有帮助。在对聚合数据应用动态分区时，这也特别有用，因为作为解析过程的一部分，多记录解聚（必须先应用于聚合数据，然后才能对其进行动态分区）会从记录中删除新行。

向聚合数据应用动态分区

您可以将动态分区应用于聚合数据（例如，聚合到单个 `PutRecord` 和 `PutRecordBatch` API 调用中的多个事件、日志或记录），但必须先对这些数据进行解聚。您可以通过启用多记录解聚来取消聚合数据，即解析 Firehose 流中的记录并将其分离的过程。

多记录解聚可以是 `JSON` 类型，这意味着记录的分离是基于连续的 JSON 对象执行的。解聚也可以是 `Delimited` 类型，这意味着记录的分隔是基于指定的自定义分隔符执行的。该自定义分隔符必须是 base-64 编码的字符串。例如，如果要使用以下字符串作为自定义分隔符 `####`，则必须以 base-64 编码格式指定该字符串，将其转换为 `IyMjIw==`。按 JSON 或分隔符对记录进行解聚的上限为每条记录 500。

> **Note**
>
> 解聚 JSON 记录时，请确保您的输入仍以支持的 JSON 格式显示。JSON 对象必须在一行上，没有分隔符或只有换行符分隔（JSONL）。JSON 对象数组不是有效输入。
> 以下是正确输入的示例：`{"a":1}{"a":2}` and `{"a":1}\n{"a":2}`
> 以下是错误输入的示例：`[{"a":1}, {"a":2}]`

对于聚合数据，启用动态分区时，Firehose 会解析记录，并根据指定的多记录解聚类型在每次 API 调用中查找有效的 JSON 对象或分隔记录。

> **Important**
>
> 如果数据是聚合的，则只有在首次对数据进行解聚时才能应用动态分区。

> **Important**
>
> 在 Firehose 中使用数据转换功能时，将在数据转换之前应用解聚。进入 Firehose 的数据将按以下顺序处理：解聚 → 通过 Lambda 进行数据转换 → 分区键。

对动态分区错误进行问题排查

如果 Amazon Data Firehose 无法解析 Firehose 流中的数据记录，或者无法提取指定的分区键，或者无法计算 S3 前缀值中包含的表达式，那么这些数据记录将传输到 S3 错误存储桶前缀，在创建 Firehose 流时，您必须指定该前缀，以启用动态分区。S3 错误存储桶前缀包含 Firehose 无法传输到指定 S3 目的地的所有记录。这些记录是根据错误类型组织的。除了记录外，传输的对象还包括有关错误的信息，以帮助理解和解决错误。

如果要为 Firehose 流启用动态分区，则必须为此 Firehose 流指定 S3 错误存储桶前缀。如果您不想为 Firehose 流启用动态分区，则可以选择指定 S3 错误存储桶前缀。

用于动态分区的缓冲区数据

Amazon Data Firehose 将传入的流数据缓冲到一定大小，或缓冲一定时间后再将其传输到目的地。您可以在创建新的 Firehose 流时配置缓冲区大小和缓冲区时间间隔，或者更新现有 Firehose 流的缓冲区大小和缓冲区时间间隔。缓冲区大小的单位是 MB，缓冲区间隔的单位是秒。

> **Note**
>
> 零缓冲功能不适用于动态分区。

启用动态分区后，Firehose 会根据配置的缓冲区提示（大小和时间）在内部缓冲属于给定分区的记录，然后再将这些记录传输到 Amazon S3 存储桶。为了传输最大大小的对象，Firehose 在内部使用多级缓冲。因此，一批记录的端到端延迟可能是配置的缓冲提示时间的 1.5 倍。这会影响 Firehose 流的数据新鲜度。

活动分区计数是传输缓冲区内活动分区的总数。例如，如果动态分区查询每秒构造 3 个分区，并且您有一个缓冲区提示配置，每 60 秒触发一次传输，那么平均您将拥有 180 个活动分区。如果 Firehose 无法将分区中的数据传输到目的地，则该分区在传输缓冲区中将计为活动状态，直到可以传输为止。

当根据记录数据字段和 S3 前缀表达式将 S3 前缀计算为新值时，将创建一个新分区。为每个活动分区创建一个新缓冲区。具有相同计算 S3 前缀的每个后续记录都将传输到该缓冲区。

一旦缓冲区达到缓冲区大小限制或缓冲区时间间隔 Firehose 就会创建一个包含缓冲区数据的对象，并将其传输到指定的 Amazon S3 前缀。传输对象后，该分区的缓冲区和分区本身将被删除，并从活动分区计数中移除。

一旦分别满足每个分区的缓冲大小或间隔，Firehose 就会将每个缓冲区数据作为单个对象传输。一旦每个 Firehose 流的活动分区数量达到上限 500 个，Firehose 流中的其余记录就会被传送到指定的 S3 错误存储桶前缀（活动）。PartitionExceeded您可以使用 Amazon Data Firehose 限制表单申请将此配额增加到每个给定的 Firehose 流最多可达 2500 个活跃分区。如果您需要更多分区，则可以创建更多 Firehose 流，并在这些流之间分配活动分区。

在 Amazon Data Firehose 中转换输入数据格式

在将数据存储在 Amazon S3 之前，Amazon Data Firehose 可以将输入数据的格式从 JSON 转换为 Apache Parquet 或 Apache ORC。Parquet 和 ORC 是列式数据格式，与 JSON 等行式格式相比，其可节省空间并更快地启用查询。如果要转换除 JSON 之外的输入格式，例如逗号分隔值 (CSV) 或结构化文本，则可以先使用 AWS Lambda 将其转换为 JSON。有关更多信息，请参阅 转换源数据。

即使您在将记录发送到 Amazon Data Firehose 之前聚合了记录，也可以转换数据的格式。

Amazon Data Firehose 需要以下三个元素才能转换记录数据的格式：

Deserializer

Amazon Data Firehose 需要反串行化器才能读取输入数据的 JSON。您可以选择以下两种类型的反串行化器。

如要将多个 JSON 文档合并到同一记录中，请确保您的输入仍以支持的 JSON 格式显示。JSON 文档数组不是有效输入。

例如，这是正确的输入：`{"a": 1}{"b": 1}`；这是错误的输入：`[{"a":1}, {"a":2}]`。

- Apache Hive Json SerDe
- OpenX JSON SerDe

选择 JSON 反串行化器

SerDe如果您的输入 JSO N 包含以下格式的时间戳，请选择 OpenX JSON：

- yyyy-MM-dd't'hh: mm: ss [.S] 'Z'，其中分数最多可以有 9 位数字，例如，。`2017-02-07T15:13:01.39256Z`
- yyyy-[M]M-[d]d HH:mm:ss[.S]，其中小数最多有 9 位，例如：`2017-02-07 15:13:01.14`。
- 秒，以 Epoch 格式表示，例如：`1518033528`。
- 毫秒，以 Epoch 格式表示，例如：`1518033528123`。
- 浮点秒，以 Epoch 格式表示，例如：`1518033528.123`。

OpenX JSON SerDe 可以将句点 (.) 转换为下划线 (_)。它还可以在对 JSON 键进行反串行化前将其转换为小写。有关此反序列化器通过 Amazon Data Firehose 提供的选项的更多信息，请参阅打开。XJson SerDe

如果你不确定要选择哪个反序列化器，请使用 OpenX JSON SerDe，除非你有它不支持的时间戳。

如果您的时间戳格式与之前列出的格式不同，请使用 Apache Hiv e JSON。 SerDe选择此解串器后，您可以指定要使用的时间戳格式。为此，请遵循 Joda-Time `DateTimeFormat` 格式字符串的模式语法。有关更多信息，请参阅类 DateTimeFormat。

您还可以使用特殊值 `millis` 来解析时间戳（毫秒，以 Epoch 格式表示）。如果您不指定格式，Amazon Data Firehose 将默认使用 `java.sql.Timestamp::valueOf`。

Hive JSON SerDe 不允许执行以下操作：

- 列名称中的句点 (.)。
- 类型为 `uniontype` 的字段。
- 架构中具有数字类型但属于 JSON 中的字符串的字段。例如，如果架构是（整数），而 JSON 是`{"a":"123"}`，则 Hive SerDe 会给出错误。

Hive SerDe 不会将嵌套的 JSON 转换为字符串。例如，如果您有 `{"a":{"inner":1}}`，它不会将 `{"inner":1}` 视为字符串。

架构

Amazon Data Firehose 需要一个架构来确定如何解释该数据。使用 AWS Glue 在 AWS Glue Data Catalog中创建架构。然后，Amazon Data Firehose 会引用该架构并使用其解释您的输入数据。您可以使用同一架构来配置 Amazon Data Firehose 和分析软件。有关更多信息，请参阅《AWS Glue 开发者指南》中的 “填充 AWS Glue 数据目录”。

> **Note**
>
> 在 AWS Glue 数据目录中创建的架构应与输入数据结构相匹配。否则，转换后的数据将不会包含架构中未指定的属性。如果您使用嵌套 JSON，请在架构中使用可镜像 JSON 数据结构的 STRUCT 类型。有关如何使用 STRUCT 类型处理嵌套 JSON 的信息，请参阅本例。

> **Important**
>
> 对于未指定大小限制的数据类型，单行中所有数据的实际限制为 32 MBs。
> 如果您为 CHAR 或 VARCHAR 指定长度，则 Firehose 会在读取输入数据时按指定长度截断字符串。如果底层数据字符串较长，则将保持不变。

Serializer

Firehose 需要串行化器将数据转换为目标列式存储格式（Parquet 或 ORC）：您可以选择以下两种类型的串行化器之一。

- 兽人 SerDe
- 拼花地板 SerDe

选择串行化器

选择的串行化器取决于您的业务需求。要了解有关这两个序列化器选项的更多信息，请参阅 ORC SerDe 和 P arquet。 SerDe

启用记录格式转换

如果您启用记录格式转换，则无法将亚马逊数据 Firehose 目标设置为亚马逊 OpenSearch 服务、亚马逊 Redshift 或 Splunk。启用格式转换后，Amazon S3 就是您可用于 Firehose 流的唯一目的地。下一节将介绍如何从控制台和 Firehose API 操作启用记录格式转换。有关如何使用设置记录格式转换的示例 CloudFormation，请参阅AWS::DataFirehose:: DeliveryStream。

从控制台前期用记录格式转换

您可以在创建或更新 Firehose 流时在控制台上启用数据格式转换。启用数据格式转换后，Amazon S3 就是您可为 Firehose 流进行配置的唯一目的地。此外，启用格式转换时，系统将禁用 Amazon S3 压缩。但是，Snappy 压缩会作为自动转换过程的一部分自动进行。在这种情况下，Amazon Data Firehose 使用的 Snappy 的构造格式与 Hadoop 兼容。这意味着，您可以使用 Snappy 压缩的结果并在 Athena 中对这些数据运行查询。有关 Hadoop 所依赖的 Snappy 取景格式，请参阅.java。BlockCompressorStream

要对数据 Firehose 流启用数据格式转换

1. 登录并打开 Amazon Data Firehose 控制台，网址为。 AWS 管理控制台https://console.aws.amazon.com/firehose/
2. 选择要更新的 Firehose 流，或按照 教程：从控制台创建 Firehose 流 中的步骤创建新的 Firehose 流。
3. 在转换记录格式下，将记录格式转换设置为已启用。
4. 选择所需的输出格式。有关这两个选项的更多信息，请参阅 Apache Parquet 和 Apache ORC。
5. 选择一个 AWS Glue 表，为您的源记录指定架构。设置区域、数据库、表和表版本。

管理 Firehose API 的记录格式转换

如果你想让 Amazon Data Firehose 将你的输入数据格式从 JSON 转换为 Parquet 或 ORC，请在 extendedS3 或 Extended DestinationConfiguration S DataFormatConversionConfiguration3 中指定可选元素。DestinationUpdate如果您指定 DataFormatConversionConfiguration，则适用以下限制。

- 在中 BufferingHints，如果启用记录格式转换，则不能`SizeInMBs`将值设置为小于 64。此外，如果未启用格式转换，则默认值为 5。在启用格式转换后，该值将变为 128。
- 你必须`CompressionFormat`在 extendedS3 DestinationConfiguration 或 extendes3 中将设置为。DestinationUpdate `UNCOMPRESSED`CompressionFormat 的默认值为 `UNCOMPRESSED`。因此，您也可以在 ext en DestinationConfiguration dedS3 中将其保留为未指定。默认情况下，数据将使用 Snappy 压缩来作为串行化过程的一部分得到压缩。在这种情况下，Amazon Data Firehose 使用的 Snappy 的构造格式与 Hadoop 兼容。这意味着，您可以使用 Snappy 压缩的结果并在 Athena 中对这些数据运行查询。有关 Hadoop 所依赖的 Snappy 取景格式，请参阅.java。BlockCompressorStream当配置串行化器时，您可以选择其他类型的压缩。

处理数据格式转换错误

当 Amazon Data Firehose 无法解析或解串记录时（例如，当数据与架构不匹配时），会将记录写入 Amazon S3，且带有错误前缀。如果此写入失败，Amazon Data Firehose 会一直重试，同时阻止进一步传输。对于每条失败的记录，Amazon Data Firehose 会使用以下架构写入 JSON 文档：

```
{
  "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
  }
}
```

了解 Amazon Data Firehose 中的数据传输

当你向 Firehose 直播发送数据时，数据会自动传送到你选择的目的地。下表说明了到不同目的地的传输数据。

目标位置	Details
Amazon S3	对于到 Amazon S3 的数据传输，Firehose 根据 Firehose 流的缓冲配置连接多个传入记录。然后，将记录作为 Amazon S3 对象传输到 Amazon S3。默认情况下，Firehose 连接数据时不使用任何分隔符。如果您想在记录之间使用新行分隔符，则可以通过在 Firehose 控制台配置或 API 参数中启用此功能来添加新行分隔符。Firehose 和 Amazon S3 目的地之间的数据传输使用 TLS（HTTPS）加密。
Amazon Redshift	为向 Amazon Redshift 进行数据传输，Firehose 首先会将传入数据按上述格式传输到 S3 存储桶。然后，Firehose 会发出 Amazon Redshift COPY 命令，将数据从 S3 存储桶加载到 Amazon Redshift 预置集群或 Amazon Redshift Serverless 工作组。确保在 Amazon Data Firehose 将多个传入记录连接到 Amazon S3 对象后，可以将 Amazon S3 对象复制到您的 Amazon Redshift 预置集群或 Amazon Redshift Serverless 工作组。有关更多信息，请参阅 Amazon Redshift COPY 命令数据格式参数。
OpenSearch 服务和 OpenSearch 无服务器	为了向 OpenSearch 服务和 OpenSearch 无服务器传输数据，Amazon Data Firehose 会根据您的 Firehose 流的缓冲配置来缓冲传入的记录。然后，它会生成一个 OpenSearch 服务或 OpenSearch无服务器批量请求，将多条记录索引到您的 OpenSearch 服务集群或 OpenSearch 无服务器集合。在将记录发送到 Amazon Data Firehose 之前，确保记录采用 UTF-8 编码并展平为单行 JSON 对象。此外，必须将 OpenSearch 服务集群的`rest.action.multi.allow_explicit_index` 选项设置为 true（默认），才能使用为每条记录设置的显式索引来接受批量请求。有关更多信息，请参阅《Amazon OpenSearch 服务开发者指南》中的 OpenSearch 服务配置高级选项。
Splunk	为向 Splunk 进行数据传输，Amazon Data Firehose 会连接您发送的字节。如果要在您的数据中使用分隔符（如换行符），您必须自行插

目标位置	Details
	入这些分隔符。确保将 Splunk 配置为解析任何此类分隔符。要将传输到 S3 错误存储桶（S3 备份）的数据重新驱动回 Splunk，请按照 Splunk 文档中提到的步骤进行操作。
HTTP 端点	在向受支持的第三方服务提供商拥有的 HTTP 端点进行数据传输时，您可以使用集成的 Amazon Lambda 服务创建一个函数，将传入记录转换为与服务提供商集成预期的格式匹配的格式。请联系您选择其 HTTP 端点作为目标的第三方服务提供商，了解有关他们接受的记录格式的更多信息。
Snowflake	为了将数据传输到 Snowflake，Amazon Data Firehose 会在内部缓冲数据一秒钟，并使用 Snowflake 流式传输 API 操作将数据插入到 Snowflake。默认情况下，每秒都会刷新您插入的记录并将其提交到 Snowflake 表中。在您进行插入调用后，Firehose 会发出一个 CloudWatch 指标，用于衡量将数据提交到 Snowflake 所花费的时间。Firehose 目前仅支持将单个 JSON 项目作为记录有效负载，并且不支持 JSON 数组。请确保您的输入有效负载是有效的 JSON 对象，并且格式正确，没有任何额外的双引号、引号或转义字符。

每个 Firehose 目的地都有自己的数据传输频率。有关更多信息，请参阅 配置缓冲提示。

重复记录

Amazon Data Firehose 使用 at-least-once语义进行数据传输。在某些情况下（比如数据传输超时），如果原始数据传输请求最终通过，Amazon Data Firehose 的传输重试可能会导致重复。这适用于 Amazon Data Firehose 支持的所有目的地类型，但 Amazon S3 目的地、Apache Iceberg 表和 Snowflake 目的地除外。

主题

- 了解跨 AWS 账户和地区的配送情况
- 了解 HTTP 端点传输请求和响应规范
- 处理数据传输失败
- 配置 Amazon S3 对象名称格式
- 为 OpenSearch 服务配置索引轮换
- 暂停和恢复数据传输

了解跨 AWS 账户和地区的配送情况

Amazon Data Firehose 支持跨 AWS 账户向 HTTP 终端节点目标传输数据。您选择作为目标的 Firehose 流和 HTTP 端点可以属于不同的 AWS 账户。

Amazon Data Firehose 还支持跨 AWS 区域向 HTTP 终端节点目标传输数据。您可以将数据从一个区域的 Firehose 流传输到另一个 AWS AWS 区域的 HTTP 终端节点。您还可以将数据从 Firehose 流传输到 AWS 区域以外的 HTTP 终端节点目标，例如通过将 HTTP 端点网址设置为所需目的地，将数据传输到您自己的本地服务器。在这些情况下，您的传输成本将增加额外的传输费用。有关更多信息，请参阅“按需定价”页面中的数据传输部分。

了解 HTTP 端点传输请求和响应规范

要使 Amazon Data Firehose 成功将数据传输到自定义 HTTP 端点，这些端点必须接受请求，并使用某些 Amazon Data Firehose 请求和响应格式发送响应。本节介绍 Amazon Data Firehose 服务发送到自定义 HTTP 端点的 HTTP 请求的格式规范，以及 Amazon Data Firehose 服务预期的 HTTP 响应的格式规范。在 Amazon Data Firehose 使请求超时之前，HTTP 端点有 3 分钟的时间响应请求。Amazon Data Firehose 将不符合正确格式的响应视为传输失败。

请求格式

路径和 URL 参数

由您直接配置为单个 URL 字段的一部分。Amazon Data Firehose 会按配置发送，无需修改。仅支持 https 目标。URL 限制在传输流配置期间应用。

> **Note**
>
> 目前，HTTP 端点数据传输仅支持端口 443。

HTTP 标头 X-Amz-Firehose-Protocol-版本

此标头用于指示请求/响应格式的版本。目前唯一的版本是 1.0。

HTTP 标头- X-Amz-Firehose-Request-ID

此标头的值是一个不透明的 GUID，可用于调试和重复数据删除。如果可能的话，端点实现应记录这个标头的值，包括成功和失败的请求。在多次尝试同一请求时，请求 ID 保持不变。

HTTP 标头 - Content-Type

Content-Type 标头的值始终为 `application/json`。

HTTP 标头 - Content-Encoding

可以配置 Firehose 流，在发送请求时使用 GZIP 压缩正文。启用此压缩后，按照标准做法，Content-Encoding 标头的值将设置为 gzip。如果未启用压缩，则 Content-Encoding 标头就完全不存在。

HTTP 标头 - Content-Length

按照标准方式使用。

HTTP 标头- X-Amz-Firehose-Source-Arn:

以 ASCII 字符串格式表示的 Firehose 流的 ARN。ARN 对区域、 AWS 账户 ID 和直播名称进行编码。例如 `arn:aws:firehose:us-east-1:123456789:deliverystream/testStream`。

HTTP 标头- X-Amz-Firehose-Access-Key

此标头包含 API 密钥或其他凭证。在创建或更新传输流时，您可以创建或更新 API 密钥（又称授权令牌）。Amazon Data Firehose 将访问密钥的大小限制为 4096 字节。Amazon Data Firehose 不会尝试以任何方式解释此密钥。配置的密钥将逐字复制到此标头的值中。但是，如果您使用 Secrets Manager 来配置密钥，则密钥必须遵循特定的 JSON 对象格式：`{"api_key": "..."}`。

内容可以是任意的，可能表示 JWT 令牌或 ACCESS_KEY。如果端点需要多字段凭证（如用户名和密码），则应将所有字段的值以端点可理解的格式（JSON 或 CSV）一起存储在单个访问密钥中。如果原始内容是二进制的，则可以对此字段进行 base-64 编码。Amazon Data Firehose 不会修改对配置值的 and/or 编码，而是按原样使用内容。

HTTP 标头- X-Amz-Firehose-Common-属性

此标头将与整个请求相关的公共属性（元数据）带 and/or 到请求中的所有记录。在创建 Firehose 流时直接配置。此属性的值被编码为具有以下架构的 JSON 对象：

```
"$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
```

示例如下：

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

正文 - 最大大小

最大正文大小由您配置，压缩前最大可达 64MiB。

正文 - 架构

正文包含一个 JSON 文档，该文档具有以下 JSON 架构（用 YAML 编写）：

```
"$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
```

示例如下：

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

响应格式

出错时的默认行为

如果响应不符合以下要求，Firehose 服务器会将其视为无正文的 500 状态代码。

状态代码

HTTP 状态代码必须在 2XX、4XX 或 5XX 范围内。

Amazon Data Firehose 服务器不遵循重定向（3XX 状态代码）。只有响应代码 200 才被视为成功将记录传输到 HTTP/EP。响应代码 413（超出大小）被视为永久失败，并且记录批次不会发送到错误存储桶（如果已配置）。所有其他响应代码均被视为可重试的错误，并受到稍后介绍的回退重试算法的约束。

标头 - 内容类型

唯一可接受的内容类型是 application/json。

HTTP 标头 - Content-Encoding

不得使用 Content-Encoding。正文必须是未压缩状态。

HTTP 标头 - Content-Length

如果响应有正文，则必须有 Content-Length 标头。

正文 - 最大大小

响应正文的大小必须小于等于 1MiB。

```
"$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
```

示例如下：

```
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
}
```

错误响应处理

在所有错误情况下，Amazon Data Firehose 服务器都会使用指数回退算法重新尝试传输同一批记录。重试使用抖动系数为 (15%) 的初始退缩时间（1 秒）进行恢复，随后的每次重试都使用带有抖动的公式（initial-backoff-time * （乘数 (2) ^ retry_count））进行恢复。回退时间的最大间隔限制为 2 分钟。例如，在第 n 次重试时，回退时间为 = M MAX(120, 2^n) * random(0.85, 1.15)。

上式中指定的参数可能会发生变化。有关指数退避算法中使用的确切初始退缩时间、最大退避时间、乘数和抖动百分比，请参阅 Fi AWS rehose 文档。

在随后的每次重试中，传送记录的访问密钥 and/or 目标可能会根据更新后的 Firehose 流配置而发生变化。Amazon Data Firehose 服务尽力在重试中使用相同的请求 ID。最后一项功能可用于 HTTP 端点服务器进行重复数据删除。如果在允许的最长时间后仍未传输请求（基于 Firehose 流配置），则可以根据流配置选择将该批记录传输到错误存储桶。

示例

CWLog 来源请求的示例。

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

处理数据传输失败

每个 Amazon Data Firehose 目的地都有自己的数据传输失败处理流程。

在设置 Firehose 流时，对于许多目的地（例如 OpenSearch Splunk 和 HTTP 终端节点），您还会设置一个 S3 存储桶，用于备份未能交付的数据。有关 Firehose 在传输失败时如何备份数据的更多信息，请参阅此页的相关目的地部分。有关如何授予对 S3 存储桶的访问权限以备份无法传输的数据的更多信息，请参阅授予 Firehose 访问 Amazon S3 目的地的权限。当 Firehose（a）无法将数据传输到流目的地，并且（b）无法将数据写入传输失败的备份 S3 存储桶时，实际上会暂停流传输，直到数据可以传输到目的地或写入备份 S3 位置。

Amazon S3

到 S3 存储桶的数据传输可能会由于某些原因而失败。例如，存储桶可能已不存在、Amazon Data Firehose 承担的 IAM 角色可能无权访问存储桶、网络故障或类似事件。在此类情况下，Amazon Data Firehose 会在 24 小时内持续重试，直到传输成功。Amazon Data Firehose 的最长数据存储时间为 24 小时。如果数据传输失败超过 24 小时，数据将丢失。

传输数据到 S3 存储桶可能会由于各种原因而失败，例如：

- 存储桶不再存在。
- Amazon Data Firehose 担任的 IAM 角色无法访问存储桶。
- 网络问题。
- S3 错误，例如 HTTP 500s 或其他 API 故障。

在这些情况下，Amazon Data Firehose 将重新尝试传输：

- DirectPut 来源：重试最长可持续 24 小时。
- Kinesis Data Streams 或 Amazon MSK 来源：重试会无限期持续下去，具体取决于流中定义的保留策略。

只有在 Lambda 处理或 parquet 转换失败时，Amazon Data Firehose 才会将失败的记录传输到 S3 错误存储桶。其他失败情况将导致持续重新尝试传输到 S3，直到达到保留期为止。当 Firehose 成功将记录传输到 S3 时，它会创建一个 S3 对象文件；在部分记录失败的情况下，它会自动重新尝试传输，并使用成功处理的记录更新同一 S3 对象文件。

Amazon Redshift

对于 Amazon Redshift 目的地，您可以在创建 Firehose 流时指定重试持续时间（0–7200 秒）。

将数据传输到 Amazon Redshift 预置集群或 Amazon Redshift Serverless 工作组时，可能会因为以下几个原因而失败。例如，Firehose 流的集群配置不正确、集群或工作组正在维护或网络故障。在此类情况下，Amazon Data Firehose 会在指定的持续时间内重试，并跳过特定批次的 Amazon S3 对象。跳过的对象的信息会以清单文件的形式传输到您的 S3 存储桶中的 `errors/` 文件夹内，您可以利用该清单文件进行手动回填。有关如何使用清单文件手动 COPY 数据的信息，请参阅使用清单指定数据文件。

Amazon OpenSearch 服务和 OpenSearch 无服务器

对于 OpenSearch 服务和 OpenSearch 无服务器目标，您可以在 Firehose 直播创建期间指定重试持续时间（0—7200 秒）。

由于多种原因，向您的 OpenSearch 服务集群或 OpenSearch 无服务器集合传输数据可能会失败。例如，您的 Firehose 流可能存在错误的 OpenSearch 服务集群或 OpenSearch 无服务器集合配置、OpenSearch 正在维护的服务集群 OpenSearch 或无服务器集合、网络故障或类似事件。在此类情况下，Amazon Data Firehose 会在指定的持续时间内重试，并跳过特定的索引请求。跳过的文档会传输到您的 S3 存储桶中的 `AmazonOpenSearchService_failed/` 文件夹内，您可以利用它进行手动回填。

对于 OpenSearch 服务，每个文档都具有以下 JSON 格式：

```
{
    "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)"
}
```

对于 OpenSearch 无服务器，每个文档都采用以下 JSON 格式：

```
{
    "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

当 Amazon Data Firehose 向 Splunk 发送数据时，会等待 Splunk 的确认。如果出现错误或未在确认超时期限内收到确认，Amazon Data Firehose 将启动重试持续时间计数器。它将不断重试，直到重试持续时间到期为止。重试持续时间到期后，Amazon Data Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

每次 Amazon Data Firehose 向 Splunk 发送数据时，无论是首次尝试还是重试，都会重新启动确认超时计数器。然后，等待 Splunk 的确认。即使重试持续时间已过，Amazon Data Firehose 仍会等待确认，直到收到确认或确认超时。如果确认超时，Amazon Data Firehose 会检查重试计数器是否还有剩余时间。如果有剩余时间，它将再次重试并重复该逻辑，直到收到确认或确定重试时间已到期。

未收到确认并不是可能出现的唯一一类数据传输错误。有关其他类型的数据传输错误的信息，请参阅 Splunk 数据传输错误。如果重试持续时间大于 0，任何数据传输错误都会触发重试逻辑。

下面是一个错误记录示例。

```
{
  "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"
```

```
}
```

HTTP 端点目标

当 Amazon Data Firehose 向 HTTP 端点目的地发送数据时，会等待来自该目的地的响应。如果出现错误或未在响应超时期限内收到响应，Amazon Data Firehose 将启动重试持续时间计数器。它将不断重试，直到重试持续时间到期为止。重试持续时间到期后，Amazon Data Firehose 将这种情况视为数据传输失败，并将数据备份到 Amazon S3 存储桶。

每次 Amazon Data Firehose 向 HTTP 端点目的地发送数据时，无论是首次尝试还是重试，都会重新启动响应超时计数器。然后，等待来自 HTTP 端点目标的响应。即使重试持续时间已过，Amazon Data Firehose 仍会等待响应，直到收到响应或响应超时。如果响应超时，Amazon Data Firehose 会检查重试计数器是否还有剩余时间。如果还有时间，会再次重试并重复逻辑，直到收到响应或确定重试时间已过期。

未收到响应并不是唯一可能发生的数据传输错误类型。有关其他类型的数据传输错误的信息，请参阅 HTTP 端点数据传输错误。

下面是一个错误记录示例。

```
{
 "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

对于 Snowflake 目的地，在创建 Firehose 流时，您可以指定可选的重试持续时间（0–7200 秒）。重试持续时间的默认值是 60 秒。

向 Snowflake 表传输数据可能会失败，原因有很多，例如 Snowflake 目的地配置不正确、Snowflake 中断、网络故障等。重试策略不适用于不可重试的错误。例如，如果 Snowflake 因为表中缺少额外的

一列而拒绝了您的 JSON 有效负载，则 Firehose 不会尝试再次进行传输。相反，它会为所有因您的 S3 错误存储桶的 JSON 有效负载问题而导致的插入失败创建备份。

同样，如果由于角色、表或数据库错误而导致传输失败，则 Firehose 不会重试并将数据写入您的 S3 存储桶。重试持续时间仅适用于因 Snowflake 服务问题、临时网络故障等导致的失败。在此类情况下，Firehose 会在指定的持续时间内重试，然后再将它们传输到 S3。失败的记录将传输在 snowflake-failed/ 文件夹中，您可以使用该文件夹进行手动回填。

以下是您传输到 S3 的每条记录的 JSON 示例。

```
{
    "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"
}
```

配置 Amazon S3 对象名称格式

当 Firehose 向 Amazon S3 传输数据时，S3 对象键名称遵循以下格式 <evaluated prefix><suffix>，其中后缀的格式为 <Firehose stream name>-<Firehose stream version>-<year>-<month>-<day>-<hour>-<minute>-<second>-<uuid><file extension> <Firehose stream version>，以 1 开头，每进行一次 Firehose 流的配置更改就增加 1。您可以更改 Firehose 流配置（例如，S3 存储桶名称、缓冲提示、压缩以及加密）。您可以使用 Firehose 控制台或 UpdateDestinationAPI 操作来实现此目的。

对于 <evaluated prefix>，Firehose 在格式 YYYY/MM/dd/HH 中添加了默认的时间前缀。此前缀在存储桶中创建一个逻辑层级结构，其中，每个正斜杠（/）均在该层级结构中创建一个层级。您可以通过指定包含在运行时估算的表达式的自定义前缀来修改此结构。有关如何指定自定义前缀的信息，请参阅 Custom Prefixes for Amazon Simple Storage Service Objects。

默认情况下，时间前缀和后缀的时区为 UTC，但您可以将其更改为偏好的时区。例如，要使用日本标准时间代替 UTC，您可以在 API 参数设置 CustomTimeZone () AWS 管理控制台 中 Asia/Tokyo 将时区配置为。以下列表包含 Firehose 为 S3 前缀配置支持的时区。

支持的时区

以下是 Firehose 为 S3 前缀配置支持的时区列表。

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
```

除 <file extension> 之外，您不能更改后缀字段。启用数据格式转换或压缩时，Firehose 将根据配置附加文件扩展名。下表说明了 Firehose 附加的默认文件扩展名：

配置	文件扩展名
数据格式转换：Parquet	.parquet
数据格式转换：ORC	.orc
压缩：Gzip	.gz
压缩：Zip	.zip
压缩：Snappy	.snappy
压缩：Hadoop-Snappy	.hsnappy

您还可以在 Firehose 控制台或 API 中指定偏好的文件扩展名。文件扩展名必须以句点（.）开头，并且可以包含允许的字符：0-9a-z!-_.*‘()。文件扩展名不能超过 128 个字符。

> **Note**
>
> 指定文件扩展名时，它将覆盖 Firehose 在启用数据格式转换或压缩时添加的默认文件扩展名。

了解 Amazon S3 对象的自定义前缀

传输到 Amazon S3 的对象遵循 <evaluated prefix><suffix> 的名称格式。您可以指定包含在运行时进行估算的表达式的自定义前缀。您指定的自定义前缀将覆盖的默认前缀 `yyyy/MM/dd/HH`。

您可以在自定义前缀中使用以下形式的表达式：`!{namespace:value}`，其中 `namespace` 可以是以下形式之一，如以下各部分所述。

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

如果前缀以斜杠结束，则在 Amazon S3 存储桶中将显示为文件夹。有关更多信息，请参阅《亚马逊数据 FirehoseDeveloper指南》中的 Amazon S3 对象名称格式。

`timestamp` 命名空间

此命名空间的有效值是有效的 Java DateTimeFormatter 字符串的字符串。例如，对于 2018 年，表达式 `!{timestamp:yyyy}` 的计算结果为 `2018`。

在计算时间戳时，Firehose 将使用所写入 Amazon S3 对象中包含的最早记录的大致到达时间戳。

默认情况下，时间戳按 UTC 时间。但是，您可以指定偏好的时区。例如，如果您想使用日本标准时间而不是 UTC，则可以在 AWS 管理控制台 或 API 参数设置 (CustomTimeZone) 中将时区配置为。Asia/Tokyo 要查看支持的时区列表，请参阅 Amazon S3 Object Name Format。

如果您在同一前缀表达式中多次使用 `timestamp` 命名空间，则每个实例的计算结果为同一时刻。

`firehose` 命名空间

您可以将两个值用于此命名空间：`error-output-type` 和 `random-string`。下表介绍了其使用方法。

firehose 命名空间值

转换	说明	示例输入	输出示例	注意
error-output-type	根据您的 Firehose 直播的配置和失败原因，计算结果为以下字符串之一：{处理失败、-失败、splunk-failed、splunk-failed AmazonOpenSearchService、、}。format-conversion-failed http-endpoint-failed 如果您在同一表达式中多次使用它，则每个实例的计算结果为同一错误字符串。	myPrefix/result=!{firehose:error-output-type}/!{timestamp:yyyy/MM/dd}	myPrefix/result=processing-failed/2018/08/03	该 error-output-type值只能在 ErrorOutputPrefix字段中使用。
random-string	计算结果为 11 个字符的随机字符串。如果您在同一表达式中多次使用它，则每个实例的计算结果为一个新的随机字符串。	myPrefix/!{firehose:random-string}/	myPrefix/046b6c7f-0b/	您可以将它用于这两种前缀类型。 您可以将此字符串放在格式字符串的开头以获取随机前缀，若想使用 Amazon S3 获得极高的吞吐

转换	说明	示例输入	输出示例	注意
				量，此操作有时十分必要。

partitionKeyFromLambda 和 partitionKeyFromQuery 命名空间

对于动态分区，您必须在 S3 存储桶前缀中使用以下表达式格式：!{namespace:value}，其中命名空间可以是 partitionKeyFromQuery 或 partitionKeyFromLambda，也可以是两者。如果使用内联解析为源数据创建分区键，则必须指定一个 S3 存储桶前缀值，该值由以下格式指定的表达式组成："partitionKeyFromQuery:keyID"。如果使用 AWS Lambda 函数为源数据创建分区键，则必须指定一个 S3 存储桶前缀值，该值由以下格式指定的表达式组成："partitionKeyFromLambda:keyID"。有关更多信息，请参阅 Creating an Amazon Firehose stream 中的“Choose Amazon S3 for Your Destination”。

语义规则

以下规则适用于 Prefix 和 ErrorOutputPrefix 表达式。

- 对于 timestamp 命名空间，将计算未包含在单引号内的任何字符。换句话说，将按字面意思处理值字段中使用单引号进行转义的任何字符串。
- 如果指定的前缀不包含时间戳名称空间表达式，则 Firehose 会将该表达式 !{timestamp:yyyy/MM/dd/HH/} 追加到 Prefix 字段中的值。
- 序列 !{ 只能出现在 !{namespace:*value*} 表达式中。
- 仅当 Prefix 未包含任何表达式时，ErrorOutputPrefix 才能为空。在这种情况下，Prefix 的计算结果为 <specified-prefix>yyyy/MM/DDD/HH/，ErrorOutputPrefix 计算结果为 <specified-prefix><error-output-type>yyyy/MM/DDD/HH/。DDD 表示一年中的某天。
- 如果您为 ErrorOutputPrefix 指定一个表达式，则您必须包含至少一个 !{firehose:error-output-type} 实例。
- Prefix 无法包含 !{firehose:error-output-type}。
- Prefix 和 ErrorOutputPrefix 在计算后都不能超过 512 个字符。
- 如果目标位置是 Amazon Redshift，则 Prefix 不能包含表达式并且 ErrorOutputPrefix 必须为空。
- 如果目标是 Amazon S OpenSearch ervice 或 Splunk，但未ErrorOutputPrefix指定任何值，Firehose 会使用Prefix该字段来记录失败的记录。

- 当目标位置是 Amazon S3 时，Amazon S3 目标配置中的 `Prefix` 和 `ErrorOutputPrefix` 将分别用于成功记录和失败记录。如果使用 AWS CLI 或 API，则可使用 `ExtendedS3DestinationConfiguration` 指定具有自己的 `Prefix` 和 `ErrorOutputPrefix` 的 Amazon S3 备份配置。
- 当您使用 AWS 管理控制台 并将目标设置为 Amazon S3 时，Firehose 将分别使用目标配置`ErrorOutputPrefix`中的`Prefix`和来记录成功记录和失败记录。如果使用表达式指定前缀，则必须指定错误前缀，包括 `!{firehose:error-output-type}`。
- 当你`ExtendedS3DestinationConfiguration`与 AWS CLI、API 或者 CloudFormation（如果您指定了）一起使用时`S3BackupConfiguration`，Firehose 不会提供默认值。`ErrorOutputPrefix`
- 创建 ErrorOutputPrefix 表达`partitionKeyFromQuery`式时不能使用`partitionKeyFromLambda`和命名空间。

示例前缀

`Prefix` 和 `ErrorOutputPrefix` 示例

Input	计算的前缀 [2018 年 8 月 27 日上午 10:30（UTC）]

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

`Prefix`: `!{timestamp:yyyy/MM/dd}` `ErrorOutputPrefix` ：未指定	输入无效：当 Prefix 包含表达式时，`ErrorOutputPrefix` 不能为空

`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`

Input	计算的前缀 [2018 年 8 月 27 日上午 10:30（UTC）]
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 ：未指定	Prefix: myFirehosePrefix/2018/08/27/ ErrorOutputPrefix ：myFirehosePrefix/processing-failed/2018/08/27/

为 OpenSearch 服务配置索引轮换

对于 OpenSearch 服务目标，您可以从以下五个选项之一中指定基于时间的索引轮换选项：NoRotation、OneHour、OneDayOneWeek、或OneMonth。

根据您选择的轮换选项，Amazon Data Firehose 会将 UTC 到达时间戳的一部分附加到您指定的索引名称。并相应地轮换附加的时间戳。以下示例显示了 S OpenSearch ervice 中每个索引轮换选项生成的索引名称，其中指定的索引名称为myindex，到达时间戳为2016-02-25T13:00:00Z。

RotationPeriod	IndexName
NoRotation	myindex
OneHour	myindex-2016-02-25-13
OneDay	myindex-2016-02-25

RotationPeriod	IndexName
OneWeek	myindex-2016-w08
OneMonth	myindex-2016-02

> **Note**
>
> 使用 OneWeek 选项时，Data Firehose 会采用 <YEAR>-w<WEEK NUMBER> 格式（例如 2020-w33）自动创建索引，其中周数是使用 UTC 时间并根据以下美国惯例计算的：
>
> - 一周从星期日开始
> - 一年的第一周是指这一年中包含星期六的第一周

暂停和恢复数据传输

设置 Firehose 流后，流来源中的可用数据将持续传输到目的地。如果遇到流目标暂时不可用的情况（例如，在计划维护操作期间），您可能需要暂时暂停数据传输，并在目标位置再次可用时恢复。

> **Important**
>
> 当您使用下述方法暂停和恢复流时，在恢复流之后，您会看到很少有记录被传输到 Amazon S3 中的错误存储桶，而其余流继续传输到目的地。这是该方法的已知局限性，之所以发生这种情况，是因为有少量记录在多次重试后无法传输到目的地，被跟踪为失败。

暂停 Firehose 流

要暂停 Firehose 中的流传输，请先删除 Firehose 因传输失败而写入 S3 备份位置的权限。例如，如果您想暂停带有 OpenSearch 目标的 Firehose 直播，则可以通过更新权限来实现。有关更多信息，请参阅授予 Firehose 访问公共 OpenSearch 服务目标的权限。

删除 s3:PutObject 操作的 "Effect": "Allow" 权限，并显式添加一条语句，该语句对用于备份失败传输的 S3 存储桶的 s3:PutObject 操作应用 Effect": "Deny" 权限。接下来，关闭直播目标（例如，关闭目标 OpenSearch 网域），或者移除 Firehose 写入目标的权限。要更新其他目的

地的权限，请查看使用 Amazon Data Firehose 控制访问权限中的相关目的地部分。完成这两个操作后，Firehose 将停止传输直播，你可以使用 Fi rehose 的CloudWatch 指标进行监控。

> **Important**
>
> 在 Firehose 中暂停流传输时，需要确保流的来源（例如，Kinesis Data Streams 中或 Kafka 托管服务中）配置为保留数据，直到恢复流传输并将数据传输到目的地为止。如果源是 DirectPut，Firehose 将保留数据 24 小时。如果您在数据留存期到期之前未恢复流并传输数据，则可能会发生数据丢失。

恢复 Firehose 流

要恢复传输，请先打开目的地，并确保 Firehose 有权限将流传输到目的地，以还原之前对流目的地所做的更改。接下来，还原之前对应用于 S3 存储桶的权限所做的更改，以备份失败的传输。也就是说，应用 `s3:PutObject` 操作 `"Effect": "Allow"` 的权限，并删除 `s3:PutObject` 操作的 `"Effect": "Deny"` 权限，后者供 S3 存储桶用于备份失败的传输。最后，使用 Firehose 的 CloudWatch 指标进行监控，以确认直播已传送到目的地。要查看错误并对其进行故障排除，请使用适用于 Firehose 的亚马逊 CloudWatch 日志监控。

使用 Amazon Data Firehose 将数据传输到 Apache Iceberg 表

Apache Iceberg 是用于执行大数据分析的高性能开源表格格式。Apache Iceberg 为 Amazon S3 数据湖带来了 SQL 表的可靠性和简单性，并使 Spark、Flink、Trino、Hive 和 Impala 等开源分析引擎可以同时处理相同的数据。有关更多信息，请参阅 Apache Iceberg 和 注意事项和限制。

您可以使用 Firehose 将流数据传输到 Amazon S3 中的 Apache Iceberg 表。您的 Apache Iceberg 表可以在 Amazon S3 中自行管理，也可以托管在 Amazon S3 表中。在自行管理的 Iceberg 表中，您可以管理所有表优化，例如压缩和快照过期。Amazon S3 表类数据存储服务提供针对大型分析工作负载进行优化的存储，其功能旨在持续提高查询性能并降低表格数据的存储成本。有关 Amazon S3 表类数据存储服务的更多信息，请参阅 Amazon S3 表类数据存储服务。

此功能允许您将记录从单个流路由到不同的 Apache Iceberg 表中。您可以自动对这些表中的记录应用插入、更新和删除操作。它还支持对 Amazon S3 中的 Apache Iceberg 表进行精细的数据访问控制。AWS Lake Formation您可以在中集中指定访问控制， AWS Lake Formation 并为 Firehose 提供更精细的表级和列级权限。

注意事项和限制

> **Note**
>
> 除AWS 区域中国地区、亚太地区（台北）、亚太地区（马来西亚）、亚太地区（新西兰）和墨西哥（中部）外 AWS GovCloud (US) Regions，Firehose 支持将 Apache Iceberg Tables 作为目的地。

Firehose 对 Apache Iceberg 表的支持具有以下注意事项和限制。

- 吞吐量 — 如果您使用 Dire ct PUT 作为数据源向 Apache Iceberg 表传送数据，则在美国东部（弗吉尼亚北部）、美国西部（俄勒冈）和欧洲（爱尔兰）区域，每个数据流的最大吞吐量为 5 MiB/second ，其他所有区域的最大吞吐量为 1 MiB/second 。 AWS 区域如果您将数据插入到 Iceberg 表但不执行更新和删除，并且想要提高流的吞吐量，则可以使用 Firehose 限制表单请求提高吞吐量限制。

如果您只想插入数据而不想执行更新和删除，也可以将 AppendOnly 标志设置为 True。将 AppendOnly 标志设置为 True，Firehose 会自动缩放以匹配您的吞吐量。目前，您只能通过 CreateDeliveryStreamAPI 操作设置此标志。

如果由于较高的数据摄取量超过 Firehose 流的吞吐能力而导致 Direct PUT 流受到节流，则 Firehose 会自动提高流的吞吐量限制，直到节流得到控制。根据吞吐量的增加和节流，Firehose 可能需要更长的时间才能将数据流的吞吐量提高到所需的水平。因此，请继续重试失败的数据摄取记录。如果您预计数据量会突然大幅增加，或者您的新数据流需要的吞吐量比默认吞吐量限制更高，则请求提高吞吐量限制。

- 吞吐量和分区扩展 — 该服务经过优化，可以支持大量 Iceberg 分区或非常高的摄取吞吐量。随着载入吞吐量的增加，可以主动写入的分区数量会减少。

  以下是采集吞吐量的限制和支持的最大活跃分区。

摄取吞吐量	支持的最大活跃分区
≤ 20 Mb/s	最多 3,000
20—40 Mb/s	1000
40—400 Mb/s	100
400—750 Mb/s	50
750 MB/s–1.5 GB/s	1

- 每秒 S3 事务数 (TPS) — 为了优化 S3 性能，如果您使用 Kinesis Data Streams 或 Amazon MSK 作为来源，我们建议您使用正确的分区键对源记录进行分区。这样，路由到同一 Iceberg 表的数据记录就会映射到一个或几个被称为分片的源分区。如果可能，将属于不同目标 Iceberg 表的数据记录分散到不同的 Iceberg 表中。partitions/shards, so that you can use all the aggregate throughput available across all the partitions/shards of the source topic/stream
- 列：对于列名称和值，Firehose 仅采用多级嵌套 JSON 中的第一级节点。例如，Firehose 选择第一级中可用的节点，包括位置字段。源数据的列名和数据类型应与目标表的列名和数据类型完全匹配，Firehose 才能成功传输。在这种情况下，Firehose 希望您的 Iceberg 表中有结构或地图数据类型列来匹配位置字段。Firehose 支持 16 个级别的嵌套。以下是嵌套 JSON 的示例。

```
{
   "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
   }
}
```

如果列名或数据类型不匹配，则 Firehose 会抛出错误并将数据传输到 S3 错误存储桶。如果 Apache Iceberg 表中的所有列名和数据类型都匹配，但源记录中存在其他字段，则 Firehose 会跳过新字段。

- 每条记录一个 JSON 对象：您只能在一条 Firehose 记录中发送一个 JSON 对象。如果您在记录中聚合并发送多个 JSON 对象，则 Firehose 会抛出错误并将数据传输到 S3 错误存储桶。如果您使用 KPL 聚合记录，并将数据摄取到 Firehose 中，将 Amazon Kinesis Data Streams 作为源，则 Firehose 会自动解聚，并为每条记录使用一个 JSON 对象。
- 压缩和存储优化：每次使用 Firehose 写入 Iceberg 表时，它都会提交和生成快照、数据文件和删除文件。拥有许多数据文件会增加元数据开销并影响读取性能。为了获得高效的查询性能，您可能需要考虑一种解决方案，即定期获取小的数据文件并重写为较少的较大数据文件。这个过程称为压实。AWS Glue Data Catalog 支持自动压缩 Apache 冰山表。有关更多信息，请参阅《AWS Glue 用户指南》中的 Compaction management。有关更多信息，请参阅 Automatic compaction of Apache Iceberg Tables。或者，您可以运行 Athena Optimize 命令来手动执行压缩。有关“优化”命令的更多信息，请参阅 Athena Optimize。

  除了压缩数据文件外，您还可以使用对 Apache Iceberg 表执行表维护的 VACUUM 语句来减少存储消耗，例如快照过期和孤立文件移除。或者 AWS Glue Data Catalog ，您可以使用它来自动删除不再需要的数据文件、孤立文件和过期快照，从而支持 Apache Iceberg 表的托管表优化。有关更多信息，请参阅这篇关于 Storage optimization of Apache Iceberg Tables 的博客文章。
- 我们不支持 Apache Iceberg 表的 Amazon MSK 无服务器源作为目的地。
- 对于更新操作，Firehose 会先放置一个删除文件，然后再执行插入操作。放置删除文件会产生 Amazon S3 放置费用。
- Firehose 建议不使用多个 Firehose 流将数据写入同一 Apache Iceberg 表中。这是因为 Apache Iceberg 依赖于乐观并发控制 (OCC)。如果多个 Firehose 流尝试同时写入单个 Iceberg 表，则在给定时间只有一个流成功提交数据。其他无法提交回退的流，然后重试提交操作，直到配置的重试

持续时间到期。重试持续时间用完后，数据和删除文件密钥（Amazon S3 路径）将发送到配置的 Amazon S3 错误前缀。

- Firehose 支持的当前 Iceberg 库版本为 1.5.2 版。
- 要将加密数据传送到亚马逊 S3 表，您应该在 Amazon S3 表中配置 AWS Key Management Service 参数，而不是在 Firehose 配置中配置参数。如果您在 Firehose 中配置用于将加密数据传送到亚马逊 S3 表的 AWS Key Management Service 参数，则 Firehose 无法使用这些参数进行加密。有关更多信息，请参阅对AWS KMS 密钥使用服务器端加密。
- Firehose 直播仅支持向通过 Iceberg 的 API 创建的数据库和表进行传输。 GlueCatalog 不支持传输到通过 Glue SDK 创建的数据库和表。请注意，Iceberg 库中的数据库和表名不支持连字符 (-)。有关更多详细信息，请参阅 Iceberg 库支持的 Glue 数据库正则表达式和 Glue 表正则表达式。
- Firehose 写入的所有文件都是使用记录中存在的分区计算的。这也适用于删除的文件。不支持全局删除，例如为分区表写入未分区的删除文件。
- 在向 Apache Iceberg 表传送数据时，Firehose 目前不支持布隆过滤器属性。在 Iceberg 表上配置布隆过滤器属性时，Firehose 将在数据传输操作期间忽略这些属性。

使用 Apache Iceberg 表作为目的地的先决条件

从以下选项中进行选择，以完成所需的先决条件。

主题

在 Amazon S3 中传输到 Iceberg 表的先决条件

开始之前，请满足以下先决条件。

- 创建 Amazon S3 存储桶：您必须创建 Amazon S3 存储桶，以便在创建表期间添加元数据文件路径。有关更多信息，请参阅创建 S3 存储桶。
- 创建具有所需权限的 IAM 角色：Firehose 需要具有特定权限的 IAM 角色才能访问 AWS Glue 表并将数据写入 Amazon S3。相同的角色用于授予 AWS Glue 对 Amazon S3 存储桶的访问权限。在创建 Iceberg 表和 Firehose 流时，您需要此 IAM 角色。有关更多信息，请参阅 授予 Firehose 访问 Amazon S3 表的权限。
- 创建 Apache Iceberg 表：如果您在 Firehose 流中配置用于更新和删除的唯一键，则 Firehose 会在创建流时验证表和唯一键是否存在。在这种情况下，您必须在创建 Firehose 流之前创建表。您可以

使用 AWS Glue 创建 Apache 冰山表。有关更多信息，请参阅 Creating Apache Iceberg tables。如果您没有在 Firehose 流中配置唯一键，则无需在创建 Firehose 流之前创建 Iceberg 表。

> **Note**
>
> Firehose 支持 Apache Iceberg 表的以下表格版本和格式。
>
> - 表格格式版本：Firehose 仅支持 V2 表格格式。请勿创建 V1 格式的表，否则会出现错误，数据将改为传输到 S3 错误存储桶。
> - 数据存储格式：Firehose 以 Parquet 格式将数据写入 Apache Iceberg 表。
> - 行级操作 — Firehose 支持向 Apache Iceberg Tables 写入数据的 Merge-on-Read (MOR) 模式。

传输到 Amazon S3 表类数据存储服务的先决条件

要将数据传输到 Amazon S3 表类数据存储服务存储桶，请先满足以下先决条件。

- 创建 S3 表存储桶、命名空间、表存储桶中的表以及 Amazon S3 表类数据存储服务入门中概述的其他集成步骤。由于 S3 表目录集成施加的限制（如 S3 表目录集成限制中所述），列名必须为小写。
- 创建具有所需权限的 IAM 角色：Firehose 需要具有特定权限的 IAM 角色才能访问 AWS AWS Glue 表并将数据写入 Amazon S3 表类数据存储服务存储桶。要写入 Amazon S3 表存储桶中的表，您还必须为 IAM 角色提供所需的权限。Amazon S3 表目录所需的权限取决于您使用的访问控制模式：
  - IAM 访问控制 — Firehose 交付角色需要直接对 Amazon S3 表格资源的 IAM 权限。
  - Lake Formation 访问控制 — Firehose 交付角色需要 AWS AWS Lake Formation 权限才能管理对您的表格资源的访问权限。 AWS Lake Formation 使用自己的权限模型，可以对数据目录资源进行精细的访问控制。

  在创建 Firehose 流时，您配置此 IAM 角色。有关更多信息，请参阅向 Firehose 授予对 Amazon S3 表类数据存储服务的访问权限。

有关 step-by-step集成，请参阅博客使用 Amazon S3 表和 Amazon Data Firehose 构建用于流式传输数据的数据湖。有关更多信息，另请参阅在AWS 分析服务中使用 Amazon S3 表。

设置 Firehose 流

要创建以 Apache Iceberg 表作为目的地的 Firehose 流，您必须配置以下内容。

> **Note**
>
> 用于传输到 S3 表存储桶中的表的 Firehose 流的设置与 Amazon S3 中的 Apache Iceberg Tables 相同。

配置源位置和目的地

要向 Apache Iceberg 表传输数据，请为您的流选择源。

要为流配置源，请参阅配置源设置。

接下来，选择 Apache Iceberg 表作为目的地并提供 Firehose 流名称。

配置数据转换

要对数据执行自定义转换（例如，在传入流中添加或修改记录），您可以将 Lambda 函数添加到您的 Firehose 流中。有关在 Firehose 流中使用 Lambda 进行数据转换的更多信息，请参阅在 Amazon Data Firehose 中转换源数据。

对于 Apache Iceberg 表，您必须指定如何将传入记录路由到不同的目标表以及要执行的操作。向 Firehose 提供所需路由信息的其中一种方法是使用 Lambda 函数。

有关更多信息，请参阅将记录路由到不同的 Iceberg 表。

连接数据目录

Apache Iceberg 需要数据目录才能写入 Apache Iceberg 表。Firehose 与 Apache Iceber AWS Glue Data Catalog g Tables 集成。

你可以在与 Firehose 直播相同的账号 AWS Glue Data Catalog 中使用，也可以在跨账号中使用，也可以在与 Firehose 直播相同的区域（默认）中使用，也可以在不同的区域中使用。

如果您要传输到 Amazon S3 表类数据存储服务，并且正在使用控制台设置 Firehose 流，请选择与您的 Amazon S3 表类数据存储服务目录相对应的目录。如果您使用 CLI 来设置 Firehose 流，那么在 `CatalogConfiguration` 输入中，请按以下格式使用 `CatalogARN`：`arn:aws:glue:<region>:<account-id>:catalog/s3tablescatalog/<s3 table bucket name>`。有关更多信息，请参阅设置 Firehose 流到 Amazon S3 表类数据存储服务。

> **Note**
>
> Firehose 支持对 Iceberg 表执行三种操作：插入、更新和删除。如果没有指定操作，Firehose 默认为插入，将每条传入的记录添加为新行，并保留重复的记录。要改为修改现有记录，请指定“更新”操作，该操作使用主键来定位和更改现有行。
> 示例：
>
> - 默认（插入）：多个相同的客户记录会创建重复的行。
> - 指定更新：新的客户地址会更新现有记录。

配置 JQ 表达式

对于 Apache Iceberg 表，您必须指定如何将传入记录路由到不同的目标表以及要执行的插入、更新和删除等操作。为此，您可以为 Firehose 配置 JQ 表达式，以解析并获取所需信息。有关更多信息，请参阅 ???。

配置唯一键

使用多个表进行更新和删除：唯一键是源记录中的一个或多个字段，用于唯一标识 Apache Iceberg 表中的一行。如果您有包含多个表的仅插入场景，则不必配置唯一键。如果您要对某些表进行更新和删除，则必须为这些必需的表配置唯一键。请注意，如果表中缺少行，则更新将自动插入该行。如果您只有一个表，则可以配置唯一键。对于更新操作，Firehose 会先放置一个删除文件，然后再执行插入操作。

您可以在 Firehose 直播创建过程中为每个表配置唯一密钥，也可以在创建表或更改表操作期间在 Iceberg 中进行identifier-field-ids本地设置。在创建流期间为每个表配置唯一键是可选项。如果您在流创建期间没有为每个表配置唯一键，则 Firehose 会检查 `identifier-field-ids` 有无所需的表并将其用作唯一键。如果两者都未配置，则通过更新和删除操作传输数据将失败。

要配置此部分，请为要更新或删除数据的表提供数据库名称、表名称和唯一键。在配置中，每个表只能有一个条目。对于仅限附加的场景，您无需配置此部分。或者，如果表中的数据如以下示例所示无法传输，则您也可以选择提供错误存储桶前缀。

```
[
  {
    "DestinationDatabaseName": "MySampleDatabase",
    "DestinationTableName": "MySampleTable",
    "UniqueKeys": [
```

```
      "COLUMN_PLACEHOLDER"
    ],
    "S3ErrorOutputPrefix": "OPTIONAL_PREFIX_PLACEHOLDER"
  }
]
```

如果提供的列名在整个表中是唯一的，Firehose 支持配置唯一键。但是，它不支持将完全限定的列名作为唯一键。例如，如果列名 `_id` 也出现在顶层，则名为 `top._id` 的键不被视为唯一键。如果 `_id` 在整个表中是唯一的，那么无论它在表格结构中的位置如何，都会使用它。无论是顶级列还是嵌套列都是如此。在以下示例中，`_id` 是架构的有效唯一键，因为列名在架构中是唯一的。

```
[
 "schema": {
  "type": "struct",
  "fields": [
    {
      "name": "top",
      "type": {
        "type": "struct",
        "fields": [
          { "name": "_id", "type": "string" },
          { "name": "name", "type": "string" }
        ]
      }
    },
    { "name": "user", "type": "string" }
  ]
}
]
```

在以下示例中，`_id` 不是架构的有效唯一键，因为它既用于顶级列，也用于嵌套结构。

```
[
"schema": {
  "type": "struct",
  "fields": [
    {
      "name": "top",
      "type": {
        "type": "struct",
        "fields": [
          { "name": "_id", "type": "string" },
```

```
          { "name": "name", "type": "string" }
        ]
      }
    },
    { "name": "_id", "type": "string" }
  ]
}
]
```

指定重试持续时间

您可以使用此配置来指定 Firehose 在写入 Amazon S3 中的 Apache Iceberg 表中遇到失败时应尝试重试的持续时间（以秒为单位）。您可以设置 0 到 7200 秒之间的任何值，以执行重试。默认情况下，Firehose 的重试时间为 300 秒。

处理失败的传输或处理

您必须将 Firehose 配置为将记录传输到 S3 备份存储桶，以防它在重试持续时间到期后遇到处理或传输数据流失败问题。为此，请在控制台的备份设置中配置 S3 备份存储桶和 S3 备份存储桶错误输出前缀。

处理错误

Firehose 将所有传送错误发送到 CloudWatch 日志和 Amazon S3 错误存储桶。

错误列表：

错误消息	描述
`Iceberg.NoSuchTable`	Firehose 正在写入一个不存在的表，或者该表不是 V2 格式。Firehose 不支持 V1 格式的表。
`Iceberg.InvalidTableName`	传递的表名为 null 或为空，或者表的格式不是 V2。Firehose 不支持 V1 格式的表。
`S3.AccessDenied`	确保在先决条件步骤中创建的 IAM 角色具有所需的权限和信任策略。
`Glue.AccessDenied`	确保在先决条件步骤中创建的 IAM 角色具有所需的权限和信任策略。

配置缓冲区提示

Firehose 将内存中传入的流数据缓冲到一定大小（缓冲大小），或缓冲一定时间（缓冲时间间隔）后再将其传输到 Apache Iceberg 表。您可以选择 1—128 的缓冲区大小 MiBs 和 0—900 秒的缓冲间隔。缓冲区提示越高，S3 写入次数越少，压缩成本越低（由于越大的数据文件），查询运行时速度更快，但延迟更高。较低的缓冲区提示值以较低的延迟传输数据。

配置高级设置

您可以为 Apache Iceberg 表配置服务器端加密、错误日志记录、权限和标签。有关更多信息，请参阅配置高级设置。您必须添加作为 ??? 一部分而创建的 IAM 角色。Firehose 将担任访问 AWS Glue 表和写入 Amazon S3 存储桶的角色。

可能需要几分钟的时间才能完成 Firehose 流创建。成功创建 Firehose 流后，您可以开始向其中摄取数据，并可以查看 Apache Iceberg 表中的数据。

将传入记录路由到单个 Iceberg 表

如果您要 Firehose 将数据插入到单个 Iceberg 表，则只需在流配置中配置单个数据库和表，如以下示例 JSON 所示。对于单个表，您不需要 JQ 表达式和 Lambda 函数来向 Firehose 提供路由信息。如果您将这些字段与 JQ 或 Lambda 一起提供，则 Firehose 将从 JQ 或 Lambda 获取输入。

```
[
  {
    "DestinationDatabaseName": "UserEvents",
    "DestinationTableName": "customer_id",
    "UniqueKeys": [
      "COLUMN_PLACEHOLDER"
    ],
    "S3ErrorOutputPrefix": "OPTIONAL_PREFIX_PLACEHOLDER"
  }
]
```

在此示例中，Firehose 将所有输入记录路由到 `UserEvents` 数据库中的 `customer_id` 表。如果要对单个表执行更新或删除操作，则必须使用方法或 Lambda JSONQuery方法向 Firehose 提供每条传入记录的操作。

将传入记录路由到不同的 Iceberg 表

Amazon Data Firehose 可以根据记录的内容将流中的传入记录路由到不同的 Iceberg 表。从 Amazon Data Firehose 传输记录时，记录不按顺序保存。请参阅以下示例输入记录。

```
{
  "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"
}
```

在此示例中，**deviceId** 字段有两个可能的值 – Device1234 和 Device4567。当传入记录的 **deviceId** 字段为 Device1234 时，我们要将记录写入名为 Device1234 的 Iceberg 表，而当传入记录的 **deviceId** 字段为 Device4567，我们想要将记录写入名为 Device4567 的表中。

请注意，带有 Device1234 和 Device4567 的记录可能有一组不同的字段，这些字段映射到相应的 Iceberg 表中的不同列。传入的记录可能具有嵌套的 JSON 结构，其中 **deviceId** 可以嵌套在 JSON 记录中。在接下来的几节中，我们将讨论在这种情况下如何通过向 Firehose 提供适当的路由信息将记录路由到不同的表。

使用表达式向 Firehose 提供路由信息 JSONQuery

向 Firehose 提供记录路由信息的最简单、最具成本效益的方法是 JSONQuery 提供表达式。使用这种方法，您可以为三个参数提供 JSONQuery 表达式 — `Database Name` `Table Name`、和（可选）`Operation`。Firehose 使用您提供的表达式从传入的流记录中提取信息以路由记录。

`Database Name` 参数指定目标数据库的名称。`Table Name` 参数指定目标表的名称。`Operation` 是一个可选参数，用于指示是将传入的流记录作为新记录插入目标表，还是修改或删除目标表中的现有记录。“操作”字段必须具有以下值之一 – `insert`、`update` 或 `delete`。

对于这三个参数中的每一个，您可以提供静态值或动态表达式，在其中从传入的记录中检索值。例如，如果要将所有传入的流记录传输到名为 `IoTevents` 的单个数据库，则数据库名称的静态值将为“`IoTevents`”。如果必须从传入记录中的字段中获取目标表名，则表名是一个动态表达式，用于指定传入记录中需要从中检索目标表名的字段。

在以下示例中，我们为数据库名称使用静态值，为表名使用动态值，为操作使用静态值。请注意，指定操作是可选项。如果未指定任何操作，则 Firehose 会默认将传入的记录作为新记录插入到目标表中。

```
Database Name : "IoTevents"
Table Name : .deviceId
Operation : "insert"
```

如果 `deviceId` 字段嵌套在 JSON 记录中，则我们将具有嵌套字段信息的表名指定为 `.event.deviceId`。

> **Note**
>
> - 当您将操作指定为`update`或时`delete`，您必须在设置 Firehose 流时为目标表指定唯一键，或者在 Iceberg 中运行创建表或更改表操作时identifier-field-ids在 Iceberg 中设置唯一的键。如果您未指定此项，则 Firehose 会抛出错误并将数据传输到 S3 错误存储桶。
> - `Database Name` 和 `Table Name` 值必须与目标数据库和表名完全匹配。如果它们不匹配，则 Firehose 会抛出错误并将数据传输到 S3 错误存储桶。

使用 AWS Lambda 函数提供路由信息

在某些情况下，您可能有复杂的规则来决定如何将传入记录路由到目标表。例如，您可能有一个规则来定义字段是否包含值 A、B 或 F，该规则应路由到名为 `TableX` 的目标表，或者您可能希望通过添加其他属性来增加传入的流记录。例如，如果一条记录包含的字段 `device_id` 为 1，则可能需要添加

另一个 device_type 为“modem”的字段，并将该附加字段写入目标表列。在这种情况下，您可以使用 Firehose 中的 AWS Lambda 函数转换源流，并在 Lambda 转换函数的输出中提供路由信息。要了解如何使用 Firehose 中的 AWS Lambda 函数转换源流，请参阅在 Amazon Data Firehose 中转换源数据。

当您使用 Lambda 在 Firehose 中转换源流时，输出必须包含 recordId、result 和 data 或 KafkaRecordValue 参数。参数 recordId 包含输入流记录，result 指示转换是否成功，data 包含 Lambda 函数的 Base64 编码的转换输出。有关更多信息，请参阅 ???。

```
{
  "recordId": "49655962066601463032522589543535113056108699331451682818000000",
  "result": "Ok",
  "data": "1IiwiI6ICJmYWxsIiwgImdgU21IiwiI6ICJmYWxsIiwg==tcHV0ZXIgU2NpZW5jZSIsICJzZW1"
}
```

要向 Firehose 指定如何将流记录路由到目标表的路由信息，作为 Lambda 函数的一部分，Lambda 函数的输出必须包含 metadata 的附加部分。以下示例显示了如何将元数据部分添加到 Firehose 流的 Lambda 输出中，该流使用 Kinesis Data Streams 作为数据来源，指示 Firehose 必须将该记录作为新记录插入到数据库 IoTevents 的名为 Device1234 的表中。

```
 {
"recordId": "49655962066601463032522589543535113056108699331451682818000000",
    "result": "Ok",
    "data":
 "1IiwiI6ICJmYWxsIiwgImdgU21IiwiI6ICJmYWxsIiwg==tcHV0ZXIgU2NpZW5jZSIsICJzZW1",

    "metadata":{
"otfMetadata":{
            "destinationTableName":"Device1234",
            "destinationDatabaseName":"IoTevents",
            "operation":"insert"
        }
    }
  }
```

同样，以下示例显示了如何将元数据部分添加到 Firehose 的 Lambda 输出中，它使用 Amazon Managed Streaming for Apache Kafka 作为数据来源，指示 Firehose 必须将该记录作为新记录插入到数据库 IoTevents 中名为 Device1234 的表中。

```
{
```

```
"recordId": "49655962066601463032522589543535113056108699331451682818000000",
    "result": "Ok",
    "kafkaRecordValue":
 "1IiwiI6ICJmYWxsIiwgImdgU21IiwiI6ICJmYWxsIiwg==tcHV0ZXIgU2NpZW5jZSIsICJzZW1",

    "metadata":{
"otfMetadata":{
            "destinationTableName":"Device1234",
            "destinationDatabaseName":"IoTevents",
            "operation":"insert"
        }
    }
  }
```

对于这个示例，

- `destinationDatabaseName` 指的是目标数据库的名称，是必填字段。
- `destinationTableName` 指的是目标表的名称，是必填字段。
- `operation` 是一个可选字段，可能的值为 `insert`、`update` 和 `delete`。如果您未指定任何值，则默认操作为 `insert`。

> **Note**
>
> - 当您将操作指定为`update`或时`delete`，您必须在设置 Firehose 流时为目标表指定唯一键，或者在 Iceberg 中运行创建表或更改表操作时identifier-field-ids在 Iceberg 中设置唯一的键。如果您未指定此项，则 Firehose 会抛出错误并将数据传输到 S3 错误存储桶。
> - `Database Name` 和 `Table Name` 值必须与目标数据库和表名完全匹配。如果它们不匹配，则 Firehose 会抛出错误并将数据传输到 S3 错误存储桶。
> - 当您的 Firehose 流同时具有 Lambda 转换函数和表达式时 JSONQuery ，Firehose 会首先检查 Lambda 输出中的元数据字段，以确定如何将记录路由到相应的目标表，然后查看表达式的输出中是否有缺失的字段。 JSONQuery
>
>   如果 Lambda 或 JSONQuery 表达式未提供所需的路由信息，则 Firehose 会将其视为单表场景，并在唯一密钥配置中查找单表信息。
>
>   有关更多信息，请参阅将传入记录路由到单个 Iceberg 表。如果 Firehose 无法确定路由信息并将记录与指定的目标表进行匹配，则会将数据传输到您指定的 S3 错误存储桶。

示例 Lambda 函数

此 Lambda 函数是示例 Python 代码，用于解析传入的流记录并添加必填字段，以指定应如何将数据写入特定表。您可以使用此示例代码添加用于路由信息的元数据部分。

```
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
```

监控指标。

为了向 Apache Iceberg Tables 传输数据，Firehose 会在直播级别发布以下 CloudWatch 指标。

指标	说明
DeliveryToIceberg.Bytes	在指定时段内传输到 Apache Iceberg 表的字节数。 单位：字节
DeliveryToIceberg.IncomingRowCount	Firehose 尝试向 Apache Iceberg 表传输的记录数量。 单位：计数
DeliveryToIceberg.SuccessfulRowCount	成功传输到 Apache Iceberg 表的行数。 单位：计数
DeliveryToIceberg.FailedRowCount	传输到 S3 备份存储桶失败的行数。 单位：计数
DeliveryToIceberg.DataFreshness	Firehose 中最早记录的期限（从进入 Firehose 到现在）。任何大于此期限的记录已传输到 Apache Iceberg 表。 单位：秒
DeliveryToIceberg.Success	成功提交到 Apache Iceberg 表的总和。
JQProcessing.Duration	运行 JQ 表达式所花的时间。 单位：毫秒

了解支持的数据类型

Firehose 支持 Apache Iceberg 所支持的所有原始和复杂数据类型。有关更多信息，请参阅 Schemas and Data Types。以字符串形式发送二进制数据时，必须使用 Firehose 支持的编码类型：基本 Base64、MIME Base64、URL 和文件名安全 Base64 以及 Hex。对于 Timestamp 数据类型，必须始终以微秒为单位发送。

数据类型示例

下一节介绍不同数据类型的示例。

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 默认、base64 mime、base64 url-safe、十六进制 ）


```
{
    "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 （以微秒为单位的时代， LocalTime Java 对象）

```
{
    "destination_column_0": 68175096000,
    "destination_column_1": "18:56:15.096"
}
```

TimestampType.withZone（以微秒为单位的纪元、Java 对象、 OffsetDateTime Java 对象） 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（以微秒为单位的纪元、Java 对象、 LocalDateTime Java 对象、J OffsetDateTime ava 对象） 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（UUID Java 对象）

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

资源

使用以下资源了解更多信息：

- 使用 Amazon Data Firehose 将实时数据流式传输到 Amazon S3 中的 Apache Iceberg 表中
- 使用 Apache Iceber AWS WAF g 和 Amazon Data Firehose 简化日志分析
- 使用 Amazon S3 表类数据存储服务和 Amazon Data Firehose 构建用于流式传输数据的数据湖

标记 Firehose 流

您可以将自己的元数据以标签的形式分配给您在 Amazon Data Firehose 中创建的 Firehose 流。标签是您为流定义的键值对。使用标签是一种管理 AWS 资源和整理数据（包括账单数据）的简单而强大的方法。

您可以在调用创建新的 Fire CreateDeliveryStreamhose 直播时指定标签。对于现有的 Firehose 流，您可以使用以下三个操作添加、列出和移除标签：

- TagDeliveryStream
- ListTagsForDeliveryStream
- UntagDeliveryStream

了解标签基本知识

您可以使用 Amazon Data Firehose API 操作完成以下任务：

- 向 Firehose 流添加标签。
- 列出 Firehose 流的标签。
- 从 Firehose 流中移除标签。

您可以使用标签对 Firehose 流进行分类。例如，您可以按用途、所有者或环境对 Firehose 流进行分类。由于您定义每个标签的键和值，因此您可以创建一组自定义类别来满足您的特定需求。例如，您可以定义一组标签来帮助您按拥有者和关联应用程序跟踪 Firehose 流。

以下是标签的多个示例：

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

如果您在 `CreateDeliveryStream` 操作中执行了标签，则 Amazon Data Firehose 会对 `firehose:TagDeliveryStream` 操作执行额外的授权，以验证用户是否具备创建标签的权

限。如果您不提供此权限，则创建带有 IAM 资源标签的新 Firehose 流的请求将失败，并显示 `AccessDeniedException`，如下所示。

```
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.
```

以下示例演示了允许用户创建 Firehose 流并应用标签的策略。

通过标记跟踪成本

您可以使用标签对 AWS 费用进行分类和跟踪。当您将标签应用于 AWS 资源（包括 Firehose 流）时，您的 AWS 成本分配报告将包括按标签汇总的使用量和成本。您可通过应用代表业务类别（如成本中心、应用程序名称或拥有者）的标签来整理多种服务的成本。有关更多信息，请参阅《AWS Billing 用户指南》中的对自定义账单报告使用成本分配标签。

了解标签限制

以下限制适用于 Amazon Data Firehose 中的标签。

基本限制

- 每个资源（流）的最大标签数是 50。
- 标签键和值区分大小写。
- 无法更改或编辑已删除的流的标签。

标签键限制

- 每个标签键必须是唯一的。如果您添加的标签具有已使用的键，则您的新标签将覆盖现有键值对。
- 标签键不能以 `aws:` 开头，因为此前缀将预留以供 AWS使用。 AWS 将代表您创建以此前缀开头的标签，但您不能编辑或删除这些标签。
- 标签键的长度必须介于 1 和 128 个 Unicode 字符之间。
- 标签键必须包含以下字符：Unicode 字母、数字、空格和以下特殊字符：`_ . / = + - @`。

标签值限制

- 标签值的长度必须介于 0 和 255 个 Unicode 字符之间。
- 标签值可以为空。另外，它们必须包含以下字符：Unicode 字母、数字、空格和以下任意特殊字符：_ . / = + - @。

Amazon Data Firehose 中的安全性

云安全 AWS 是重中之重。作为 AWS 客户，您将受益于专为满足大多数安全敏感型组织的要求而构建的数据中心和网络架构。

安全是双方共同承担 AWS 的责任。责任共担模式将其描述为云的 安全性和云中 的安全性：

- 云安全 — AWS 负责保护在 AWS 云中运行 AWS 服务的基础架构。 AWS 还为您提供可以安全使用的服务。作为 AWS 合规性计划的一部分，我们的安全措施的有效性定期由第三方审计员进行测试和验证。要了解适用于 Data Firehose 的合规计划，请参阅按合规性计划提供的范围内AWS 服务。
- 云端安全-您的责任由您使用的 AWS 服务决定。您还需要对其他因素负责，包括您的数据的敏感性、您组织的要求以及适用的法律法规。

本文档可帮助您了解如何在使用 Data Firehose 时应用责任共担模式。以下主题展示了如何配置 Data Firehose 以实现您的安全性和合规性目标。您还将学习如何使用其他 AWS 服务来帮助您监控和保护您的 Data Firehose 资源。

主题

Amazon Data Firehose 中的数据保护

Amazon Data Firehose 使用 TLS 协议对所有传输中数据进行加密。此外，对于处理期间存储在临时存储中的数据，Amazon Data Firehose 使用 AWS Key Management Service 来加密数据，并使用校验和验证来验证数据完整性。

如果有敏感数据，您可以在使用 Amazon Data Firehose 时启用服务器端数据加密。执行此操作的方式取决于您的数据源。

> **Note**
>
> 如果您在 AWS 通过命令行界面或 API 进行访问时需要经过 FIPS 140-2 验证的加密模块，请使用 FIPS 端点。有关可用的 FIPS 端点的更多信息，请参阅《美国联邦信息处理标准（FIPS）第 140-2 版》。

Server-side 使用 Kinesis Data Streams 进行加密

当您将数据从数据生成器发送到数据流时，Kinesis Data Streams 会在存储静态数据之前使用 AWS Key Management Service AWS KMS() 密钥对数据进行加密。当 Firehose 流从数据流中读取数据时，Kinesis Data Streams 会先解密数据，然后将其发送到 Amazon Data Firehose。Amazon Data Firehose 根据您指定的缓冲提示在内存中缓冲数据。然后将其传输到目标，而不以静态方式存储未加密数据。

有关如何为 Kinesis Data Streams 启用服务器端加密的信息，请参阅亚马逊 Kinesis Data Streams 开发者指南中的 Server-Side 使用加密。

Server-side 使用直接 PUT 或其他数据源进行加密

如果您使用或向 Firehose 流发送数据 PutRecordBatch，PutRecord或者使用 AWS IoT Amazon L CloudWatch ogs 或 CloudWatch Events 发送数据，则可以使用该操作开启服务器端加密。StartDeliveryStreamEncryption

要停止服务器端加密，请使用操作。StopDeliveryStreamEncryption

您还可以在创建 Firehose 流时启用 SSE。为此，请指定DeliveryStreamEncryptionConfigurationInput何时调用CreateDeliveryStream。

要成功使用CUSTOMER_MANAGED_CMK，调用方的 IAM 策略和 KMS 密钥策略都必须允许`kms:GenerateDataKey`和`kms:Decrypt`操作。Firehose 会在您调用 PutRecord 或 PutRecordBatch 加密时验证这些权限。CUSTOMER_MANAGED_CMK此外，调 StartDeliveryStreamEncryption 用 CreateDeliveryStream 或CUSTOMER_MANAGED_CMK加密时需要`kms:CreateGrant`获得许可。

当 CMK 类型为 CUSTOMER_MANAGED_CMK，时，如果 Amazon Data Firehose 服务因 KMSNotFoundException、KMSInvalidStateException、KMSDisabledException 或 KMSAccessDeniedException 而无法解密记录，服务最多会等待 24 小时（保留期），以便您解决问题。如果保留期过后，问题仍然存在，服务会跳过已过保留期但无法解密的记录，然后丢弃数

据。Amazon Data Firehose 提供了以下四个 CloudWatch 指标，您可以使用这些指标来跟踪这四个 AWS KMS 异常：

- `KMSKeyAccessDenied`
- `KMSKeyDisabled`
- `KMSKeyInvalidState`
- `KMSKeyNotFound`

有关这 4 个指标的更多信息，请参阅the section called “使用 CloudWatch 指标进行监控”。

> **Important**
>
> 要加密您的 Firehose 流，请使用对称 CMK。Amazon Data Firehose 不支持非对称 CMK。有关对称和非对称 CMK 的信息，请参阅开发者指南中的关于对称和非对称 CM K。 AWS Key Management Service

> **Note**
>
> 当您使用客户托管密钥（CUSTOMER_MANAGED_CMK）为 Firehose 流启用服务器端加密（SSE）时，Firehose 服务会在使用您的密钥时设置加密上下文。由于此加密上下文表示使用您 AWS 账户拥有的密钥的情况，因此它会作为您 AWS 账户 AWS CloudTrail 事件日志的一部分进行记录。此加密上下文是由 Firehose 服务生成的系统。您的应用程序不应对 Firehose 服务设置的加密上下文的格式或内容做出任何假设。

使用 Amazon Data Firehose 控制访问权限

以下几节介绍了如何控制对 Amazon Data Firehose 资源的访问以及来自这些资源的访问。涵盖的信息包括如何授予您的应用程序访问这些资源的权限，以便可以向您的 Firehose 流发送数据。它们还描述了如何授予 Amazon Data Firehose 访问您的亚马逊简单存储服务 (Amazon S3) 存储桶、亚马逊 Redshift 集群或亚马逊 OpenSearch 服务集群的权限，以及使用 Datadog、Dynatrace、 LogicMonitor MongoDB、New Relic、Splunk 或 Sumo Logic 作为目的地时所需的访问权限。最后，您将在本主题中找到有关如何配置 Amazon Data Firehose 的指南，以便可以将数据传输到属于不同 AWS 账户的目的地。管理所有这些形式的访问的技术是 AWS Identity and Access Management (IAM)。有关 IAM 的更多信息，请参阅什么是 IAM？。

内容

授予对您的 Firehose 资源的访问权限

要授予您的应用程序访问 Firehose 流的权限，请使用类似下例的策略。您可以通过修改 `Action` 部分对要授予权限的各个 API 操作进行调整，或通过 `"firehose:*"` 向所有操作授予权限。

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"
            ]
        }
    ]
}
```

授予 Firehose 对私有 Amazon MSK 集群的访问权限

如果 Firehose 流的源是私有 Amazon MSK 集群，则使用类似本例的策略。

您必须在集群的基于资源的策略中添加这样的策略，以向 Firehose 服务主体授予调用 Amazon MSK `CreateVpcConnection` API 操作的权限。

允许 Firehose 担任 IAM 角色

本节介绍了授予 Amazon Data Firehose 访问权限，以摄取、处理数据并将数据从源传输到目的地的权限和策略。

> **Note**
>
> 如果您使用控制台创建 Firehose 直播并选择创建新角色的选项，则会将所需的信任策略 AWS 附加到该角色。如果您希望 Amazon Data Firehose 使用现有 IAM 角色或者您自己创建角色，请将以下信任策略附加到该角色，以便 Amazon Data Firehose 可以承担该角色。编辑政策以

替换*account-id*为您的 AWS 账户 ID。有关如何修改角色的信任关系的更多信息，请参阅修改角色。

Amazon Data Firehose 使用 IAM 角色来获取 Firehose 流处理和传输数据所需的权限。确保将以下信任策略附加到该角色，以便 Amazon Data Firehose 可以承担该角色。

如果您选择 Amazon MSK 作为 Firehose 流的源，则必须指定另一个 IAM 角色，该角色授予 Amazon Data Firehose 从指定的 Amazon MSK 集群摄取源数据的权限。确保将以下信任策略附加到该角色，以便 Amazon Data Firehose 可以承担该角色。

JSON

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

该角色授予 Amazon Data Firehose 从指定的 Amazon MSK 集群摄取源数据的权限，要确保该角色授予以下权限：

授予 Firehose 访问权限 AWS Glue 用于数据格式转换

如果您的 Firehose 流执行数据格式转换，Amazon Data Firehose 会引用存储在 AWS Glue中的表定义。要向 Amazon Data Firehose 提供必要的访问权限 AWS Glue，请在您的政策中添加以下声明。有关如何查找表格的 ARN 的信息，请参阅指定 Glue 资源 AR AWS N。

```
{
```

```
    "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: ['*'],
}
```

从架构注册表获取架构的建议策略没有资源限制。有关更多信息，请参阅开发人员指南中的反序列化器的 IAM 示例。 AWS Glue

授予 Firehose 访问 Amazon S3 目的地的权限

当您使用 Amazon S3 目标时，Amazon Data Firehose 会将数据传输到您的 S3 存储桶，并且可以选择使用您拥有的 AWS KMS 密钥进行数据加密。如果启用了错误记录，Amazon Data Firehose 还会将数据传输错误发送到您的 CloudWatch 日志组和流。创建 Firehose 流时，您需要拥有 IAM 角色。Amazon Data Firehose 担任该 IAM 角色并获得对指定存储桶、密钥、 CloudWatch 日志组和流的访问权限。

使用以下访问策略，以便 Amazon Data Firehose 能够访问您的 S3 存储桶和 AWS KMS 密钥。如果您没有 S3 存储桶，请将 `s3:PutObjectAcl` 添加到 Amazon S3 操作列表中。这将授予存储桶所有者对 Amazon Data Firehose 传输的对象的完全访问权限。

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"
            ]
        }
    ]
}
```

上面的策略还有一个允许访问 Amazon Kinesis Data Streams 的语句。如果您不使用 Kinesis Data Streams 作为数据来源，可以删除该语句。如果您使用 Amazon MSK 作为源，则可以用以下内容代替该语句：

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

有关允许其他 AWS 服务访问您的 AWS 资源的更多信息，请参阅 IAM 用户指南中的创建角色以向 AWS 服务委派权限。

要了解如何授予 Amazon Data Firehose 对其他账户中的 Amazon S3 目的地的访问权限，请参阅the section called “Cross-account 配送到亚马逊 S3 目的地”。

授予 Firehose 访问 Amazon S3 表的权限

Firehose 需要一个具有特定权限的 IAM 角色才能访问 AWS AWS Glue 表并将数据写入 Amazon S3 表存储桶中的表。要写入 Amazon S3 表存储桶中的表，您还必须为 IAM 角色提供所需的权限。Amazon S3 表目录所需的权限取决于您使用的访问控制模式：

- IAM 访问控制 — Firehose 交付角色需要直接对 Amazon S3 表格资源的 IAM 权限。
- Lake Formation 访问控制 — Firehose 交付角色需要 AWS AWS Lake Formation 权限才能管理对您的表格资源的访问权限。 AWS Lake Formation 使用自己的权限模型，可以对数据目录资源进行精细的访问控制。

在创建 Firehose 流时，您配置此 IAM 角色。选择与您的访问控制模式相对应的选项卡。

IAM 访问控制

如果您使用的是 IAM 访问控制模式（不使用 AWS Lake Formation），则 Firehose 交付角色需要直接对 Amazon S3 表格资源和 AWS Glue 数据目录对象的 IAM 权限。

登录 AWS 管理控制台 并打开 IAM 控制台，网址为https://console.aws.amazon.com/iam/。

创建策略并在策略编辑器中选择 JSON。添加以下授予所需权限的内联策略。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3TablesAccessPermission",
            "Effect": "Allow",
            "Action": [
                "s3tables:GetTable",
                "s3tables:GetTableData",
                "s3tables:GetTableMetadataLocation",
                "s3tables:UpdateTableMetadataLocation"
            ],
            "Resource": [
                "arn:aws:s3tables:region:account-id:bucket/*",
                "arn:aws:s3tables:region:account-id:bucket/*/table/*"
            ]
        },
        {
            "Sid": "S3TableBucketAccessPermission",
            "Effect": "Allow",
            "Action": [
                "s3tables:GetTableBucket"
            ],
            "Resource": "arn:aws:s3tables:region:account-id:bucket/*"
        },
        {
            "Sid": "GlueCatalogAccessPermission",
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabase",
                "glue:GetDatabases",
                "glue:GetTable",
                "glue:GetTables",
                "glue:UpdateTable"
            ],
```

```
            "Resource": [
                "arn:aws:glue:region:account-id:catalog",
                "arn:aws:glue:region:account-id:catalog/s3tablescatalog",
                "arn:aws:glue:region:account-id:catalog/s3tablescatalog/*",
                "arn:aws:glue:region:account-id:database/*",
                "arn:aws:glue:region:account-id:table/*/*"
            ]
        },
        {
            "Sid": "S3DeliveryErrorBucketPermission",
            "Effect": "Allow",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::error-delivery-bucket",
                "arn:aws:s3:::error-delivery-bucket/*"
            ]
        },
        {
            "Sid": "RequiredWhenUsingKinesisDataStreamsAsSource",
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStream",
                "kinesis:GetShardIterator",
                "kinesis:GetRecords",
                "kinesis:ListShards"
            ],
            "Resource": "arn:aws:kinesis:region:account-id:stream/stream-name"
        },
        {
            "Sid": "KMSPermissionForS3TablesEncryption",
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:GenerateDataKey"
            ],
            "Resource": [
                "arn:aws:kms:region:account-id:key/key-id"
```

```
            ],
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "s3.amazonaws.com"
                },
                "StringLike": {
                    "kms:EncryptionContext:aws:s3:arn":
 "arn:aws:s3tables:region:account-id:bucket/*/table/*"
                }
            }
        },
        {
            "Sid": "RequiredWhenUsingLambdaForDataTransformation",
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction",
                "lambda:GetFunctionConfiguration"
            ],
            "Resource": "arn:aws:lambda:region:account-id:function:function-
name:function-version"
        },
        {
            "Sid": "CloudWatchLogsPermission",
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:region:account-id:log-group:log-group-name:log-
stream:log-stream-name"
        }
    ]
}
```

该策略包含允许访问亚马逊 Kinesis Data Streams、调用 Lambda 函数和访问密钥的声明。 AWS KMS 如果您不使用这些资源中的任何一个，则可以删除相应的语句。如果启用了错误记录，Amazon Data Firehose 还会将数据传输错误发送到您的 CloudWatch 日志组和流。您必须配置日志组和日志流名称才能使用此选项。对于日志组和日志流名称，请参阅使用日志监控亚马逊数据 Firehose CloudWatch。

在内联策略中，将占位符值替换为您的实际资源名称、 AWS 账户 数字和区域。

创建策略后，在上打开 IAM 控制台，https://console.aws.amazon.com/iam/并创建一个 IAM 角色AWS 服务作为可信实体类型。

对于服务或使用案例，选择 Kinesis。对于使用案例，选择 Kinesis Firehose。

在下一页上，选择上一步中创建的策略附加到该角色。在查看页面上，您会发现信任策略已附加到此角色，以授予 Firehose 服务承担此角色的权限。在您创建角色时，Amazon Data Firehose 可以假设该角色对 AWS Glue 和 Amazon S3 表执行所需的操作。将 Firehose 服务主体添加到所创建角色的信任策略中。有关更多信息，请参阅 允许 Firehose 担任 IAM 角色。

Lake Formation 访问控制

如果您使用的是 AWS Lake Formation 访问控制模式，则除了 IAM 策略外，Firehose 交付角色还需要凭证售卖 AWS Lake Formation 权限。

登录 AWS 管理控制台 并打开 IAM 控制台，网址为https://console.aws.amazon.com/iam/。

创建策略并在策略编辑器中选择 JSON。添加以下内联策略，授予 Amazon S3 read/write 权限，例如权限、更新数据目录中表的权限等。

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>"
        ]
      }
    ]
}
```

该策略包含允许访问亚马逊 Kinesis Data Streams、调用 Lambda 函数和访问密钥的声明。 AWS KMS 如果您不使用这些资源中的任何一个，则可以删除相应的语句。如果启用了错误记录，Amazon Data Firehose 还会将数据传输错误发送到您的 CloudWatch 日志组和流。您必须配置日志组和日志流名称才能使用此选项。对于日志组和日志流名称，请参阅使用日志监控亚马逊数据 Firehose CloudWatch。

在内联策略中，<error delivery bucket>用您的 Amazon S3 存储桶名称替换，aws-account-id将区域替换为资源的有效 AWS 账户 编号和区域。

除了 IAM 策略外，您还必须向中的 Firehose 交付角色授予所需的权限。 AWS Lake Formation有关更多信息，请参阅授予表权限。

创建策略后，在上打开 IAM 控制台，https://console.aws.amazon.com/iam/并创建一个 IAM 角色AWS服务作为可信实体类型。

对于服务或使用案例，选择 Kinesis。对于使用案例，选择 Kinesis Firehose。

在下一页上，选择上一步中创建的策略附加到该角色。在查看页面上，您会发现信任策略已附加到此角色，以授予 Firehose 服务承担此角色的权限。在您创建角色时，Amazon Data Firehose 可以假设该角色对 AWS Glue 和 S3 存储桶执行所需的操作。将 Firehose 服务主体添加到所创建角色的信任策略中。有关更多信息，请参阅 允许 Firehose 担任 IAM 角色。

向 Firehose 授予 Apache Iceberg 表目的地的访问权限

在使用 AWS Glue创建 Firehose 流和 Apache Iceberg 表之前，您必须具有 IAM 角色。使用以下步骤创建策略和 IAM 角色。Firehose 承担此 IAM 角色并执行所需的操作。

1. 登录 AWS 管理控制台 并打开 IAM 控制台，网址为https://console.aws.amazon.com/iam/。
2. 创建策略并在策略编辑器中选择 JSON。
3. 添加以下内联策略，授予 Amazon S3 read/write 权限，例如权限、更新数据目录中表的权限等。

   此策略有一项语句，该语句允许访问 Amazon Kinesis Data Streams、调用 Lambda 函数和访问 KMS 密钥。如果您不使用这些资源中的任何一个，则可以删除相应的语句。

   如果启用了错误记录，Firehose 还会向您的 CloudWatch 日志组和直播发送数据传输错误。为此，您必须配置日志组和日志流名称。对于日志组和日志流名称，请参阅使用日志监控亚马逊数据 Firehose CloudWatch。
4. 在内联策略中，*amzn-s3-demo-bucket*用您的 Amazon S3 存储桶名称、aws-account-id 和区域替换为有效的资源 AWS 账户 编号和区域。

   > **Note**
   >
   > 此角色授予对数据目录中的所有数据库和表的权限。如果需要，则您可以仅向特定的表和数据库授予权限。

5. 创建策略后，打开 IAM 控制台并创建一个 IAM 角色，将 AWS 服务 作为可信实体类型。
6. 对于服务或使用案例，选择 Kinesis。对于使用案例，选择 Kinesis Firehose。

7. 在下一页上，选择上一步中创建的策略附加到该角色。在查看页面上，您会发现信任策略已附加到此角色，以授予 Firehose 服务承担此角色的权限。在您创建角色时，Amazon Data Firehose 可以承担该角色以对 AWS Glue 和 S3 存储桶执行所需的操作。

授予 Firehose 对 Amazon Redshift 目的地的访问权限

在使用 Amazon Redshift 目的地时授予对 Amazon Data Firehose 的访问权限时，请参阅以下内容。

主题

IAM 角色和访问策略

在使用 Amazon Redshift 目的地时，Amazon Data Firehose 会将数据传输到 S3 存储桶作为中间位置。它可以选择使用您拥有的 AWS KMS 密钥进行数据加密。然后，Amazon Data Firehose 将数据从 S3 存储桶加载到您的 Amazon Redshift 预置集群或 Amazon Redshift Serverless 工作组。如果启用了错误记录，Amazon Data Firehose 还会将数据传输错误发送到您的 CloudWatch 日志组和流。Amazon Data Firehose 使用指定的 Amazon Redshift 用户名和密码访问您的预配置集群或 Amazon Redshift 无服务器工作组，并使用 IAM 角色访问指定的存储桶、密钥、日志组和流。CloudWatch 创建 Firehose 流时，您需要拥有 IAM 角色。

使用以下访问策略，以便 Amazon Data Firehose 能够访问您的 S3 存储桶和 AWS KMS 密钥。如果您没有 S3 存储桶，请将 `s3:PutObjectAcl` 添加到 Amazon S3 操作列表，这将授予存储桶所有者对 Amazon Data Firehose 传输的对象的完全访问权限。此策略还有一个允许访问 Amazon Kinesis Data Streams 的语句。如果您不使用 Kinesis Data Streams 作为数据来源，可以删除该语句。

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"
            ]
        }
    ]
}
```

有关允许其他 AWS 服务访问您的 AWS 资源的更多信息，请参阅 IAM 用户指南中的创建角色以向 AWS 服务委派权限。

对 Amazon Redshift 预置集群或 Amazon Redshift Serverless 工作组的 VPC 访问

如果您的 Amazon Redshift 预置集群或 Amazon Redshift Serverless 工作组位于虚拟私有云（VPC）中，则必须可以使用公有 IP 地址对其进行公开访问。此外，通过取消阻止 Amazon Data Firehose IP 地址，授予 Amazon Data Firehose 对 Amazon Redshift 预置集群或 Amazon Redshift Serverless 工作组的访问权限。目前，Amazon Data Firehose 为每个可用区域使用一个 CIDR 块。

Region	CIDR 块
美国东部（俄亥俄州）	13.58.135.96/27
美国东部（弗吉尼亚州北部）	52.70.63.192/27
美国西部（北加利福尼亚）	13.57.135.192/27

Region	CIDR 块
美国西部（俄勒冈州）	52.89.255.224/27
AWS GovCloud (US-East)	18.253.138.96/27
AWS GovCloud (US-West)	52.61.204.160/27
加拿大（中部）	35.183.92.128/27
加拿大西部（卡尔加里）	40.176.98.192/27
亚太地区（香港）	18.162.221.32/27
Asia Pacific (Mumbai)	13.232.67.32/27
亚太地区（海得拉巴）	18.60.192.128/27
亚太地区（首尔）	13.209.1.64/27
亚太地区（新加坡）	13.228.64.192/27
亚太地区（悉尼）	13.210.67.224/27
亚太地区（雅加达）	108.136.221.64/27
亚太地区（东京）	13.113.196.224/27
亚太地区（大阪）	13.208.177.192/27
亚太地区（泰国）	43.208.112.96/27
亚太地区（台北）	43.212.53.160/27
中国（北京）	52.81.151.32/27
中国（宁夏）	161.189.23.64/27
欧洲（苏黎世）	16.62.183.32/27

Region	CIDR 块
欧洲地区（法兰克福）	35.158.127.160/27
欧洲地区（爱尔兰）	52.19.239.192/27
欧洲地区（伦敦）	18.130.1.96/27
欧洲地区（巴黎）	35.180.1.96/27
欧洲地区（斯德哥尔摩）	13.53.63.224/27
欧洲（西班牙）	18.100.71.96/27
中东（巴林）	15.185.91.0/27
墨西哥（中部）	78.12.207.32/27
南美洲（圣保罗）	18.228.1.128/27
欧洲地区（米兰）	15.161.135.128/27
非洲（开普敦）	13.244.121.224/27
中东（阿联酋）	3.28.159.32/27
以色列（特拉维夫）	51.16.102.0/27
亚太地区（墨尔本）	16.50.161.128/27
亚太地区（马来西亚）	43.216.58.0/27

有关如何取消阻止 IP 地址的更多信息，请参阅《Amazon Redshift 入门指南》中授予访问集群的权限的步骤。

授予 Firehose 访问公共 OpenSearch 服务目标的权限

当您使用 OpenSearch 服务目标时，Amazon Data Firehose 会将数据传输到您的 OpenSearch 服务集群，并同时将失败的文档或所有文档备份到您的 S3 存储桶。如果启用了错误记录，Amazon Data Firehose 还会将数据传输错误发送到您的 CloudWatch日志组和流。Amazon Data Firehose 使用 IAM

角色访问指定的 OpenSearch服务域、S3 存储桶、 AWS KMS 密钥以及 CloudWatch 日志组和流。创建 Firehose 流时，您需要拥有 IAM 角色。

使用以下访问策略允许 Amazon Data Firehose 访问您的 S3 存储桶、 OpenSearch服务域和 AWS KMS 密钥。如果您没有 S3 存储桶，请将 `s3:PutObjectAcl` 添加到 Amazon S3 操作列表，这将授予存储桶所有者对 Amazon Data Firehose 传输的对象的完全访问权限。此策略还有一个允许访问 Amazon Kinesis Data Streams 的语句。如果您不使用 Kinesis Data Streams 作为数据来源，可以删除该语句。

有关允许其他 AWS 服务访问您的 AWS 资源的更多信息，请参阅 IAM 用户指南中的创建角色以向 AWS 服务委派权限。

要了解如何授予 Amazon Data Firehose 访问其他账户中 OpenSearch 服务集群的权限，请参阅。the section called “Cross-account 配送到 OpenSearch 服务目的地”

向 Firehose 授予对 VPC 中 OpenSearch 服务目标的访问权限

如果您的 OpenSearch 服务域位于 VPC 中，请务必向 Amazon Data Firehose 授予上一节中描述的权限。此外，您需要向 Amazon Data Firehose 授予以下权限，使其能够访问您的 OpenSearch 服务域的 VPC。

- `ec2:DescribeVpcs`
- `ec2:DescribeVpcAttribute`
- `ec2:DescribeSubnets`
- `ec2:DescribeSecurityGroups`
- `ec2:DescribeNetworkInterfaces`
- `ec2:CreateNetworkInterface`
- `ec2:CreateNetworkInterfacePermission`
- `ec2:DeleteNetworkInterface`

> **Important**
>
> 创建 Firehose 流后，请勿撤销这些权限。如果您撤消这些权限，则每当服务尝试查询或更新 ENI 时，您的 Firehose 流就会降级或停止向 OpenSearch 您的服务域传送数据。

> **Important**
>
> 在私有 VPC 中指定将数据传输到目的地的子网时，请确保所选子网中有足够数量的免费 IP 地址。如果指定子网中没有可用的免费 IP 地址，则 Firehose 将无法为私有 VPC 中的数据传输创建或添加 ENI，并且传输将降级或失败。

创建或更新您的 Firehose 直播时，您可以指定一个安全组，让 Firehose 在向您的服务域发送数据时使用。 OpenSearch 您可以使用 OpenSearch 服务域使用的相同安全组，也可以使用不同的安全组。如果您指定其他安全组，请确保它允许 OpenSearch服务域安全组的出站 HTTPS 流量。此外，请确保 OpenSearch 服务域的安全组允许来自您在配置 Firehose 直播时指定的安全组的 HTTPS 流量。如果您对 Firehose 直播和 OpenSearch 服务域使用相同的安全组，请确保安全组入站规则允许 HTTPS 流量。有关安全组规则的更多信息，请参阅 Amazon VPC 文档中的安全组规则。

授予 Firehose 访问公共 OpenSearch 无服务器目标的权限

当您使用 OpenSearch 无服务器目标时，Amazon Data Firehose 会将数据传输到 OpenSearch 您的无服务器集合，并同时将失败的文档或所有文档备份到您的 S3 存储桶。如果启用了错误记录，Amazon Data Firehose 还会将数据传输错误发送到您的 CloudWatch 日志组和流。Amazon Data Firehose 使用 IAM 角色访问指定的 OpenSearch 无服务器集合、S3 存储桶、 AWS KMS 密钥、 CloudWatch 日志组和流。创建 Firehose 流时，您需要拥有 IAM 角色。

使用以下访问策略允许 Amazon Data Firehose 访问您的 S3 存储桶、 OpenSearch无服务器域和密钥。 AWS KMS 如果您没有 S3 存储桶，请将 `s3:PutObjectAcl` 添加到 Amazon S3 操作列表，这将授予存储桶所有者对 Amazon Data Firehose 传输的对象的完全访问权限。此策略还有一个允许访问 Amazon Kinesis Data Streams 的语句。如果您不使用 Kinesis Data Streams 作为数据来源，可以删除该语句。

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"
       }
     ]
}
```

除上述策略外，您还必须配置 Amazon Data Firehose，以便在数据访问策略中分配以下最低权限：

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

有关允许其他 AWS 服务访问您的 AWS 资源的更多信息，请参阅 IAM 用户指南中的创建角色以向 AWS 服务委派权限。

向 Firehose 授予对 OpenSearch VPC 中无服务器目标的访问权限

如果您的 OpenSearch 无服务器集合位于 VPC 中，请务必向 Amazon Data Firehose 授予上一节中描述的权限。此外，您需要向 Amazon Data Firehose 授予以下权限，使其能够访问您的 OpenSearch 无服务器集合的 VPC。

- `ec2:DescribeVpcs`
- `ec2:DescribeVpcAttribute`
- `ec2:DescribeSubnets`
- `ec2:DescribeSecurityGroups`
- `ec2:DescribeNetworkInterfaces`
- `ec2:CreateNetworkInterface`
- `ec2:CreateNetworkInterfacePermission`
- `ec2:DeleteNetworkInterface`

> **Important**
>
> 创建 Firehose 流后，请勿撤销这些权限。如果您撤消这些权限，则每当服务尝试查询或更新 ENI 时，您的 Firehose 流就会降级或停止向 OpenSearch 您的服务域传送数据。

> **Important**
>
> 在私有 VPC 中指定将数据传输到目的地的子网时，请确保所选子网中有足够数量的免费 IP 地址。如果指定子网中没有可用的免费 IP 地址，则 Firehose 将无法为私有 VPC 中的数据传输创建或添加 ENI，并且传输将降级或失败。

创建或更新您的 Firehose 直播时，您可以指定一个安全组，让 Firehose 在向您的无服务器集合发送数据时使用。 OpenSearch 您可以使用与 OpenSearch Serverless 集合相同的安全组，也可以使用不同的安全组。如果您指定其他安全组，请确保它允许 OpenSearch 无服务器集合的安全组的出站 HTTPS 流量。此外，请确保 OpenSearch无服务器集合的安全组允许来自您在配置 Firehose 直播时指定的安全组的 HTTPS 流量。如果您对 Firehose 直播和 OpenSearch Serverless 集合使用相同的安全组，请确保安全组入站规则允许 HTTPS 流量。有关安全组规则的更多信息，请参阅 Amazon VPC 文档中的安全组规则。

授予 Firehose 访问 Splunk 目的地的权限

在使用 Splunk 目的地时，Amazon Data Firehose 会将数据传输到 Splunk HTTP 事件收集器（HEC）端点。它还会将该数据备份到您指定的 Amazon S3 存储桶，您也可以选择使用自己拥有的 AWS KMS 密钥进行 Amazon S3 服务器端加密。如果启用了错误记录，Firehose 会将数据传输错误发送到您的 CloudWatch日志流。您也可以 AWS Lambda 用于数据转换。

如果您使用 AWS 负载均衡器，请确保它是 Classic 负载均衡器或 Application 负载均衡器。此外，使用为经典负载均衡器禁用的 Cookie 有效期启用基于持续时间的粘接会话，应用程序负载均衡器的有效期设置为最大值（7 天）。有关如何执行此操作的信息，请参阅 C lassic Load Balancer 或 Application Load Bal ancer 的 Duration-Based 会话粘性。

创建 Firehose 流时，您必须有 IAM 角色。Firehose 担任该 IAM 角色并获得对指定存储桶、密钥、CloudWatch 日志组和流的访问权限。

使用以下访问策略，以便 Amazon Data Firehose 能够访问您的 S3 存储桶。如果您没有 S3 存储桶，请将 `s3:PutObjectAcl` 添加到 Amazon S3 操作列表，这将授予存储桶所有者对 Amazon Data Firehose 传输的对象的完全访问权限。该政策还授予 Amazon Data Firehose 访问权限以 CloudWatch 进行错误记录和数据转 AWS Lambda 换。此策略还有一个允许访问 Amazon Kinesis Data Streams 的语句。如果您不使用 Kinesis Data Streams 作为数据来源，可以删除该语句。Amazon Data Firehose 不使用 IAM 访问 Splunk。要访问 Splunk，它使用您的 HEC 令牌。

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"
            ]
        }
    ]
}
```

有关允许其他 AWS 服务访问您的 AWS 资源的更多信息，请参阅 IAM 用户指南中的创建角色以向 AWS 服务委派权限。

在 VPC 中访问 Splunk

如果 Splunk 平台位于 VPC 中，它必须具有公有 IP 地址以公开进行访问。此外，还要取消阻止 Amazon Data Firehose IP 地址，以便为 Amazon Data Firehose 授予对 Splunk 平台的访问权限。Amazon Data Firehose 目前使用以下 CIDR 块。

Region	CIDR 块
美国东部（俄亥俄州）	18.216.68.160/27, 18.216.170.64/27, 18.216.170.96/27 \
美国东部（弗吉尼亚州北部）	34.238.188.128/26, 34.238.188.192/26, 34.238.195.0/26
美国西部（北加利福尼亚）	13.57.180.0/26

Region	CIDR 块
美国西部（俄勒冈州）	34.216.24.32/27, 34.216.24.192/27, 34.216.24.224/27
AWS GovCloud (US-East)	18.253.138.192/26
AWS GovCloud (US-West)	52.61.204.192/26
亚太地区（香港）	18.162.221.64/26
亚太地区（台北）	43.212.53.192/26
亚太地区（孟买）	13.232.67.64/26
亚太地区（首尔）	13.209.71.0/26
亚太地区（新加坡）	13.229.187.128/26
亚太地区（悉尼）	13.211.12.0/26
亚太地区（泰国）	43.208.112.128/26
亚太地区（东京）	13.230.21.0/27, 13.230.21.32/27
加拿大（中部）	35.183.92.64/26
加拿大西部（卡尔加里）	40.176.98.128/26
欧洲地区（法兰克福）	18.194.95.192/27, 18.194.95.224/27, 18.195.48.0/27
欧洲地区（爱尔兰）	34.241.197.32/27, 34.241.197.64/27, 34.241.197.96/27
欧洲地区（伦敦）	18.130.91.0/26
Europe (Paris)	35.180.112.0/26

Region	CIDR 块
欧洲（西班牙）	18.100.194.0/26
欧洲地区（斯德哥尔摩）	13.53.191.0/26
中东（巴林）	15.185.91.64/26
墨西哥（中部）	78.12.207.64/26
南美洲（圣保罗）	18.228.1.192/26
欧洲地区（米兰）	15.161.135.192/26
非洲（开普敦）	13.244.165.128/26
亚太地区（大阪）	13.208.217.0/26
中国（北京）	52.81.151.64/26
中国（宁夏）	161.189.23.128/26
亚太地区（雅加达）	108.136.221.128/26
中东（阿联酋）	3.28.159.64/26
以色列（特拉维夫）	51.16.102.64/26
欧洲（苏黎世）	16.62.183.64/26
亚太地区（海得拉巴）	18.60.192.192/26
亚太地区（墨尔本）	16.50.161.192/26
亚太地区（马来西亚）	43.216.44.192/26
亚太地区（新西兰）	3.102.119.128/26

使用 Amazon Data Firehose 将 VPC 流日志摄取到 Splunk

要详细了解如何创建 VPC 流日志订阅、发布到 Firehose 以及将 VPC 流日志发送到支持的目的地，请参阅使用 Amazon Data Firehose 将 VPC 流日志摄取到 Splunk 中。

访问 Snowflake 或 HTTP 端点

当目的地是 HTTP 端点或 Snowflake 公有集群时，没有特定于 Amazon Data Firehose 的 AWS IP 地址范围子集。

要将 Firehose 添加到公有 Snowflake 集群的允许列表或您的公有 HTTP 或 HTTPS 端点，请将所有当前 AWS IP 地址范围添加到您的入口规则。

> **Note**
>
> 通知并不总是来自与其关联主题相同 AWS 区域的 IP 地址。您必须包括所有区域 AWS 的 IP 地址范围。

授予 Firehose 访问 Snowflake 目的地的权限

当您使用 Snowflake 作为目的地时，Firehose 会使用您的 Snowflake 账户 URL 将数据传输到 Snowflake 账户。它还会将错误数据备份到您指定的亚马逊简单存储服务存储桶，您也可以选择使用自己拥有的 AWS Key Management Service 密钥进行 Amazon S3 服务器端加密。如果启用了错误记录，Firehose 会将数据传输错误发送到您的 CloudWatch 日志流。

创建 Firehose 流之前，您必须拥有 IAM 角色。Firehose 担任该 IAM 角色并获得对指定存储桶、密钥、 CloudWatch 日志组和流的访问权限。为使 Firehose 能够访问您的 S3 存储桶，请使用以下访问策略。如果您没有 S3 存储桶，请将 `s3:PutObjectAcl` 添加到 Amazon Simple Storage Service 操作列表，这将授予存储桶所有者对 Firehose 传输的对象的完全访问权限。此政策还授予 Firehose 访问错误记录 CloudWatch 的权限。此策略还有一个允许访问 Amazon Kinesis Data Streams 的语句。如果您不使用 Kinesis Data Streams 作为数据来源，可以删除该语句。Firehose 未使用 IAM 访问 Snowflake。要访问 Snowflake，它会使用你的 Snowflake 账户 Url 和 PrivateLink Vpce ID（如果是私有集群）。

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:*"
            ]
        }
    ]
}
```

有关允许其他 AWS 服务访问您的 AWS 资源的更多信息，请参阅 IAM 用户指南中的创建角色以向 AWS 服务委派权限。

访问 VPC 中的 Snowflake

如果您的 Snowflake 集群启用了私有链接，则 Firehose 将在创建私有链接时使用以下 VPC 端点之一将数据传输到您的私有集群，而无需通过公共互联网。为此，请创建 Snowflake 网络规则，以允许 AWS 区域 您的集群所在`AwsVpceIds`的以下内容进入。有关更多信息，请参阅 Snowflake User Guide 中的 Creating network rule。

根据您的集群所在区域使用的 VPC 端点 ID

AWS 区域	**VPCE IDs**
美国东部（俄亥俄州）	vpce-0d96cafcd96a50aeb vpce-0cec34343d48f537b
美国东部（弗吉尼亚州北部）	vpce-0b4d7e8478e141ba8 vpce-0b75cd681fb507352 vpce-01c03e63820ec00d8 vpce-0c2cfc51dc2882422

AWS 区域	VPCE IDs
	vpce-06ca862f019e4e056
	vpce-020cda0cfa63f8d1c
	vpce-0b80504a1a783cd70
	vpce-0289b9ff0b5259a96
	vpce-0d7add8628bd69a12
	vpce-02bfb5966cc59b2af
	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

AWS 区域	VPCE IDs
	vpce-066b7e13cbfca6f6e vpce-0674541252d9ccc26 vpce-03540b88dedb4b000 vpce-0b1828e79ad394b95 vpce-0dc0e6f001fb1a60d vpce-0d8f82e71a244098a vpce-00e374d9e3f1af5ce vpce-0c1e3d6631ddb442f
美国西部（俄勒冈州）	vpce-0f60f72da4cd1e4e7 vpce-0c60d21eb8b1669fd vpce-01c4e3e29afdafbef vpce-0cc6bf2a88da139de vpce-0797e08e169e50662 vpce-033cbe480381b5c0e vpce-00debbdd8f9eb10a5 vpce-08ec2f386c809e889 vpce-0856d14310857b545
欧洲地区（法兰克福）	vpce-068dbb7d71c9460fb vpce-0a7a7f095942d4ec9

AWS 区域	VPCE IDs
欧洲地区（爱尔兰）	vpce-06857e59c005a6276 vpce-04390f4f8778b75f2 vpce-011fd2b1f0aa172fd
亚太地区（东京）	vpce-06369e5258144e68a vpce-0f2363cdb8926fbe8
亚太地区（新加坡）	vpce-049cd46cce7a12d52 vpce-0e8965a1a4bdb8941
亚太地区（首尔）	vpce-0aa444d9001e1faa1 vpce-04a49d4dcfd02b884
亚太地区（悉尼）	vpce-048a60a182c52be63 vpce-03c19949787fd1859
亚太地区（孟买）	vpce-0d68cb822f6f0db68 vpce-0517d32692ffcbde2
欧洲地区（伦敦）	vpce-0fd1874a0ba3b9374 vpce-08091b1a85e206029
南美洲（圣保罗）	vpce-065169b8144e4f12e vpce-0493699f0e5762d63
加拿大（中部）	vpce-07e6ed81689d5271f vpce-0f53239730541394c

AWS 区域	VPCE IDs
欧洲地区（巴黎）	vpce-09419680077e6488a vpce-0ea81ba2c08140c14
亚太地区（大阪）	vpce-0a9f003e6a7e38c05 vpce-02886510b897b1c5a
欧洲地区（斯德哥尔摩）	vpce-0d96410833219025a vpce-060a32f9a75ba969f
亚太地区（雅加达）	vpce-00add4b9a25e5c649 vpce-004ae2de34338a856

授予 Firehose 对 HTTP 端点目的地的访问权限

您可以使用 Amazon Data Firehose 将数据传输到任何 HTTP 端点目的地。Amazon Data Firehose 还会将该数据备份到您指定的 Amazon S3 存储桶，您可以选择使用您拥有的 AWS KMS 密钥进行 Amazon S3 服务器端加密。如果启用了错误记录，Amazon Data Firehose 会将数据传输错误发送到您的 CloudWatch 日志流。您也可以 AWS Lambda 用于数据转换。

创建 Firehose 流时，您需要拥有 IAM 角色。Amazon Data Firehose 担任该 IAM 角色并获得对指定存储桶、密钥、 CloudWatch 日志组和流的访问权限。

使用以下访问策略，以便 Amazon Data Firehose 能够访问您指定用于数据备份的 S3 存储桶。如果您没有 S3 存储桶，请将 `s3:PutObjectAcl` 添加到 Amazon S3 操作列表，这将授予存储桶所有者对 Amazon Data Firehose 传输的对象的完全访问权限。该政策还授予 Amazon Data Firehose 访问权限以 CloudWatch 进行错误记录和数据转 AWS Lambda 换。此策略还有一个允许访问 Amazon Kinesis Data Streams 的语句。如果您不使用 Kinesis Data Streams 作为数据来源，可以删除该语句。

> **Important**
>
> Amazon Data Firehose 不使用 IAM 访问受支持的第三方服务提供商拥有的 HTTP 终端节点目标，包括 Datadog、Dynatrace、MongoDB、New Relic LogicMonitor、Splunk 或 Sumo

Logic。要访问受支持的第三方服务提供商拥有的指定 HTTP 端点目标，请联系该服务提供商，获取从 Amazon Data Firehose 向该服务传输数据所需的 API 密钥或访问密钥。

有关允许其他 AWS 服务访问您的 AWS 资源的更多信息，请参阅 IAM 用户指南中的创建角色以向 AWS 服务委派权限。

Important

目前，Amazon Data Firehose 不支持向 VPC 中的 HTTP 端点传输数据。

Cross-account 从亚马逊 MSK 发货

当您从 Firehose 账户（例如账户 B）创建 Firehose 直播并且您的来源是另一个账户（ AWS 账户 A）中的 MSK 集群时，您必须进行以下配置。

账户 A：

1. 在 Amazon MSK 控制台中，选择预置的集群，然后选择属性。
2. 在“网络设置”下，选择“编辑”，然后打开“Multi-VPC 连接”。
3. 在安全设置下，选择编辑集群策略。
   a. 如果集群尚未配置策略，请选中包含 Firehose 服务主体和启用 Firehose 跨账户 S3 传输。AWS 管理控制台 将自动生成具有相应权限的策略。
   b. 如果集群已配置了策略，请向现有策略添加以下权限：

```
{
      "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. 在 AWS 主体下，输入账户 B 的主体 ID。
5. 在主题下，指定您需要 Firehose 流从中摄取数据的 Apache Kafka 主题。创建 Firehose 流后，您将无法更新此主题。
6. 选择保存更改

账户 B：

1. 在 Firehose 控制台中，选择使用账户 B 创建 Firehose 流。
2. 在源下，选择 Amazon Managed Streaming for Apache Kafka。
3. 在源设置下，对于 Amazon Managed Streaming for Apache Kafka 集群，请在账户 A 中输入 Amazon MSK 集群的 ARN。

4. 在主题下，指定您需要 Firehose 流从中摄取数据的 Apache Kafka 主题。创建 Firehose 流后，您将无法更新此主题。
5. 在传输流名称中，指定 Firehose 流的名称。

在账户 B 中创建 Firehose 直播时，您必须拥有一个 IAM 角色（使用时默认创建 AWS 管理控制台），该角色授予 Firehose 流对已配置主题的跨账户 Amazon MSK 集群的 “读取” 权限。

以下是由 AWS 管理控制台配置的内容：

```
{
    "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
    },
 }
```

接下来，您可以完成配置记录转换和记录格式转换的可选步骤。有关更多信息，请参阅 [（可选）配置记录转换和格式转换](#)。

Cross-account 配送到亚马逊 S3 目的地

您可以使用 AWS CLI 或 Amazon Data Firehose API 在一个账户中创建 Firehose 直播，在另一个 AWS 账户中创建 Amazon S3 目标。以下过程展示了一个示例，该示例配置账户 A 拥有的 Firehose 流，以向账户 B 拥有的 Amazon S3 存储桶传输数据。

1. 使用[授予 Firehose 对 Amazon S3 目的地的访问权限](#)中所述的步骤，以账户 A 身份创建 IAM 角色。

   > **Note**
   >
   > 在本例中，访问策略中指定的 Amazon S3 存储桶由账户 B 拥有。务必将 `s3:PutObjectAcl` 添加到访问策略中的 Amazon S3 操作列表，该访问策略为账户 B 授予对 Amazon Data Firehose 传输的对象的完全访问权限。跨账户传输需要此权限。Amazon Data Firehose 将请求的“x-amz-acl”标头设置为“bucket-owner-full-control”。

2. 要允许从之前创建的 IAM 角色进行访问，请以账户 B 身份创建 S3 存储桶策略。以下代码是存储桶策略的示例。有关更多信息，请参阅[使用存储桶策略和用户策略](#)。

   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. 使用您在步骤 1 中创建的 IAM 角色，以账户 A 身份创建 Firehose 流。

Cross-account 配送到 OpenSearch 服务目的地

您可以使用 AWS CLI 或 Amazon Data Firehose API 在一个 AWS 账户中创建 Firehose 直播，并在另一个账户中创建 OpenSearch 服务目标。以下过程显示了一个示例，说明如何在账户 A 下创建 Firehose 流并将其配置为向账户 B 拥有的 OpenSearch 服务目标传送数据。

1. 使用 the section called “授予 Firehose 访问公共 OpenSearch 服务目标的权限” 中所述的步骤以账户 A 身份创建 IAM 角色。
2. 要允许从您在上一步中创建的 IAM 角色进行访问，请在账户 B 下创建 OpenSearch 服务策略。以下 JSON 为例。

   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. 使用您在步骤 1 中创建的 IAM 角色，以账户 A 身份创建 Firehose 流。创建 Firehose 流时，请使用 AWS CLI 或亚马逊 Data Firehose API，并指定该`ClusterEndpoint`字段而不是 “服务” 字段。`DomainARN` OpenSearch

> **Note**
>
> 要在一个 AWS 账户中创建 Firehose 直播，而 OpenSearch 服务目标位于另一个账户中，则必须使用 AWS CLI 或 Amazon Data Firehose API。您不能使用 AWS 管理控制台 来创建此类跨账户配置。

使用标签控制访问

您可以使用 IAM 策略中的可选 `Condition` 元素（或 `Condition` 块），根据标签键和值微调对 Amazon Data Firehose 操作的访问。以下小节介绍了如何针对不同的 Amazon Data Firehose 操作执

行上述操作。有关使用 Condition 元素以及您可在其内使用的运算符的更多信息，请参阅 IAM JSON 策略元素：条件。

CreateDeliveryStream

对于 CreateDeliveryStream 操作，请使用 aws:RequestTag 条件键。在以下示例中，MyKey 和 MyValue 表示标签的键和对应的值。有关更多信息，请参阅 了解标签基本知识。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "firehose:CreateDeliveryStream",
            "firehose:TagDeliveryStream"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/MyKey": "MyValue"
            }
        }
    }]
}
```

TagDeliveryStream

对于 TagDeliveryStream 操作，请使用 aws:TagKeys 条件键。在以下示例中，MyKey 为示例标签键。

JSON

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

```
            "Action": "firehose:TagDeliveryStream",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "MyKey"
                }
            }
        }
    ]
}
```

UntagDeliveryStream

对于 `UntagDeliveryStream` 操作，请使用 `aws:TagKeys` 条件键。在以下示例中，`MyKey` 为示例标签键。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "firehose:UntagDeliveryStream",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "MyKey"
                }
            }
        }
    ]
}
```

ListDeliveryStreams

您不能将基于标签的访问控制用于 `ListDeliveryStreams`。

其他操作

对于 CreateDeliveryStream、TagDeliveryStream、UntagDeliveryStream 和 ListDeliveryStreams之外的所有 Firehose 操作，请使用 aws:RequestTag 条件键。在以下示例中，MyKey 和 MyValue 表示标签的键和对应的值。

ListDeliveryStreams，使用 firehose:ResourceTag 条件键根据该 Firehose 流上的标签来控制访问。

在以下示例中，MyKey 和 MyValue 表示标签的键和对应的值。该策略仅适用于具有名为 MyKey 的标签，值为 MyValue 的 Data Firehose 流。有关基于资源标签控制访问权限的更多信息，请参阅 IAM 用户指南中的使用标签控制对 AWS 资源的访问权限。

使用进行身份验证 AWS Secrets Manager 在亚马逊 Data Firehose 中

Amazon Data Firehose 与 AWS Secrets Manager 之集成，可让您安全地访问您的密钥并自动轮换证书。通过这种集成，Firehose 可在运行时从 Secrets Manager 检索密钥，以连接到前面提到的流式传输目的地并传输您的数据流。这样，无论是在还是 API 参数中，在直播创建工作流程中，您的密钥都不会以 AWS 管理控制台 纯文本形式显示。它提供了一种安全的实践方法来管理您的密钥，并使您免于复杂的凭证管理活动（例如，设置自定义 Lambda 函数来管理密码轮换）。

有关更多信息，请参阅 AWS Secrets Manager 《用户指南》。

主题

- 了解密钥
- 创建密钥
- 使用密钥
- 轮换密钥

了解密钥

秘密可以是密码、一组凭证（如用户名和密码）、OAuth 令牌或以加密形式存储在 Secrets Manager 中的其他秘密信息。

对于每个目标，您必须以正确的 JSON 格式指定密钥键值对，如下一节所示。如果您的密钥没有按照目的地的正确 JSON 格式，则 Amazon Data Firehose 将无法连接到您的目的地。

MySQL 和 PostgreSQL 等数据库的密钥格式

```
{
    "username":  "<username>",
    "password":  "<password>"
}
```

Amazon Redshift 预置集群和 Amazon Redshift Serverless 工作组的密钥格式

```
{
    "username":  "<username>",
    "password":  "<password>"
}
```

Splunk 的密钥格式

```
{
    "hec_token":  "<hec token>"
}
```

Snowflake 的密钥格式

```
{
    "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
}
```

HTTP 端点、Coralogix、Datadog、Dynatrace、Elastic、Honeycomb、、、MongoDB Cloud 和 New Relic 的秘密格式 LogicMonitor Logz.io

```
{
    "api_key":  "<apikey>"
}
```

创建密钥

要创建密钥，请按照《AWS Secrets Manager 用户指南》中创建 AWS Secrets Manager 密钥中的步骤进行操作。

使用密钥

我们建议你使用存储凭证或密钥 AWS Secrets Manager 来连接直播目的地，例如亚马逊 Redshift、HTTP 终端节
点、Snowflake、Splunk、Coralogix、Datadog、Dynatrace、Elastic、Honeycomb、、、MongoDB Cloud 和 New Relic。 LogicMonitor Logz.io

在创建 Firehose 流时，您可以通过 AWS 管理控制台使用 Secrets Manager 为这些目的地配置身份验证。有关更多信息，请参阅 配置目的地设置。或者，您也可以使用CreateDeliveryStream和UpdateDestinationAPI 操作来配置 Secrets Manager 的身份验证。

Firehose 使用加密方法缓存密钥，并在每次连接到目的地时使用这些密钥。它每 10 分钟刷新一次缓存，以确保可使用最新的凭证。

在流的生命周期中，您可以选择随时关闭从 Secrets Manager 检索密钥的功能。如果您不想使用 Secrets Manager 来检索密钥，则可以改用 username/password 或 API 密钥。

> **Note**
>
> 尽管在 Firehose 中的此功能无需支付额外费用，但您需要为访问和维护 Secrets Manager 付费。有关更多信息，请参阅 AWS Secrets Manager 定价页。

授予 Firehose 检索密钥的访问权限

要让 Firehose 从中检索密钥 AWS Secrets Manager，您必须向 Firehose 提供访问密钥所需的权限以及加密您的密钥的密钥。

使用 AWS Secrets Manager 存储和检索密钥时，有几种不同的配置选项，具体取决于密钥的存储位置和加密方式。

- 如果密钥与您的 IAM 角色存储在同一个 AWS 账户中，并且使用默认 AWS 托管密钥 (`aws/secretsmanager`) 进行加密，则 Firehose 担任的 IAM 角色只需要`secretsmanager:GetSecretValue`获得该密钥的权限即可。

```
// secret role policy
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "Secret ARN"
        }
    ]
}
```

有关 IAM 策略的更多信息，请参阅 AWS Secrets Manager的权限策略示例。

- 如果密钥与角色存储在同一个账户中，但使用客户自主管理型密钥（CMK）加密，则该角色同时需要 `secretsmanager:GetSecretValue` 和 `kms:Decrypt` 权限。CMK 策略还需要允许 IAM 角色执行 `kms:Decrypt`。
- 如果密钥存储在与您的角色不同的 AWS 账户中，并且使用默认 AWS 托管密钥进行加密，则无法进行此配置，因为当使用托管密钥加密密钥时，Secrets Manager 不允许跨账户访问。 AWS
- 如果密钥存储在不同的账户中并使用 CMK 加密，则 IAM 角色需要对该密钥的 `secretsmanager:GetSecretValue` 权限和对 CMK 的 `kms:Decrypt` 权限。密钥的资源策略和其他账户中的 CMK 策略还需要允许 IAM 角色具有必要的权限。有关更多信息，请参阅Cross-account 访问权限。

轮换密钥

轮换是指定期更新密钥。您可以配置 AWS Secrets Manager 为按照您指定的时间表自动轮换密钥。这样，您可以将长期密钥替换为短期密钥。这有助于降低泄露的风险。有关更多信息，请参阅《AWS Secrets Manager 用户指南》中的轮换 AWS Secrets Manager 密钥。

通过 Amazon Data Firehose 控制台管理 IAM 角色

Amazon Data Firehose 是一项完全托管的服务，用于实时将流数据传递到目的地。您还可以配置 Firehose，以便在传输之前转换数据格式。要使用这些功能，您必须先提供 IAM 角色，以便在创建或编辑 Firehose 流时向 Firehose 授予权限。Firehose 使用此 IAM 角色来获取 Firehose 流所需的所有权限。

例如，假设您创建一个向 Amazon S3 传输数据的 Firehose 流，而此 Firehose 流在启用功能的情况下启用了 “转换源记录”。 AWS Lambda 在这种情况下，您必须提供 IAM 角色，以向 Firehose 授予访问 S3 存储桶和调用 Lambda 函数的权限，如下所示。

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>/
*"]
    }]
}
```

您可以通过 Firehose 控制台选择如何提供这些角色。您可以从以下选项之一进行选择。

- 选择现有 IAM 角色
- 从控制台中创建新的 IAM 角色

选择现有 IAM 角色

您可以从现有的 IAM 角色中进行选择。使用此选项，请确保您选择的 IAM 角色具有适当的信任策略以及源和目的地所需的权限。有关更多信息，请参阅 使用 Amazon Data Firehose 控制访问权限。

从控制台中创建新的 IAM 角色

或者，您也可以使用 Firehose 控制台来代表您创建新的角色。

当 Firehose 代表您创建 IAM 角色时，该角色会自动包含基于 Firehose 流配置授予所需权限的所有权限和信任策略。

例如，如果您没有启用使用 AWS Lambda转换源记录功能，则控制台会在权限策略中生成以下语句。

```
{
  "Sid": "lambdaProcessing",
  "Effect": "Allow",
```

```
   "Action": [
     "lambda:InvokeFunction",
     "lambda:GetFunctionConfiguration"
   ],
   "Resource": "arn:aws:lambda:us-east-1123456789012:function:
%FIREHOSE_POLICY_TEMPLATE_PLACEHOLDER%"
}
```

> **Note**
>
> 可以放心地忽略包含 %FIREHOSE_POLICY_TEMPLATE_PLACEHOLDER% 的策略声明，因为它们不授予对任何资源的权限。

控制台创建和编辑 Firehose 流工作流程，也会创建信任策略并将其附加到 IAM 角色。此信任策略允许 Firehose 承担 IAM 角色。以下是信任策略的示例。

JSON

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

> **Important**
>
> - 您应避免为多个 Firehose 流使用相同的控制台托管 IAM 角色。否则，IAM 角色可能会变得权限过于宽松或导致错误。
> - 要在控制台托管的 IAM 角色的权限策略中使用不同的策略声明，您可以创建自己的 IAM 角色，并将策略声明复制到新角色所附加的权限策略中。要将角色附加到 Firehose 流，请在服务访问权限中选择选择现有 IAM 角色选项。

- 控制台管理其 ARN 中包含字符串 service-role 的 IAM 角色。选择现有 IAM 角色选项时，请务必选择一个 ARN 中没有 service-role 字符串的 IAM 角色，这样控制台就不会对其进行任何更改。

从控制台中创建 IAM 角色的步骤

1. 打开 Firehose 控制台，网址为。https://console.aws.amazon.com/firehose/
2. 选择创建 Firehose 流。
3. 选择源和目的地。有关更多信息，请参阅 教程：从控制台创建 Firehose 流。
4. 选择目的地设置。有关更多信息，请参阅 配置目的地设置。
5. 在高级设置下，针对服务访问权限，选择创建或更新 IAM 角色。

> **Note**
>
> 这是默认选项。要使用现有角色，请选择选择现有 IAM 角色选项。Firehose 控制台不会对您自己的角色进行任何更改。

6. 选择创建 Firehose 流。

使用控制台编辑 IAM 角色

当您编辑 Firehose 流时，Firehose 会根据情况更新相应的权限策略以反映配置和权限的更改。

例如，当您编辑 Firehose 流并将最新版本的 Lambda 函数用作 `exampleLambdaFunction` 来启用使用 AWS Lambda转换源记录功能时，会在权限策略中获得以下策略声明。

```
{
  "Sid": "lambdaProcessing",
  "Effect": "Allow",
  "Action": [
    "lambda:InvokeFunction",
    "lambda:GetFunctionConfiguration"
  ],
  "Resource": "arn:aws:lambda:us-east-1:123456789012:function:exampleLambdaFunction:
$LATEST"
}
```

> **Important**
>
> 将控制台管理的 IAM 角色设计为自主角色。建议不要在控制台之外修改权限策略或信任策略。

从控制台中编辑 IAM 角色的步骤

1. 打开 Firehose 控制台，网址为。https://console.aws.amazon.com/firehose/
2. 选择 Firehose 流并选择要更新的 Firehose 流的名称。
3. 在配置选项卡的服务器访问权限部分中，选择编辑。
4. 更新 IAM 角色选项。

> **Note**
>
> 默认情况下，控制台始终使用其 ARN 中的模式 service-role 更新 IAM 角色。选择现有 IAM 角色选项时，请务必选择一个 ARN 中没有 service-role 字符串的 IAM 角色，这样控制台就不会对其进行任何更改。

5. 选择保存更改。

了解 Amazon Data Firehose 的合规性

Third-party 作为多项合规计划的一部分，审计师会评估 Amazon Data Firehose 的安全 AWS 性和合规性。其中包括 SOC、PCI、FedRAMP、HIPAA 及其他。

有关特定合规计划范围内的 AWS 服务列表，请参阅合规性计划范围内的AWS 服务。有关一般信息，请参阅 AWS 合规性计划。

您可以使用下载第三方审计报告 AWS Artifact。有关更多信息，请参阅在 Artifac t 中 AWS 下载报告。

使用 Data Firehose 时，您的合规责任根据您的数据敏感性、公司的合规目标以及适用的法律法规决定。如果您对 Data Firehose 的使用必须符合 HIPAA、PCI 或 FedRAMP 等标准，请提供资源来帮助： AWS

- 安全与合规性快速入门指南 — 这些部署指南讨论了架构注意事项，并提供了在上部署以安全性和合规性为重点的基准环境的步骤。 AWS
- HIPAA 安全与合规架构白皮书 — 本白皮书描述了公司如何使用来创建应用程序。 AWS HIPAA-compliant

- AWS 合规资源 — 此工作簿和指南集可能适用于您所在的行业和所在地。
- AWS Config— 该 AWS 服务评估您的资源配置在多大程度上符合内部实践、行业指导方针和法规。
- AWS Security Hub CSPM— 此 AWS 服务可全面了解您的安全状态 AWS ，帮助您检查是否符合安全行业标准和最佳实践。

Amazon Data Firehose 中的弹性

AWS 全球基础设施是围绕 AWS 区域和可用区构建的。 AWS 区域提供多个物理隔离和隔离的可用区，这些可用区通过低延迟、高吞吐量和高度冗余的网络相连。利用可用区，您可以设计和操作在可用区之间无中断地自动实现失效转移的应用程序和数据库。与传统的单个或多个数据中心基础架构相比，可用区具有更高的可用性、容错性和可扩展性。

有关 AWS 区域和可用区的更多信息，请参阅AWS 全球基础设施。

除了 AWS 全球基础架构外，Data Firehose 还提供多项功能来帮助支持您的数据弹性和备份需求。

灾难恢复

Amazon Data Firehose 可在无服务器模式下运行，并通过执行自动迁移来处理主机降级、可用区可用性以及其他基础设施相关问题。如果发生这种情况，Amazon Data Firehose 可确保 Firehose 流迁移过程中不会丢失任何数据。

了解 Amazon Data Firehose 中的基础设施安全性

作为一项托管服务，Amazon Data Firehose 受到 AWS 全球网络安全的保护。有关 AWS 安全服务以及如何 AWS 保护基础设施的信息，请参阅AWS 云安全。要使用基础设施安全的最佳实践来设计您的 AWS 环境，请参阅 S AWS ecurity Pillar Well-Architected Fram ework 中的基础设施保护。

您可以使用 AWS 已发布的 API 调用通过网络访问 Firehose。客户端必须支持以下内容：

- 传输层安全性协议（TLS）。我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 具有完全向前保密 (PFS) 的密码套件，例如 DHE（短暂的）或 ECDHE（椭圆曲线短暂的 Diffie-Hellman）。 Diffie-Hellman大多数现代系统（如 Java 7 及更高版本）都支持这些模式。

> **Note**
>
> 对于传出的 HTTPS 请求，Amazon Data Firehose 将使用 HTTP 库，该库会自动选择目标端支持的最高 TLS 协议版本。

将 Amazon Data Firehose 与 AWS PrivateLink

您可以使用接口 VPC 终端节点 (AWS PrivateLink) 从您的 VPC 访问 Amazon Data Firehose，而无需互联网网关或 NAT 网关。接口 VPC 终端节点不需要互联网网关、NAT 设备、VPN 连接或 Direct Connect 连接。接口 VPC 终端节点由一项 AWS 技术提供支持 AWS PrivateLink，该技术允许使用弹性网络接口在 AWS 服务之间进行私密通信，并使用您的 Amazon VPC 中的私有 IP。有关更多信息，请参阅 Amazon Virtual Private Cloud。

使用接口 VPC 终端节点 (AWS PrivateLink) 适用于 Firehose

首先，创建一个接口 VPC 端点，以便来自 Amazon VPC 资源的 Amazon Data Firehose 流量开始流经接口 VPC 端点。创建端点时，可以为其附加端点策略，以控制对 Amazon Data Firehose 的访问。有关使用策略控制 VPC 端点对 Amazon Data Firehose 的访问的更多信息，请参阅使用 VPC 端点控制对服务的访问。

以下示例展示了如何在 VPC 中设置 AWS Lambda 函数并创建 VPC 终端节点，以允许该函数与 Amazon Data Firehose 服务进行安全通信。在本例中，您使用的策略允许 Lambda 函数列出当前区域中的 Firehose 流，但不描述任何 Firehose 流。

创建 VPC 端点

1. 登录 AWS 管理控制台 并打开 Amazon VPC 控制台，网址为https://console.aws.amazon.com/vpc/。
2. 在 VPC 控制面板中，选择 Endpoints (终端节点)。
3. 选择创建端点。
4. 在服务名称列表中，选择 `com.amazonaws.`*`your_region`*`.kinesis-firehose`。
5. 选择要在其中创建终端节点的 VPC 以及一个或多个子网。
6. 选择一个或多个要与终端节点关联的安全组。
7. 对于 Policy (策略)，选择 Custom (自定义) 并粘贴以下策略：

```
{
```

```
    "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. 选择创建端点。

创建用于 Lambda 函数的 IAM 角色

1. 使用 https://console.aws.amazon.com/iam/ 打开 IAM 控制台。
2. 在左侧窗格中，选择角色，然后选择创建角色。
3. 在选择可信实体的类型下，保留默认选择 AWS 服务。
4. 在选择将使用此角色的服务下，选择 Lambda。
5. 选择 Next: Permissions（下一步: 权限）。
6. 在策略列表中，搜索并添加名为 `AWS LambdaVPCAccessExecutionRole` 和 `AmazonDataFirehoseReadOnlyAccess` 的两个策略。

> **Important**
>
> 下面给出了一个示例：您的生产环境可能需要更严格的策略。

7. 选择下一步：标签。在本练习中，您不需要添加标签。选择下一步：审核。
8. 输入角色的名称，然后选择创建角色。

在 VPC 中创建 Lambda 函数

1. 打开 AWS Lambda 控制台，网址为https://console.aws.amazon.com/lambda/。
2. 选择创建函数。
3. 选择从头开始创作。
4. 输入函数的名称，然后将运行时设置为 Python 3.9 或更高版本。
5. 在权限下，展开选择或创建执行角色。
6. 在 Execution role (执行角色) 列表中，选择 Use an existing role (使用现有角色)。
7. 在 Existing role (现有角色) 列表中，选择您先前创建的角色。
8. 选择创建函数。
9. 在 Function code (函数代码) 下，粘贴以下代码。

```
    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. 在 Basic settings (基本设置) 下，将超时时间设置为 1 分钟。
11. 在 Network (网络) 中，选择您之前在其中创建终端节点的 VPC，然后选择在您创建终端节点时与其关联的子网和安全组。
12. 在页面顶部附近，选择保存。
13. 选择测试。
14. 输入事件名称，然后选择创建。
15. 再次选择 Test (测试)。这将使该函数运行。在执行结果显示之后，展开 Details (详细信息)，并将日志输出与函数代码进行比较。运行成功的结果将显示区域中的 Firehose 流的列表以及以下输出：

```
Calling describe_delivery_stream.

AccessDeniedException

Access denied test succeeded.
```

支持 AWS 区域

接口 VPC 终端节点当前在以下区域受支持。

- 美国东部（俄亥俄州）
- 美国东部（弗吉尼亚州北部）

- 美国西部（北加利福尼亚）
- 美国西部（俄勒冈州）
- 亚太地区（孟买）
- 亚太地区（首尔）
- 亚太地区（新加坡）
- 亚太地区（悉尼）
- 亚太地区（泰国）
- 亚太地区（东京）
- 亚太地区（香港）
- 加拿大（中部）
- 加拿大西部（卡尔加里）
- 中国（北京）
- 中国（宁夏）
- 欧洲地区（法兰克福）
- 欧洲地区（爱尔兰）
- 欧洲地区（伦敦）
- Europe (Paris)
- 墨西哥（中部）
- 南美洲（圣保罗）
- AWS GovCloud (US-East)
- AWS GovCloud (US-West)
- 欧洲（西班牙）
- 中东（阿联酋）：
- 亚太地区（雅加达）
- 亚太地区（大阪）
- 以色列（特拉维夫）
- 亚太地区（马来西亚）

实施 Amazon Data Firehose 的安全最佳实践

Amazon Data Firehose 提供了许多安全功能，供您在开发和实施自己的安全策略时考虑。以下最佳实践是一般指导原则，并不代表完整安全解决方案。由于这些最佳实践可能不适合您的环境或不满足您的环境要求，因此将其视为有用的考虑因素而不是惯例。

实施最低权限访问

授予权限时，您可以决定谁将获得对哪些 Amazon Data Firehose 资源的哪些权限。您可以对这些资源启用希望允许的特定操作。因此，您应仅授予执行任务所需的权限。实施最低权限访问对于减小安全风险以及可能由错误或恶意意图造成的影响至关重要。

使用 IAM 角色

生产者和客户端应用程序必须具有有效凭证才能访问 Firehose 流，并且您的 Firehose 流必须具有有效凭证才能访问目标。您不应将 AWS 证书直接存储在客户端应用程序或 Amazon S3 存储桶中。这些是不会自动轮换的长期凭证，如果它们受到损害，可能会对业务产生重大影响。

而是应该使用 IAM 角色来管理生产者和客户端应用程序的临时凭证，以访问 Firehose 流。在使用角色时，您不必使用长期凭证（如用户名和密码或访问密钥）来访问其他资源。

有关更多信息，请参阅 IAM 用户指南中的以下主题：

- IAM 角色
- 针对角色的常见情形：用户、应用程序和服务

实现从属资源中的服务器端加密

可在 Amazon Data Firehose 中加密静态数据和传输中数据。有关更多信息，请参阅 Amazon Data Firehose 中的数据保护。

CloudTrail 用于监控 API 调用

Amazon Data Firehose 与 AWS CloudTrail一项服务集成，可记录用户、角色或 AWS 服务在 Amazon Data Firehose 中采取的操作。

通过收集的信息 CloudTrail，您可以确定向 Amazon Data Firehose 发出的请求、发出请求的 IP 地址、谁提出了请求、何时提出请求以及其他详细信息。

有关更多信息，请参阅 the section called “日志记录 Firehose API 调用”。

监控 Amazon Data Firehose

您可以使用以下功能监控 Amazon Data Firehose：

主题

使用 CloudWatch 警报实施最佳实践

添加 CloudWatch 警报，告知以下指标何时超过缓冲限制（最长 15 分钟）。

- `DeliveryToS3.DataFreshness`
- `DeliveryToIceberg.DataFreshness`
- `DeliveryToSplunk.DataFreshness`
- `DeliveryToAmazonOpenSearchService.DataFreshness`
- `DeliveryToAmazonOpenSearchServerless.DataFreshness`
- `DeliveryToHttpEndpoint.DataFreshness`

此外，还要基于以下指标数学表达式创建警报。

- `IncomingBytes (Sum per 5 Minutes) / 300` 接近 `BytesPerSecondLimit` 百分比。
- `IncomingRecords (Sum per 5 Minutes) / 300` 接近 `RecordsPerSecondLimit` 百分比。
- `IncomingPutRequests (Sum per 5 Minutes) / 300` 接近 `PutRequestsPerSecondLimit` 百分比。

我们建议作为根据发出警报的另一个指标是 `ThrottledRecords`。

有关在警告进入 ALARM 状态时进行故障排查的信息，请参阅排查错误。

使用指标监控亚马逊数据 Firehose CloudWatch

> **Important**
>
> 请务必对属于您的目的地的所有 CloudWatch 指标启用警报，以便及时发现错误。

Amazon Data Firehose 与亚马逊 CloudWatch 指标集成，因此您可以收集、查看和分析 Firehose 直播的 CloudWatch 指标。例如，您可以监控 `IncomingBytes` 和 `IncomingRecords` 指标，以跟踪从数据生产者摄取到 Amazon Data Firehose 的数据。

Amazon Data Firehose 每分钟收集和发布一次 CloudWatch 指标。但是，如果传入数据突发仅持续几秒钟，则可能无法在一分钟指标中完全捕获或可见。这是因为 CloudWatch 指标是在一分钟内从 Amazon Data Firehose 汇总的。

为 Firehose 流收集的指标不收费。有关 Kinesis 代理指标的更多信息，请参阅 监控 Kinesis 代理运行状况。

主题

CloudWatch 动态分区的指标

如果启用了动态分区，则 AWS/Firehose 命名空间将包含以下指标。

指标	说明
ActivePartitionsLimit	Firehose 流在将数据发送到错误存储桶之前处理的最大活动分区数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
PartitionCount	正在处理的分区数，也即活动分区数。此数字在 1 和分区计数限制 500（默认）之间变化。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
PartitionCountExceeded	此指标指示您是否超出了分区计数限制。根据是否违反限制发出 1 或 0。
JQProcessing.Duration	返回在 JQ Lambda 函数中执行 JQ 表达式所花的时间。 单位：毫秒
PerPartitionThroughput	表示每个分区正在处理的吞吐量。此指标使您能够监控每个分区的吞吐量。 单位： StandardUnit。 BytesSecond
DeliveryToS3.ObjectCount	指示正在传输到 S3 存储桶的对象数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数

CloudWatch 数据传输指标

AWS/Firehose 命名空间包括以下服务级指标。如果您看到 BackupToS3.Success、DeliveryToS3.Success、DeliveryToSplunk.Success、DeliveryToAma 或 DeliveryToRedshift.Success 的平均值小幅下降，这并不表示存在数据丢失。Amazon Data Firehose 会重试传输错误，并且在记录成功传输到配置的目的地或备份 S3 存储桶之前不会继续前进。

主题

送货到 OpenSearch 服务

指标	说明
DeliveryToAmazonOpenSearchService.Bytes	在指定时间段内索引到 S OpenSearch ervice 的字节数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
DeliveryToAmazonOpenSearchService.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何超过此年龄的记录都已交付给 OpenSearch 服务部门。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：秒

指标	说明
DeliveryToAmazonOpenSearchService.Records	在指定时间段内编入 OpenSearch 服务索引的记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToAmazonOpenSearchService.Success	已成功编制索引的记录的总和。
DeliveryToS3.Bytes	在指定时间段内传输到 Amazon S3 的字节数。仅当您为所有文档启用备份后，Amazon Data Firehose 才会发出此指标。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToS3.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何大于此年龄的记录已传送到 S3 存储桶。仅当您为所有文档启用备份后，Amazon Data Firehose 才会发出此指标。 单位：秒
DeliveryToS3.Records	在指定时间段内传输到 Amazon S3 的记录数。仅当您为所有文档启用备份后，Amazon Data Firehose 才会发出此指标。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToS3.Success	成功 Amazon S3 put 命令的总和。无论是仅对失败的文档还是对所有文档启用备份，Amazon Data Firehose 始终会发出此指标。

指标	说明
DeliveryToAmazonOpenSearchService.AuthFailure	Authentication/authorization error. Verify the OS/ES集群策略和角色权限。 0 表示没有问题。1 表示身份验证失败。
DeliveryToAmazonOpenSearchService.DeliveryRejected	传输被拒绝错误。验证集 OS/ES 群策略和角色权限。 0 表示没有问题。1 表示传输失败。

交付到 OpenSearch 无服务器

指标	说明
DeliveryToAmazonOpenSearchServerless.Bytes	在指定时间段内索引到 OpenSearch Serverless 的字节数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
DeliveryToAmazonOpenSearchServerless.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何超过这个年龄的记录都已发送到 OpenSearch Serverless。 单位：秒
DeliveryToAmazonOpenSearchServerless.Records	在指定时间段内索引到 OpenSearch Serverless 的记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数

指标	说明
DeliveryToAmazonOpenSearchServerless.Success	已成功编制索引的记录的总和。
DeliveryToS3.Bytes	在指定时间段内传输到 Amazon S3 的字节数。仅当您为所有文档启用备份后，Amazon Data Firehose 才会发出此指标。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToS3.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何大于此年龄的记录已传送到 S3 存储桶。仅当您为所有文档启用备份后，Amazon Data Firehose 才会发出此指标。 单位：秒
DeliveryToS3.Records	在指定时间段内传输到 Amazon S3 的记录数。仅当您为所有文档启用备份后，Amazon Data Firehose 才会发出此指标。 单位：计数
DeliveryToS3.Success	成功 Amazon S3 put 命令的总和。无论是仅对失败的文档还是对所有文档启用备份，Amazon Data Firehose 始终会发出此指标。
DeliveryToAmazonOpenSearchServerless.AuthFailure	Authentication/authorization error. Verify the OS/ES集群策略和角色权限。 0 表示没有问题。1 表示身份验证失败。
DeliveryToAmazonOpenSearchServerless.DeliveryRejected	传输被拒绝错误。验证集 OS/ES 群策略和角色权限。 0 表示没有问题。1 表示传输失败。

传输到 Amazon Redshift

指标	说明
DeliveryToRedshift.Bytes	在指定时间段内复制到 Amazon Redshift 的字节数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToRedshift.Records	在指定时间段内复制到 Amazon Redshift 的记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToRedshift.Success	成功 Amazon Redshift COPY 命令的总和。
DeliveryToS3.Bytes	在指定时间段内传输到 Amazon S3 的字节数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
DeliveryToS3.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何大于此期限的记录都传送到 S3 存储桶。 单位：秒
DeliveryToS3.Records	在指定时间段内传输到 Amazon S3 的记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数

指标	说明
DeliveryToS3.Success	成功 Amazon S3 put 命令的总和。
DeliveryToRedshift.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何大于此期限的记录都传送到 Amazon Redshift 集群。
BackupToS3.Bytes	在指定时间段内传输到 Amazon S3 进行备份的字节数。启用备份到 Amazon S3 后，Amazon Data Firehose 会发出此指标。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
BackupToS3.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何早于此期限的记录均已传输到 Amazon S3 存储桶进行备份。启用备份到 Amazon S3 后，Amazon Data Firehose 会发出此指标。 统计数据：Minimum、Maximum、Average、Samples 单位：秒
BackupToS3.Records	在指定时间段内传输到 Amazon S3 进行备份的记录数。启用备份到 Amazon S3 后，Amazon Data Firehose 会发出此指标。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
BackupToS3.Success	成功的 Amazon S3 备份 put 命令的总和。启用备份到 Amazon S3 后，Amazon Data Firehose 会发出此指标。

传输到 Amazon S3

下表中的指标与 Amazon S3 是 Firehose 流的主要目的地时的传输相关。

指标	说明
DeliveryToS3.Bytes	在指定时间段内传输到 Amazon S3 的字节数。启用数据转换后，该指标会反映转换前预处理的字节大小。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
DeliveryToS3.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何大于此年龄的记录已传送到 S3 存储桶。 统计数据：Minimum、Maximum、Average、Samples 单位：秒
DeliveryToS3.Records	在指定时间段内传输到 Amazon S3 的记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToS3.Success	成功 Amazon S3 put 命令的总和。
BackupToS3.Bytes	在指定时间段内传输到 Amazon S3 进行备份的字节数。启用备份后，Amazon Data Firehose 会发出此指标（只有在同时启用数据转换的情况下才可以）。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数

指标	说明
BackupToS3.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何早于此期限的记录均已传输到 Amazon S3 存储桶进行备份。启用备份后，Amazon Data Firehose 会发出此指标（只有在同时启用数据转换的情况下才可以）。 统计数据：Minimum、Maximum、Average、Samples 单位：秒
BackupToS3.Records	在指定时间段内传输到 Amazon S3 进行备份的记录数。启用备份后，Amazon Data Firehose 会发出此指标（只有在同时启用数据转换的情况下才可以）。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
BackupToS3.Success	成功的 Amazon S3 备份 put 命令的总和。启用备份后，Amazon Data Firehose 会发出此指标（只有在同时启用数据转换的情况下才可以）。

传输到 Snowflake

指标	说明
DeliveryToSnowflake.Bytes	在指定时段内传输到 Snowflake 的字节数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
DeliveryToSnowflake.DataFreshness	Firehose 中最早记录的期限（从进入 Firehose 到现在）。任何大于此时间的记录已传输到 Snowflake。请注意，在 Firehose 插入调用成功后，可能需要几秒钟才能将数

指标	说明
	据提交给 Snowflake。有关向 Snowflake 提交数据所需的时间，请参阅 DeliveryToSnowflake.DataCommitLatency 指标。 统计数据：Minimum、Maximum、Average、Samples 单位：秒
DeliveryToSnowflake.DataCommitLatency	Firehose 成功插入记录后，将数据提交到 Snowflake 所需的时间。 统计数据：Minimum、Maximum、Average、Samples 单位：秒
DeliveryToSnowflake.Records	在指定时间段内传输到 Snowflake 的记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToSnowflake.Success	对 Snowflake 进行的成功插入调用的总和。
DeliveryToS3.Bytes	在指定时间段内传输到 Amazon S3 的字节数。仅当向 Snowflake 传输失败且 Firehose 尝试将失败的数据备份到 S3 时，此指标才可用。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节

指标	说明
DeliveryToS3.Records	在指定时间段内传输到 Amazon S3 的记录数。仅当向 Snowflake 传输失败且 Firehose 尝试将失败的数据备份到 S3 时，此指标才可用。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToS3.Success	成功 Amazon S3 put 命令的总和。仅当向 Snowflake 传输失败且 Firehose 尝试将失败的数据备份到 S3 时，此指标才可用。
BackupToS3.DataFreshness	Firehose 中最早记录的期限（从进入 Firehose 到现在）。任何早于此期限的记录均已备份到 Amazon S3 存储桶。当 Firehose 流配置为备份所有数据时，此指标可用。 统计数据：Minimum、Maximum、Average、Samples 单位：秒
BackupToS3.Records	在指定时间段内传输到 Amazon S3 进行备份的记录数。当 Firehose 流配置为备份所有数据时，此指标可用。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
BackupToS3.Bytes	在指定时间段内传输到 Amazon S3 进行备份的字节数。当 Firehose 流配置为备份所有数据时，此指标可用。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数

指标	说明
BackupToS3.Success	成功的 Amazon S3 备份 put 命令的总和。当 Firehose 流配置为备份所有数据时，Firehose 会发出此指标。

传输到 Splunk

指标	说明
DeliveryToSplunk.Bytes	在指定时段内传送到 Splunk 的字节数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
DeliveryToSplunk.DataAckLatency	Amazon Data Firehose 向 Splunk 发送数据后，从 Splunk 接收确认所需的大致时间。此指标的增加或减少趋势比绝对近似值更有用。增长趋势可能表明 Splunk 索引器的索引和确认速度较慢。 统计数据：Minimum、Maximum、Average、Samples 单位：秒
DeliveryToSplunk.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何大于此年龄的记录已传送到 Splunk。 统计数据：Minimum、Maximum、Average、Samples 单位：秒
DeliveryToSplunk.Records	在指定时段内传送到 Splunk 的记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数

指标	说明
DeliveryToSplunk.Success	已成功编制索引的记录的总和。
DeliveryToS3.Success	成功 Amazon S3 put 命令的总和。启用备份到 Amazon S3 时，将发出此指标。
BackupToS3.Bytes	在指定时间段内传输到 Amazon S3 进行备份的字节数。当 Firehose 流配置为备份所有文档时，Amazon Data Firehose 会发出此指标。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
BackupToS3.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何早于此期限的记录均已传输到 Amazon S3 存储桶进行备份。当 Firehose 流配置为备份所有文档时，Amazon Data Firehose 会发出此指标。 统计数据：Minimum、Maximum、Average、Samples 单位：秒
BackupToS3.Records	在指定时间段内传输到 Amazon S3 进行备份的记录数。当 Firehose 流配置为备份所有文档时，Amazon Data Firehose 会发出此指标。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
BackupToS3.Success	成功的 Amazon S3 备份 put 命令的总和。当 Firehose 流配置为备份所有文档时，Amazon Data Firehose 会发出此指标。

传输到 HTTP 端点

指标	说明
DeliveryToHttpEndpoint.Bytes	成功传输到 HTTP 端点的字节数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
DeliveryToHttpEndpoint.Records	成功传输到 HTTP 端点的记录数。此指标仅在成功尝试交付时发出，并且在交付尝试失败时不会发出。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToHttpEndpoint.DataFreshness	Amazon Data Firehose 中最早记录的期限。 统计数据：Minimum、Maximum、Average、Samples 单位：秒
DeliveryToHttpEndpoint.Success	每次尝试传送成功传送到 HTTP 端点的记录数。与之不同的是DeliveryToHttpEndpoint.Records ，每次尝试交付都会发出此指标。成功后，该值等于传送尝试中的记录数。如果传送尝试中的所有记录都失败，则值为 0。使用最小值统计数据来监控传送失败。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToHttpEndpoint.ProcessedBytes	尝试处理的字节数，包括重试次数。

指标	说明
DeliveryToHttpEndpoint.ProcessedRecords	尝试的记录数，包括重试次数。

数据摄取指标

主题

- 通过 Kinesis Data Streams 进行的数据摄取
- 通过 Direct PUT 进行的数据摄取
- 从 MSK 进行的数据摄取

通过 Kinesis Data Streams 进行的数据摄取

指标	说明
DataReadFromKinesisStream.Bytes	当数据来源是 Kinesis 数据流时，此指标指示从该数据流读取的字节数。此数字包括由于故障转移而重新读取的数量。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
DataReadFromKinesisStream.Records	当数据来源是 Kinesis 数据流时，此指标指示从该 Kinesis 数据流读取的记录数。此数字包括由于故障转移而重新读取的数量。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
ThrottledDescribeStream	当数据来源是 Kinesis 数据流时，DescribeStream 操作受到限制的总次数。

指标	说明
	统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
ThrottledGetRecords	当数据来源是 Kinesis 数据流时，GetRecords 操作受到限制的总次数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
ThrottledGetShardIterator	当数据来源是 Kinesis 数据流时，GetShardIterator 操作受到限制的总次数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
KinesisMillisBehindLatest	当数据来源是 Kinesis 数据流时，此度量标准指示最后一条读取记录落后于 Kinesis 数据流中的最新记录的毫秒数。 统计数据：Minimum、Maximum、Average、Samples 单位：毫秒

通过 Direct PUT 进行的数据摄取

指标	说明
BackupToS3.Bytes	在指定时间段内传输到 Amazon S3 进行备份的字节数。当为 Amazon S3 或 Amazon Redshift 目的地启用数据转换时，Amazon Data Firehose 会发出此指标。

指标	说明
	统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
BackupToS3.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何早于此期限的记录均已传输到 Amazon S3 存储桶进行备份。当为 Amazon S3 或 Amazon Redshift 目的地启用数据转换时，Amazon Data Firehose 会发出此指标。 统计数据：Minimum、Maximum、Average、Samples 单位：秒
BackupToS3.Records	在指定时间段内传输到 Amazon S3 进行备份的记录数。当为 Amazon S3 或 Amazon Redshift 目的地启用数据转换时，Amazon Data Firehose 会发出此指标。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
BackupToS3.Success	成功的 Amazon S3 备份 put 命令的总和。当为 Amazon S3 或 Amazon Redshift 目的地启用数据转换时，Amazon Data Firehose 会发出此指标。
BytesPerSecondLimit	在限制之前 Firehose 流当前每秒可以摄取的最大字节数。要请求提高此限制，请转至 AWS Support 中心并选择创建案例，然后选择提高服务限制。
DeliveryToAmazonOpenSearchService.Bytes	在指定时间段内索引到 S OpenSearch ervice 的字节数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节

指标	说明
DeliveryToAmazonOpenSearchService.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何超过此年龄的记录都已交付给 OpenSearch 服务部门。 统计数据：Minimum、Maximum、Average、Samples 单位：秒
DeliveryToAmazonOpenSearchService.Records	在指定时间段内编入 OpenSearch 服务索引的记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToAmazonOpenSearchService.Success	已成功编制索引的记录的总和。
DeliveryToRedshift.Bytes	在指定时间段内复制到 Amazon Redshift 的字节数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
DeliveryToRedshift.Records	在指定时间段内复制到 Amazon Redshift 的记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToRedshift.Success	成功 Amazon Redshift COPY 命令的总和。

指标	说明
DeliveryToS3.Bytes	在指定时间段内传输到 Amazon S3 的字节数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
DeliveryToS3.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何大于此年龄的记录已传送到 S3 存储桶。 统计数据：Minimum、Maximum、Average、Samples 单位：秒
DeliveryToS3.Records	在指定时间段内传输到 Amazon S3 的记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToS3.Success	成功 Amazon S3 put 命令的总和。
DeliveryToSplunk.Bytes	在指定时段内传送到 Splunk 的字节数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
DeliveryToSplunk.DataAckLatency	Amazon Data Firehose 向 Splunk 发送数据后，从 Splunk 接收确认所需的大致时间。此指标的增加或减少趋势比绝对近似值更有用。增长趋势可能表明 Splunk 索引器的索引和确认速度较慢。 统计数据：Minimum、Maximum、Average、Samples 单位：秒

指标	说明
DeliveryToSplunk.DataFreshness	Amazon Data Firehose 中最早记录的期限（从进入 Amazon Data Firehose 到现在）。任何大于此年龄的记录已传送到 Splunk。 统计数据：Minimum、Maximum、Average、Samples 单位：秒
DeliveryToSplunk.Records	在指定时段内传送到 Splunk 的记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DeliveryToSplunk.Success	已成功编制索引的记录的总和。
IncomingBytes	在指定时段内成功摄取到 Firehose 流的字节数。当数据摄取超过 Firehose 流限制之一时，可能会受到限制。受限制的数据将不计为 IncomingBytes 。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
IncomingPutRequests	指定时间段内的成功 PutRecordBatch请求数 PutRecord 和请求数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数

指标	说明
IncomingRecords	在指定时段内成功摄取到 Firehose 流的记录数。当数据摄取超过 Firehose 流限制之一时，可能会受到限制。受限制的数据将不计为 IncomingRecords 。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
RecordsPerSecondLimit	在限制之前 Firehose 流当前每秒可以摄取的最大记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
ThrottledRecords	由于数据摄取超过其中一个 Firehose 流限制而受到限制的记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数

从 MSK 进行的数据摄取

指标	说明
DataReadFromSource.Records	从源 Kafka 主题读取的记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
DataReadFromSource.Bytes	从源 Kafka 主题读取的字节数。

指标	说明
	统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
SourceThrottled.Delay	源 Kafka 集群从源 Kafka 主题返回记录的延迟时间。 统计数据：Minimum、Maximum、Average、Samples 单位：毫秒
BytesPerSecondLimit	Firehose 从源 Kafka 主题的每个分区读取数据的当前吞吐量限制。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节/秒
KafkaOffsetLag	Firehose 从源 Kafka 主题读取的记录的最大偏移量与源 Kafka 主题可用记录的最大偏移量之间的差异。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
FailedValidation.Records	记录验证失败的记录数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
FailedValidation.Bytes	记录验证失败的字节数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节

指标	说明
DataReadFromSource.Backpressured	表示 Firehose 流延迟从源分区读取记录，要么是因为 BytesPerSecondLimit 每个分区已超过，要么是正常的传送流程缓慢或已停止 单位：布尔值

API 级别 CloudWatch 的指标

AWS/Firehose 命名空间包括以下 API 级指标。

指标	说明
DescribeDeliveryStream.Latency	在指定时段内测量的每个 DescribeDeliveryStream 操作所用的时间。 统计数据：Minimum、Maximum、Average、Samples 单位：毫秒
DescribeDeliveryStream.Requests	DescribeDeliveryStream 请求的总数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
ListDeliveryStreams.Latency	在指定时段内测量的每个 ListDeliveryStream 操作所用的时间。 统计数据：Minimum、Maximum、Average、Samples 单位：毫秒
ListDeliveryStreams.Requests	ListFirehose 请求的总数。 统计数据：Minimum、Maximum、Average、Sum、Samples

指标	说明
	单位：计数
PutRecord.Bytes	在指定时段内使用 PutRecord 放入 Firehose 流的字节数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
PutRecord.Latency	在指定时段内测量的每个 PutRecord 操作所用的时间。 统计数据：Minimum、Maximum、Average、Samples 单位：毫秒
PutRecord.Requests	PutRecord 请求的总数，此数目等于来自 PutRecord 操作的记录的总数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
PutRecordBatch.Bytes	在指定时段内使用 PutRecordBatch 放入 Firehose 流的字节数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
PutRecordBatch.Latency	在指定时段内测量的每个 PutRecordBatch 操作所用的时间。 统计数据：Minimum、Maximum、Average、Samples 单位：毫秒

指标	说明
PutRecordBatch.Records	来自 PutRecordBatch 操作的记录的总数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
PutRecordBatch.Requests	PutRecordBatch 请求的总数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
PutRequestsPerSecondLimit	在限制之前 Firehose 流可以处理的每秒最大输入请求数。此数字包括 PutRecord 和 PutRecordBatch 请求。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
ThrottledDescribeStream	当数据来源是 Kinesis 数据流时，DescribeStream 操作受到限制的总次数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
ThrottledGetRecords	当数据来源是 Kinesis 数据流时，GetRecords 操作受到限制的总次数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数

指标	说明
ThrottledGetShardIterator	当数据来源是 Kinesis 数据流时，GetShardIterator 操作受到限制的总次数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
UpdateDeliveryStream.Latency	在指定时段内测量的每个 UpdateDeliveryStream 操作所用的时间。 统计数据：Minimum、Maximum、Average、Samples 单位：毫秒
UpdateDeliveryStream.Requests	UpdateDeliveryStream 请求的总数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数

数据转换 CloudWatch 指标

如果启用 Lambda 数据转换，AWS/Firehose 命名空间将包含以下指标。

指标	说明
ExecuteProcessing.Duration	Firehose 执行每次 Lambda 函数调用所花的时间。 单位：毫秒
ExecuteProcessing.Success	成功的 Lambda 函数调用总和相比总的 Lambda 函数调用次数。

指标	说明
SucceedProcessing.Records	在指定时间段内成功处理的记录数。 单位：计数
SucceedProcessing.Bytes	在指定时间段内成功处理的字节数。 单位：字节

CloudWatch 日志解压缩指标

如果为 CloudWatch 日志传输启用了解压缩，则AWS/Firehose命名空间将包含以下指标。

指标	说明
OutputDecompressedBytes.Success	解压缩成功的数据（以字节为单位） 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
OutputDecompressedBytes.Failed	解压缩失败的数据（以字节为单位） 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：字节
OutputDecompressedRecords.Success	解压缩成功的记录数量 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
OutputDecompressedRecords.Failed	解压缩失败的记录数量

指标	说明
	统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数

格式化转化 CloudWatch 指标

如果启用了格式转换，AWS/Firehose 命名空间会包括以下指标。

指标	说明
SucceedConversion.Records	成功转换的记录的数量。 单位：计数
SucceedConversion.Bytes	成功转换的记录的大小。 单位：字节
FailedConversion.Records	未能转换的记录的数量。 单位：计数
FailedConversion.Bytes	未能转换的记录的大小。 单位：字节

服务器端加密 (SSE) 指标 CloudWatch

AWS/Firehose 命名空间包括以下与 SSE 相关的指标。

指标	说明
KMSKeyAccessDenied	服务遇到 Firehose 流的 KMSAccessDeniedException 的次数。

指标	说明
	统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
KMSKeyDisabled	服务遇到 Firehose 流的 KMSDisabledException 的次数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
KMSKeyInvalidState	服务遇到 Firehose 流的 KMSInvalidStateException 的次数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数
KMSKeyNotFound	服务遇到 Firehose 流的 KMSNotFoundException 的次数。 统计数据：Minimum、Maximum、Average、Sum、Samples 单位：计数

Amazon Data Firehose 的维度

要按 Firehose 流筛选指标，请使用 DeliveryStreamName 维度。

Amazon Data Firehose 用量指标

您可以使用 CloudWatch 使用量指标来了解您的账户的资源使用情况。使用这些指标在 CloudWatch 图表和仪表板上可视化您当前的服务使用情况。

服务配额使用情况指标位于 AWS/Usage 命名空间中，每三分钟收集一次。

目前，该命名空间中唯一 CloudWatch 发布的指标名称是ResourceCount。此指标与 Service、Class、Type 和 Resource 维度一同发布。

指标	描述
ResourceCount	您账户中运行的指定资源的数量。资源由与指标关联的维度定义。 此指标最有用的统计数据是 MAXIMUM，它表示在 3 分钟内使用的最大资源数。

以下维度用于优化 Amazon Data Firehose 发布的用量指标。

维度	说明
Service	包含资源的 AWS 服务的名称。对于 Amazon Data Firehose 用量指标，此维度的值为 Firehose。
Class	所跟踪的资源的类。Amazon Data Firehose API 用量指标使用值为 None 的维度。
Type	所跟踪的资源的类型。当前，当服务维度为 Firehose 时，类型的唯一有效值为 Resource。
Resource	AWS 资源的名称。目前，当服务维度为 Firehose 时，资源的唯一有效值为 DeliveryStreams 。

亚马逊 Data Firehose 的访问 CloudWatch 指标

您可以使用 CloudWatch 控制台、命令行或 CloudWatch API 监控 Amazon Data Firehose 的指标。以下过程介绍如何使用这些不同的方式访问指标。

使用 CloudWatch 控制台访问指标

1. 打开 CloudWatch 控制台，网址为https://console.aws.amazon.com/cloudwatch/。

2. 在导航栏中，选择一个区域。
3. 在导航窗格中，选择指标。
4. 选择 Firehose 命名空间。
5. 选择 Firehose 流指标或 Firehose 指标。
6. 选择要添加到图表的指标。

要访问指标，请使用 AWS CLI

使用列表指标和命令。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
```

使用日志监控亚马逊数据 Firehose CloudWatch

Amazon Data Firehose 与亚马逊 CloudWatch 日志集成，因此当用于数据转换或数据传输的 Lambda 调用失败时，您可以查看特定的错误日志。当您创建 Firehose 流时，可以启用 Amazon Data Firehose 错误日志记录。

在 Amazon Data Firehose 控制台中启用 Amazon Data Firehose 错误日志记录时，系统会代表您为 Firehose 流创建日志组和相应的日志流。日志组名称的格式为 `/aws/kinesisfirehose/`*`delivery-stream-name`*，其中 *`delivery-stream-name`* 是相应 Firehose 流的名称。`DestinationDelivery` 是创建的日志流，用于日志记录与传输到主要目的地相关的所有错误。只有在为目标启用了 S3 备份时，才会创建另一个名为 `BackupDelivery` 的日志流。`BackupDelivery` 日志流用于记录与传输到 S3 备份相关的任何错误。

例如，如果您创建了一个以 Amazon Redshift 为目标的 Firehose 流 MyStream “”，并启用 Amazon Data Firehose 错误日志，则会代表您创建以下内容：一个名为的日志组和两个`aws/kinesisfirehose/MyStream`名为和的日志流。`DestinationDelivery` BackupDelivery在本例中，`DestinationDelivery` 用于记录与传输到 Amazon Redshift 目标以及中间 S3 目标相关的任何错误。如果启用了 S3 备份，`BackupDelivery` 用来记录与传输到 S3 备份存储桶相关的任何错误。

您可以通过 AWS CLI、API 或 CloudFormation 使用配置启用 Amazon Data Firehose 错误记录。`CloudWatchLoggingOptions`为此，请提前创建日志组和日志流。建议保留该日志组

和日志流用于 Amazon Data Firehose 错误日志记录。此外，还要确保关联的 IAM policy 拥有 "logs:putLogEvents" 权限。有关更多信息，请参阅 使用 Amazon Data Firehose 控制访问权限。

请注意，Amazon Data Firehose 不保证所有传送错误日志都会发送到 CloudWatch 日志。在交付失败率很高的情况下，Amazon Data Firehose 会先对传送错误日志进行采样，然后再将其发送到 CloudWatch 日志。

发送到 CloudWatch Logs 的错误日志会收取象征性的费用。有关更多信息，请参阅 Amazon CloudWatch 定价。

内容

数据传输错误

以下是每个 Amazon Data Firehose 目的地的数据传输错误代码和消息的列表。每个错误消息还描述了修复问题所应采取的适当操作。

错误

Amazon S3 数据传输错误

Amazon Data Firehose 可以将以下与亚马逊 S3 相关的错误发送到日志。 CloudWatch

错误代码	错误消息和信息
`S3.KMS.NotFoundException`	“找不到提供的 AWS KMS 密钥。如果你使用的是你认为有效的 AWS KMS 密钥和正确的角色，请检查关联 AWS KMS 密钥的账户是否有问题。”
`S3.KMS.RequestLimitExceeded`	"The KMS request per second limit was exceeded while attempting to encrypt S3 objects。Increase the request per second limit." 有关更多信息，请参阅《AWS Key Management Service 开发人员指南》中的限制。
`S3.AccessDenied`	"Access was denied。Ensure that the trust policy for the provided IAM role allows Amazon Data Firehose to assume the role, and the access policy allows access to the S3 bucket."
`S3.AccountProblem`	“您的 AWS 账户存在问题，导致操作无法成功完成。Contact AWS Support."
`S3.AllAccessDisabled`	"Access to the account provided has been disabled。请联系 S AWS upport。”
`S3.InvalidPayer`	"Access to the account provided has been disabled。请联系 S AWS upport。”
`S3.NotSignedUp`	"The account is not signed up for Amazon S3. Sign the account up or use a different account."
`S3.NoSuchBucket`	"The specified bucket does not exist。Create the bucket or use a different bucket that does exist."
`S3.MethodNotAllowed`	"The specified method is not allowed against this resource。Modify the bucket’s policy to allow the correct Amazon S3 operation permissions."
`InternalError`	"An internal error occurred while attempting to deliver data。将重试交付；如果错误仍然存在，则会将其报告给以 AWS 寻求解决。”

错误代码	错误消息和信息
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."
S3.InvalidToken	"The provided token is malformed or otherwise invalid. Please check the credentials provided."
S3.KMS.KeyNotConfigured	"KMS key not configured. 配置您的 KMSMaster KeyID，或者禁用您的 S3 存储桶的加密。”
S3.KMS.AsymmetricCMKNotSupported	“Amazon S3 仅支持对称模式 CMKs。You cannot use an asymmetric CMK to encrypt your data in Amazon S3. 要获取 CMK 的类型，请使用 KMS DescribeKey 操作。”
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."

错误代码	错误消息和信息
`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."

Apache Iceberg 表数据传输错误

有关 Apache Iceberg 表数据传输错误，请参阅将数据传输到 Apache Iceberg 表。

Amazon Redshift 数据传输错误

Amazon Data Firehose 可以将以下与亚马逊 Redshift 相关的错误发送到日志。 CloudWatch

错误代码	错误消息和信息
`Redshift.TableNotFound`	"The table to which to load data was not found。Ensure that the specified table exists." The destination table in Amazon Redshift to which data should be copied from S3 was not found. Note that Amazon Data Firehose does not create the Amazon Redshift table if it does not exist.
`Redshift.SyntaxError`	"The COPY command contains a syntax error。Retry the command."
`Redshift.AuthenticationFailed`	"The provided user name and password failed authentication。Provide a valid user name and password."
`Redshift.AccessDenied`	"Access was denied。Ensure that the trust policy for the provided IAM role allows Amazon Data Firehose to assume the role."

错误代码	错误消息和信息
`Redshift.S3BucketAccessDenied`	"The COPY command was unable to access the S3 bucket。Ensure that the access policy for the provided IAM role allows access to the S3 bucket."
`Redshift.DataLoadFailed`	"Loading data into the table failed。Check STL_LOAD_ERRORS system table for details."
`Redshift.ColumnNotFound`	"A column in the COPY command does not exist in the table。Specify a valid column name."
`Redshift.DatabaseNotFound`	"The database specified in the Amazon Redshift destination configuration or JDBC URL was not found. Specify a valid database name."
`Redshift.IncorrectCopyOptions`	"Conflicting or redundant COPY options were provided。Some options are not compatible in certain combinations。Check the COPY command reference for more info." 有关更多信息，请参阅《Amazon Redshift 数据库开发人员指南》中的 Amazon Redshift COPY 命令。
`Redshift.MissingColumn`	"There is a column defined in the table schema as NOT NULL without a DEFAULT value and not included in the column list。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`	"The number of jsonpaths in the COPY command and the number of columns in the destination table should match。Retry the command."

错误代码	错误消息和信息
`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`	"The provided jsonpaths file is not in a supported JSON format。Retry the command."
`Redshift.MissingS3File`	"One or more S3 files required by Amazon Redshift have been removed from the S3 bucket. Check the S3 bucket policies to remove any automatic deletion of S3 files."
`Redshift.InsufficientPrivilege`	"The user does not have permissions to load data into the table。Check the Amazon Redshift user permissions for the INSERT privilege."
`Redshift.ReadOnlyCluster`	"The query cannot be executed because the system is in resize mode。Try the query again later."
`Redshift.DiskFull`	"Data could not be loaded because the disk is full。Increase the capacity of the Amazon Redshift cluster or delete unused data to free disk space."
`InternalError`	"An internal error occurred while attempting to deliver data。将重试交付；如果错误仍然存在，则会将其报告给以 AWS 寻求解决。"
`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."

错误代码	错误消息和信息
Redshift.SchemaNotFound	“找不到在 Amazon Redshift 目标配置中指定的架构。 DataTableName 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."
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	“复制失败，出现错误\” JSONPath 格式无效。Array index is out of range \". 请纠正这个 JSONPath 表达。”
Redshift.InvalidCopyOptionRBACAclNotAllowed	"COPY failed with error \"Cannot use RBAC acl framework while permission propagation is not enabled.\"

错误代码	错误消息和信息
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."
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."

错误代码	错误消息和信息
`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."
`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`	“ JSONPath 格式无效。”
`Redshift.TooManyConnectionsException`	"Too many connections to Redshift."
`Redshift.PSQLException`	“从 Redshift 中观察到PSQl异常。”

错误代码	错误消息和信息
`Redshift.DuplicateSecondsSpecification`	“ date/time 格式中重复的秒规范。”
`Redshift.RelationCouldNotBeOpened`	"Encountered Redshift error, relation could not be opened. Check Redshift logs for the specified DB."
`Redshift.TooManyClients`	"Encountered too many clients exception from Redshift. Revisit max connections to the database if there are multiple producers writing to it simultaneously."

Snowflake 数据传输错误

Firehose 可以将以下与 Snowflake 相关的错误发送到日志。 CloudWatch

错误代码	错误消息和信息
`Snowflake.InvalidUrl`	"Firehose is unable to connect to Snowflake. Please make sure that Account url is specified correctly in Snowflake destination configuration."
`Snowflake.InvalidUser`	"Firehose is unable to connect to Snowflake. Please make sure that User is specified correctly in Snowflake destination configuration."
`Snowflake.InvalidRole`	"The specified snowflake role does not exist or is not authorized. Please make sure that the role is granted to the user specified"
`Snowflake.InvalidTable`	"The supplied table does not exist or is not authorized"
`Snowflake.InvalidSchema`	"The supplied schema does not exist or is not authorized"

错误代码	错误消息和信息
`Snowflake.InvalidDatabase`	"The supplied database does not exist or is not authorized"
`Snowflake.InvalidPrivateKeyOrPassphrase`	"The specified private key or passphrase is not valid. Note that the private key provided should be a valid PEM RSA private key"
`Snowflake.MissingColumns`	"The insert request is rejected due to missing columns in input payload. Make sure that values are specified for all non-nullable columns"
`Snowflake.ExtraColumns`	"The insert request is rejected due to extra columns. Columns not present in table shouldn't be specified"
`Snowflake.InvalidInput`	"Delivery failed due to invalid input format. Make sure that the input payload provided is in the JSON format acceptable"
`Snowflake.IncorrectValue`	"Delivery failed due to incorrect data type in the input payload. Make sure that the JSON values specified in input payload adhere to the datatype declared in Snowflake table definition"

Splunk 数据传输错误

Amazon Data Firehose 可以将以下与 Splunk 相关的错误发送到日志。 CloudWatch

错误代码	错误消息和信息
`Splunk.ProxyWithoutStickySessions`	“如果您在 Amazon Data Firehose 和 HEC 节点之间有代理（ELB 或其他），则必须启用粘性会话才能支持 HEC。” ACKs
`Splunk.DisabledToken`	"The HEC token is disabled。Enable the token to allow data delivery to Splunk."

错误代码	错误消息和信息
`Splunk.InvalidToken`	"The HEC token is invalid。Update Amazon Data Firehose with a valid HEC token."
`Splunk.InvalidDataFormat`	"The data is not formatted correctly。To see how to properly format data for Raw or Event HEC endpoints, see Splunk Event Data."
`Splunk.InvalidIndex`	"The HEC token or input is configured with an invalid index。Check your index configuration and try again."
`Splunk.ServerError`	"Data delivery to Splunk failed due to a server error from the HEC node. Amazon Data Firehose will retry sending the data if the retry duration in your Amazon Data Firehose is greater than 0. If all the retries fail, Amazon Data Firehose backs up the data to Amazon S3."
`Splunk.DisabledAck`	"Indexer acknowledgement is disabled for the HEC token。Enable indexer acknowledgement and try again。For more info, see Enable indexer acknowledgement."
`Splunk.AckTimeout`	"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."
`Splunk.MaxRetriesFailed`	"Failed to deliver data to Splunk or to receive acknowledgment。Check your HEC health and try again."
`Splunk.ConnectionTimeout`	"The connection to Splunk timed out。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`	"Could not connect to the HEC endpoint。Make sure that the HEC endpoint URL is valid and reachable from Amazon Data Firehose."

错误代码	错误消息和信息
`Splunk.ConnectionClosed`	"Unable to send data to Splunk due to a connection failure。This might be a transient error。Increasing the retry duration in your Amazon Data Firehose configuration might guard against such transient failures."
`Splunk.SSLUnverified`	"Could not connect to the HEC endpoint。主机与对等项提供的证书不匹配。请确保证书和主机是有效的。
`Splunk.SSLHandshake`	"Could not connect to the HEC endpoint。请确保证书和主机是有效的。
`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 acknowledgements are enabled on HEC endpoint."
`Splunk.InvalidHecResponseCharacter`	"Invalid characters found in HEC response, make sure to check to the service and HEC configuration."

ElasticSearch 数据传输错误

Amazon Data Firehose 可能会将以下 ElasticSearch 错误发送到 CloudWatch 日志。

错误代码	错误消息和信息
ES.AccessDenied	"Access was denied。Ensure that the provided IAM role associated with firehose is not deleted."
ES.ResourceNotFound	“指定的 AWS Elasticsearch 域不存在。”

HTTPS 端点数据传输错误

Amazon Data Firehose 可以将以下与 HTTP 端点相关的错误发送到日志。 CloudWatch 如果这些错误均与您遇到的问题不匹配，则默认错误如下："An internal error occurred while attempting to deliver data. 将重试交付；如果错误仍然存在，则会将其报告给以 AWS 寻求解决。”

错误代码	错误消息和信息
HttpEndpoint.RequestTimeout	The delivery timed out before a response was received and will be retried. If this error persists, contact the AWS Firehose service team.
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."

错误代码	错误消息和信息
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 Troublesh ooting HTTP Endpoints in the Firehose documentation for more informati on. 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."

错误代码	错误消息和信息
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	“由于 URI 中的输入无效，HttpEndpoint 请求失败。Please make sure all the characters in the input URI are valid."

错误代码	错误消息和信息
`HttpEndpoint.MakeRequestFailure.IllegalCharacterInHeaderValue`	“由于非法响应错误，HttpEndpoint 请求失败。Illegal character '\n' in header value."
`HttpEndpoint.IllegalResponseFailure`	“由于非法响应错误，HttpEndpoint 请求失败。HTTP message must not contain more than one Content-Type header."
`HttpEndpoint.IllegalMessageStart`	“由于非法响应错误，HttpEndpoint 请求失败。Illegal HTTP message start. See Troubleshooting HTTP Endpoints in the Firehose documentation for more information."

Amazon OpenSearch 服务数据传送错误

对于 OpenSearch 服务目标，Amazon Data Firehose 会在服务返回错误时向 CloudWatch 日志发送错误。 OpenSearch

除了可能从 OpenSearch 集群返回的错误外，您还可能遇到以下两个错误：

- Authentication/authorization error occurs during attempt to deliver data to destination OpenSearch Service cluster. This can happen due to any permission issues and/or当你的 Amazon Data Firehose 目标 OpenSearch 服务域配置被修改时，会间歇性地发生。Please check the cluster policy and role permissions.
- 由于 authentication/authorization 故障，无法将数据传送到目标 OpenSearch 服务集群。这可能是由于修改您的 Amazon Data Firehose 目标 OpenSearch 服务域配置时 and/or 间歇性出现任何权限问题。Please check the cluster policy and role permissions.

错误代码	错误消息和信息
OS.AccessDenied	"Access was denied。确保所提供的 IAM 角色的信任策略允许 Firehose 担任该角色，并且访问策略允许访问亚马逊 OpenSearch 服务 API。”
OS.AccessDenied	"Access was denied。确保所提供的 IAM 角色的信任策略允许 Firehose 担任该角色，并且访问策略允许访问亚马逊 OpenSearch 服务 API。”
OS.AccessDenied	"Access was denied。Ensure that the provided IAM role associated with firehose is not deleted."
OS.AccessDenied	"Access was denied。Ensure that the provided IAM role associated with firehose is not deleted."
OS.ResourceNotFound	“指定的亚马逊 OpenSearch 服务域不存在。”
OS.ResourceNotFound	“指定的亚马逊 OpenSearch 服务域不存在。”
OS.AccessDenied	"Access was denied。确保所提供的 IAM 角色的信任策略允许 Firehose 担任该角色，并且访问策略允许访问亚马逊 OpenSearch 服务 API。”
OS.RequestTimeout	“对 Amazon Serv OpenSearch ice 集群或 OpenSearch 无服务器集合的请求超时。Ensure that the cluster or collection has sufficient capacity for the current workload."
OS.ClusterError	“Amazon S OpenSearch ervice 集群返回了一个未指明的错误。”
OS.RequestTimeout	“对 Amazon OpenSearch 服务集群的请求超时。Ensure that the cluster has sufficient capacity for the current workload."
OS.ConnectionFailed	“连接到 Amazon OpenSearch 服务集群或 OpenSearch 无服务器集合时出现问题。Ensure that the cluster or collection is healthy and reachable."
OS.ConnectionReset	“无法与亚马逊 OpenSearch 服务集群或 OpenSearch 无服务器集合保持连接。Contact the owner of the cluster or collection to resolve this issue."

错误代码	错误消息和信息
OS.ConnectionReset	“无法与 Amazon OpenSearch 服务集群或 OpenSearch 无服务器集合保持连接。Ensure that the cluster or collection is healthy and has sufficient capacity for the current workload."
OS.ConnectionReset	“无法与 Amazon OpenSearch 服务集群或 OpenSearch 无服务器集合保持连接。Ensure that the cluster or collection is healthy and has sufficient capacity for the current workload."
OS.AccessDenied	"Access was denied。确保 Amazon S OpenSearch ervice 集群上的访问策略授予对已配置的 IAM 角色的访问权限。”
OS.ValidationException	“集 OpenSearch 群返回了一个 ESService异常。原因之一是集群已升级到操作系统 2.x 或更高版本，但软管仍配置了 TypeName 参数。通过将设置为空字符串 TypeName 来更新软管配置，或者将端点更改为支持 Type 参数的集群。”
OS.ValidationException	"Member must satisfy regular expression pattern: [a-z][a-z0-9\\-]+
OS.JsonParseException	“亚马逊 OpenSearch 服务集群返回了 JsonParseException. Ensure that the data being put is valid."
OS.AmazonOpenSearchServiceParseException	“亚马逊 OpenSearch 服务集群返回了 AmazonOpenSearchServiceParseException. Ensure that the data being put is valid."
OS.ExplicitIndexInBulkNotAllowed	“确保亚马逊服务集群上的 rest.action.multi.allow_explicit_index 设置为 true。” OpenSearch
OS.ClusterError	“Amazon Serv OpenSearch ice 集群或 OpenSearch 无服务器集合返回了一个未指明的错误。”
OS.ClusterBlockException	“集群返回了 ClusterBlockException. It may be overloaded."

错误代码	错误消息和信息
OS.InvalidARN	“提供的亚马逊 OpenSearch 服务 ARN 无效。请检查您的 DeliveryStream 配置。”
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. 将重试交付；如果错误仍然存在，则会将其报告给以 AWS 寻求解决。”
OS.AliasWithMultipleIndicesNotAllowed	"Alias has more than one indices associated with it. Ensure that the alias has only one index associated with it."
OS.UnsupportedVersion	“亚马逊 Data Firehose 目前不支持亚马逊 OpenSearch 服务 6.0。如需更多信息，请联系 AWS Support。”
OS.CharConversionException	"One or more records contained an invalid character."
OS.InvalidDomainNameLength	"The domain name length is not within valid OS limits."
OS.VPCDomainNotSupported	“目前不支持 VPCs 其中的亚马逊 OpenSearch 服务域。”
OS.ConnectionError	“http 服务器意外关闭了连接，请验证亚马逊 OpenSearch 服务集群或 OpenSearch 无服务器集合的运行状况。”
OS.LargeFieldData	“Amazon S OpenSearch ervice 集群中止了请求，因为它包含的字段数据大于允许值。”
OS.BadGateway	“Amazon Serv OpenSearch ice 集群或 OpenSearch 无服务器集合中止了请求，结果是：502 Bad Gateway。”

错误代码	错误消息和信息
`OS.ServiceException`	“从 Amazon OpenSearch 服务集群或 OpenSearch 无服务器集合收到错误。If the cluster or collection is behind a VPC, ensure network configuration allows connectivity."
`OS.GatewayTimeout`	“Firehose 在连接到亚马逊 OpenSearch 服务集群或 OpenSearch 无服务器集合时遇到了超时错误。”
`OS.MalformedData`	“亚马逊数据 Firehose 不支持 Firehose 记录中的亚马逊 OpenSearch 服务批量 API 命令。”
`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."

Lambda 调用错误

Amazon Data Firehose 可以将以下 Lambda 调用错误发送到日志。 CloudWatch

错误代码	错误消息和信息
`Lambda.AssumeRoleAccessDenied`	"Access was denied。Ensure that the trust policy for the provided IAM role allows Amazon Data Firehose to assume the role."
`Lambda.InvokeAccessDenied`	"Access was denied。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." 有关更多信息，请参阅 数据转换所需的参数。

错误代码	错误消息和信息
`Lambda.InvokeLimitExceeded`	"The Lambda concurrent execution limit is exceeded. Increase the concurrent execution limit." 有关更多信息，请参阅《AWS Lambda 开发人员指南》中的 AWS Lambda 限制。
`Lambda.DuplicatedRecordId`	"Multiple records were returned with the same record ID。确保 Lambda 函数 IDs 为每条记录返回唯一的记录。” 有关更多信息，请参阅 数据转换所需的参数。
`Lambda.MissingRecordId`	“一条或多条记录 IDs 没有返回。确保 Lambda 函数返回所有收到的记录 IDs。” 有关更多信息，请参阅 数据转换所需的参数。
`Lambda.ResourceNotFound`	"The specified Lambda function does not exist. Use a different function that does exist."
`Lambda.InvalidSubnetIDException`	"The specified subnet ID in the Lambda function VPC configuration is invalid. Ensure that the subnet ID is valid."
`Lambda.InvalidSecurityGroupIDException`	"The specified security group ID in the Lambda function VPC configuration is invalid. Ensure that the security group ID is valid."
`Lambda.SubnetIPAddressLimitReachedException`	“无法AWS Lambda 为 Lambda 函数设置 VPC 访问权限，因为一个或多个已配置的子网没有可用的 IP 地址。Increase the IP address limit." 有关更多信息，请参阅《Amazon VPC 用户指南》中的 Amazon VPC 限制：VPC 和子网。

错误代码	错误消息和信息
`Lambda.ENILimitReachedException`	“AWS Lambda 由于已达到网络接口的限制，无法在 Lambda 函数配置中指定的 VPC 中创建弹性网络接口 (ENI)。Increase the network interface limit." 有关更多信息，请参阅《Amazon VPC 用户指南》中的 Amazon VPC 限制：网络接口。
`Lambda.FunctionTimedOut`	The Lambda function invocation timed out. Increase the Timeout setting in the Lambda function. 有关更多信息，请参阅配置函数超时。
`Lambda.FunctionError`	This can be due to any of the following errors: • Invalid output structure. Check your function and make sure the output is in the required format. Also, make sure the processed records contain a valid result status of `Dropped`, `Ok`, or `ProcessingFailed` . • The Lambda function was successfully invoked but it returned an error result. • Lambda 无法解密环境变量，因为对 KMS 密钥的访问已被拒绝。Check the function's KMS key settings as well as the key policy. 有关更多信息，请参阅密钥访问故障排除。
`Lambda.FunctionRequestTimedOut`	Amazon Data Firehose encountered Request did not complete before the request timeout configuration error when invoking Lambda. Revisit the Lambda code to check if the Lambda code is meant to run beyond the configured timeout. If so, consider tuning Lambda configuration settings, including memory, timeout. 有关更多信息，请参阅配置 Lambda 函数选项。
`Lambda.TargetServerFailedToRespond`	Amazon Data Firehose encountered an error. 目标服务器在调用 AWS Lambda 服务时未能响应错误。

错误代码	错误消息和信息
`Lambda.InvalidZipFileException`	Amazon Data Firehose InvalidZipFileException 在调用 Lambda 函数时遇到了问题。Check your Lambda function configuration settings and the Lambda code zip file.
`Lambda.InternalServerError`	“Firehose 在调用 Lambda 服务 InternalServerError时遇到了亚马逊 D AWS ata Firehose。Amazon Data Firehose will retry sending data a fixed number of times. 您可以使用或指定或覆盖重试选项。`CreateDeliveryStream` `UpdateDestination` APIs如果错误仍然存在，请联系 AWS Lambda 支持团队。
`Lambda.ServiceUnavailable`	Amazon Data Firehose ServiceUnavailableException 在调用 Lambda AWS 服务时遇到了问题。Amazon Data Firehose will retry sending data a fixed number of times. 您可以使用或指定或覆盖重试选项。`CreateDeliveryStream` `UpdateDestination` APIs如果错误仍然存在，请联系 AWS Lambda 支持人员。
`Lambda.InvalidSecurityToken`	Cannot invoke Lambda function due to invalid security token. Cross partition Lambda invocation is not supported.
`Lambda.InvocationFailure`	This can be due to any of the following errors: • Amazon Data Firehose 在调用 Lambda AWS 时遇到了错误。The operation will be retried; if the error persists, it will be reported to AWS for resolution." • Amazon Data Firehose 遇到了来自 KMSInvalid StateException Lambda 的。Lambda 无法解密环境变量，因为所用的 KMS 密钥处于无效的解密状态。Check the lambda function's KMS key. • Amazon Data Firehose 遇到了来自 AWS LambdaException Lambda 的。Lambda was unable to initialize the provided container image. Verify the image. • Amazon Data Firehose 在调用 Lambda AWS 时遇到了超时错误。支持的最大函数超时为 5 分钟。有关更多信息，请参阅数据转换执行时间。

错误代码	错误消息和信息
`Lambda.JsonMappingException`	There was an error parsing returned records from the Lambda function. Ensure that data field is base-64 encoded.

Kinesis 调用错误

Amazon Data Firehose 可以将以下 Kinesis 调用错误发送到日志。 CloudWatch

错误代码	错误消息和信息
`Kinesis.AccessDenied`	"Access was denied when calling Kinesis. 确保所使用的 IAM 角色的访问策略允许访问相应的 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. 如果 Firehose 是 DirectPut 这种类型，那么 Firehose 可能已经不存在了。”
`Kinesis.SubscriptionRequired`	"Access was denied when calling Kinesis. 确保为访问 Kinesis 直播而传递的 IAM 角色已订阅 Kinesis AWS 。”
`Kinesis.Throttling`	"Throttling error encountered when calling Kinesis. 这可能是由于其他应用程序调用了与 Firehose 直播 APIs 相同的内容，或者是因为你创建了太多 Firehose 直播与源相同的 Kinesis 流。”
`Kinesis.Throttling`	"Throttling error encountered when calling Kinesis. 这可能是由于其他应用程序调用了与 Firehose 直播 APIs 相同的内容，或者是因为你创建了太多 Firehose 直播与源相同的 Kinesis 流。”
`Kinesis.AccessDenied`	"Access was denied when calling Kinesis. 确保所使用的 IAM 角色的访问策略允许访问相应的 Kinesis APIs。”
`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."

错误代码	错误消息和信息
`Kinesis.KMS.AccessDeniedException`	“Firehose 无法访问用于 K encrypt/decrypt inesis Stream 的 KMS 密钥。Please grant the Firehose delivery role access to the key."
`Kinesis.KMS.KeyDisabled`	“Firehose 无法从源 Kinesis Stream 读取数据，因为用于它的 KMS 密钥已被禁用。 encrypt/decrypt 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."

Kinesis 调 DirectPut 用错误

Amazon Data Firehose 可以将以下 Kinesis DirectPut 调用错误发送到日志。 CloudWatch

错误代码	错误消息和信息
`Firehose.KMS.AccessDeniedException`	"Firehose does not have access to the KMS Key. Please check the key policy."
`Firehose.KMS.InvalidStateException`	"Firehose is unable to decrypt the data because the KMS key used to encrypt it is in an invalid state."

错误代码	错误消息和信息
`Firehose.KMS.NotFoundException`	"Firehose is unable to decrypt the data because the KMS key used to encrypt it was not found."
`Firehose.KMS.KeyDisabled`	"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."

AWS Glue 调用错误

Amazon Data Firehose 可以将以下 AWS Glue 调用错误发送到日志。 CloudWatch

错误代码	错误消息和信息
`DataFormatConversion.InvalidSchema`	"The schema is invalid."
`DataFormatConversion.EntityNotFound`	“找 table/database 不到指定的。请确保 table/database 存在且架构配置中提供的值正确，尤其是在大小写方面。”
`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."
`DataFormatConversi`	"Could not find a matching schema from glue. Please make sure the catalogId provided is valid."

错误代码	错误消息和信息
on.Invali dInput	
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."
DataForma tConversi on.Invali dGlueRole	"Invalid role. Please ensure that the role specified in the data format conversion configuration exists."

错误代码	错误消息和信息
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 在您指定的区域尚不可用；请指定其他区域。”
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. 如果您有大量 Glue 表版本，请添加 “glue:GetTableVersion” 权限（推荐）或删除未使用的表格版本。如果您在 Glue 中没有大量表格，请联系 Su AWS pport。”
DataFirehose.InternalError	"Timed out while retrieving table from Glue. 如果您有大量 Glue 表版本，请添加 “glue:GetTableVersion” 权限（推荐）或删除未使用的表格版本。如果您在 Glue 中没有大量表格，请联系 Su AWS pport。”
DataFormatConversion.GlueEncryptionException	"There was an error retrieving the master key. Ensure that the key exists and state is correct."

DataFormatConversion 调用错误

Amazon Data Firehose 可以将以下 DataFormatConversion 调用错误发送到日志。 CloudWatch

错误代码	错误消息和信息
`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."
`DataFormatConversion.MalformedData`	"Length of json key must not be greater than 262144"
`DataFormatConversion.MalformedData`	"The data cannot be decoded as UTF-8."
`DataFormatConversi`	"Illegal character found between tokens."

错误代码	错误消息和信息
on.Malfor medData	
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."
DataForma tConversi on.Malfor medData	"The input JSON contained a primitive at the top level. The top level must be an object or array."
DataForma tConversi on.Malfor medData	"The record was empty or contained only whitespace."

错误代码	错误消息和信息
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."
DataFormatConversion.ConversionFailureException	"ConversionFailureException"

错误代码	错误消息和信息
DataFormatConversion.DataFormatConversionCustomerErrorException	"DataFormatConversionCustomerErrorException"
DataFormatConversion.DataFormatConversionCustomerErrorException	"DataFormatConversionCustomerErrorException"
DataFormatConversion.MalformedData	"Data does not match the schema."
DataFormatConversion.InvalidSchema	"The schema is invalid."
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."

错误代码	错误消息和信息
DataFormatConversion.EntityNotFound	“找 table/database 不到指定的。请确保 table/database 存在且架构配置中提供的值正确，尤其是在大小写方面。”
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."
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."

错误代码	错误消息和信息
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 在您指定的区域尚不可用；请指定其他区域。”
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. 如果您有大量 Glue 表版本，请添加 “glue:GetTableVersion” 权限（推荐）或删除未使用的表格版本。如果您在 Glue 中没有大量表格，请联系 Su AWS pport。”
DataFirehose.InternalError	"Timed out while retrieving table from Glue. 如果您有大量 Glue 表版本，请添加 “glue:GetTableVersion” 权限（推荐）或删除未使用的表格版本。如果您在 Glue 中没有大量表格，请联系 Su AWS pport。”

错误代码	错误消息和信息
`DataFormatConversion.MalformedData`	"One or more fields have incorrect format."

Amazon Data Firehose 的访问 CloudWatch 日志

您可以使用亚马逊数据 Firehose 控制台或控制台查看与亚马逊 Data Firehose 数据传输失败相关的错误日志。 CloudWatch 下面的过程介绍如何使用这两种方法访问错误日志。

要使用 Amazon Data Firehose 控制台访问错误日志

1. 登录 AWS 管理控制台 并在 /firehose 上打开 Firehose 控制台 https://console.aws.amazon.com
2. 在导航栏上，选择一个 AWS 区域。
3. 选择 Firehose 流名称以转到 Firehose 流详细信息页面。
4. 选择 Error Log 查看与数据传输故障有关的错误日志的列表。

使用 CloudWatch 控制台访问错误日志

1. 打开 CloudWatch 控制台，网址为https://console.aws.amazon.com/cloudwatch/。
2. 在导航栏中，选择一个区域。
3. 在导航窗格中，选择日志。
4. 选择日志组和日志流，查看与数据传输故障有关的错误日志的列表。

监控 Kinesis 代理运行状况

Kinesis 代理发布命名空间为的自定义 CloudWatch 指标。AWS KinesisAgent该代理可帮助您评测代理运行是否正常、是否按指定方式将数据提交到 Amazon Data Firehose，以及在数据创建器上是否使用适当数量的 CPU 和内存资源。

记录数和发送的字节数等指标对于了解代理将数据提交到 Firehose 流的速率非常有用。当这些指标低于预期阈值一定的百分比或者降低为零时，可能表明存在配置问题、网络错误或代理运行状况问题。诸

如主机上的 CPU 和内存消耗以及代理错误计数器等指标可用于指示数据创建器资源使用情况，并提供对潜在的配置或主机错误的深入分析。最后，代理还会记录服务异常，以帮助调查代理问题。

代理指标在代理配置设置 `cloudwatch.endpoint` 指定的区域中报告。有关更多信息，请参阅 指定代理配置设置。

从多个 Kinesis 代理发布的 Cloudwatch 指标是聚合或合并的。

对于默认启用的 Kinesis 代理发出的指标，会收取一定的费用。有关更多信息，请参阅 Amazon CloudWatch 定价。

使用监视器 CloudWatch

Kinesis Agent 将以下指标发送到。 CloudWatch

指标	说明
`BytesSent`	在指定时段内发送到 Firehose 流的字节数。 单位：字节
`RecordSendAttempts`	在指定的时间范围内对 `PutRecordBatch` 的一次调用中尝试的记录数（第一次，或者作为重试）。 单位：计数
`RecordSendErrors`	在指定时间范围内对 `PutRecordBatch` 的一次调用中返回故障状态的记录数，包括重试。 单位：计数
`ServiceErrors`	在指定时间范围内产生服务错误（限制错误之外的其他错误）的 `PutRecordBatch` 调用次数。 单位：计数

使用记录亚马逊 Data Firehose API 调用 AWS CloudTrail

Amazon Data Firehose 与一项服务集成，可记录用户 AWS CloudTrail、角色或 AWS 服务在 Amazon Data Firehose 中执行的操作。 CloudTrail 将 Amazon Data Firehose 的所有 API 调用捕获为事件。捕

获的调用包括来自 Amazon Data Firehose 控制台的调用，以及对 Amazon Data Firehose API 操作的代码调用。如果您创建了跟踪，则可以允许将 CloudTrail 事件持续传输到 Amazon S3 存储桶，包括针对 Amazon Data Firehose 的事件。如果您未配置跟踪，您仍然可以在 CloudTrail 控制台的 “事件历史记录” 中查看最新的事件。通过收集的信息 CloudTrail，您可以确定向 Amazon Data Firehose 发出的请求、发出请求的 IP 地址、谁提出了请求、何时提出请求以及其他详细信息。

要了解更多信息 CloudTrail，包括如何配置和启用它，请参阅《AWS CloudTrail 用户指南》。

Firehose 信息在 CloudTrail

CloudTrail 在您创建 AWS 账户时已在您的账户上启用。当 Amazon Data Firehose 中出现支持的事件活动时，该活动会与其他 AWS 服务 CloudTrail事件一起记录在事件历史记录中。您可以在 AWS 账户中查看、搜索和下载最新事件。有关更多信息，请参阅使用 CloudTrail 事件历史记录查看事件。

要持续记录您的 AWS 账户中的事件，包括 Amazon Data Firehose 的事件，请创建跟踪。跟踪允许 CloudTrail 将日志文件传输到 Amazon S3 存储桶。默认情况下，当您在控制台中创建跟踪时，该跟踪将应用于所有 AWS 区域。跟踪记录 AWS 分区中所有区域的事件，并将日志文件传送到您指定的 Amazon S3 存储桶。此外，您可以配置其他 AWS 服务，以进一步分析和处理 CloudTrail 日志中收集的事件数据。有关更多信息，请参阅下列内容：

- 创建跟踪概述
- CloudTrail 支持的服务和集成
- 配置 Amazon SNS 通知 CloudTrail
- 接收来自多个区域的 CloudTrail 日志文件和接收来自多个账户的 CloudTrail 日志文件

Amazon Data Firehose 支持将以下操作作为事件记录在 CloudTrail 日志文件中：

- CreateDeliveryStream
- DeleteDeliveryStream
- DescribeDeliveryStream
- ListDeliveryStreams
- ListTagsForDeliveryStream
- TagDeliveryStream
- StartDeliveryStreamEncryption
- StopDeliveryStreamEncryption
- UntagDeliveryStream

- UpdateDestination

每个事件或日志条目都包含有关生成请求的人员信息。身份信息有助于您确定以下内容：

- 请求是使用根证书还是 AWS Identity and Access Management (IAM) 用户凭证发出。
- 请求是使用角色还是联合用户的临时安全凭证发出的。
- 请求是否由其他 AWS 服务发出。

有关更多信息，请参阅 CloudTrail userIdentity 元素。

示例：Firehose 日志文件条目

跟踪是一种配置，允许将事件作为日志文件传输到您指定的 Amazon S3 存储桶。 CloudTrail 日志文件包含一个或多个日志条目。一个事件表示来自任何源的一个请求，包括有关所请求的操作、操作的日期和时间、请求参数等方面的信息。 CloudTrail 日志文件不是公用 API 调用的有序堆栈跟踪，因此它们不会以任何特定顺序显示。

以下示例显示了一个演示CreateDeliveryStream、、DescribeDeliveryStreamListDeliveryStreamsUpdateDestination和DeleteDeliveryStream操作的 CloudTrail 日志条目。

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

使用 Firehose 的代码示例 AWS SDKs

以下代码示例展示了如何将 Firehose 与 AWS 软件开发套件 (SDK) 配合使用。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

场景是向您展示如何通过在一个服务中调用多个函数或与其他 AWS 服务服务结合来完成特定任务的代码示例。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅将 Firehose 与 SDK 配合使用 AWS。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

代码示例

- 使用 Firehose 的基本示例 AWS SDKs
  - 使用 Firehose 的操作 AWS SDKs
    - PutRecord与 AWS SDK 或 CLI 配合使用
    - PutRecordBatch与 AWS SDK 或 CLI 配合使用
- 使用 Firehose 的场景 AWS SDKs
  - 使用 Amazon Data Firehose 处理单个记录和批量记录

使用 Firehose 的基本示例 AWS SDKs

以下代码示例展示了如何使用 Amazon Data Firehose 的基础知识。 AWS SDKs

示例

- 使用 Firehose 的操作 AWS SDKs
  - PutRecord与 AWS SDK 或 CLI 配合使用
  - PutRecordBatch与 AWS SDK 或 CLI 配合使用

使用 Firehose 的操作 AWS SDKs

以下代码示例演示了如何使用执行各个 Firehose 操作。 AWS SDKs每个示例都包含一个指向的链接 GitHub，您可以在其中找到有关设置和运行代码的说明。

这些代码节选调用了 Firehose API，是必须在上下文中运行的大型程序的代码节选。您可以在使用 Firehose 的场景 AWS SDKs中结合上下文查看操作。

以下示例仅包括最常用的操作。有关完整列表，请参阅 Amazon Data Firehose API 参考。

示例

- PutRecord与 AWS SDK 或 CLI 配合使用
- PutRecordBatch与 AWS SDK 或 CLI 配合使用

PutRecord与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 PutRecord。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 将记录放入 Firehose

CLI

AWS CLI

要将记录写入流

以下 put-record 示例将数据写入流中。数据以 Base64 格式编码。

```
aws firehose put-record \
    --delivery-stream-name my-stream \
    --record '{"Data":"SGVsbG8gd29ybGQ="}'
```

输出：

```
{
    "RecordId": "RjB5K/nnoGFHqwTsZlNd/
TTqvjE8V5dsyXZTQn2JXrdpMTOwssyEb6nfC8fwf1whhwnItt4mvrn+gsqeK5jB7QjuLg283+Ps4Sz/
j1Xujv31iDhnPdaLw4BOyM9Amv7PcCuB2079RuM0NhoakbyUymlwY8yt20G8X2420wu1jlFafhci4erAt7QhDEvpw
    "Encrypted": false
}
```

有关更多信息，请参阅《Amazon Kinesis Data Firehose 开发人员指南》中的将数据发送到 Amazon Kinesis Data Firehose 传输流。

- 有关 API 的详细信息，请参阅AWS CLI 命令参考PutRecord中的。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 还有更多相关信息 GitHub。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
   /**
    * 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);
        }
    }
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考PutRecord中的。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 还有更多相关信息 GitHub。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
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
```

- 有关 API 的详细信息，请参阅适用PutRecord于 Python 的AWS SDK (Boto3) API 参考。

SAP ABAP

适用于 SAP ABAP 的 SDK

> **Note**
>
> 还有更多相关信息 GitHub。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
TRY.
    DATA(lo_record) = NEW /aws1/cl_frhrecord( iv_data = iv_data ).

    DATA(lo_result) = lo_frh->putrecord(
      iv_deliverystreamname = iv_deliv_stream_name
      io_record             = lo_record ).

    MESSAGE 'Record sent to Firehose delivery stream.' TYPE 'I'.
  CATCH /aws1/cx_frhresourcenotfoundex.
    MESSAGE 'Delivery stream not found.' TYPE 'E'.
  CATCH /aws1/cx_frhinvalidargumentex.
    MESSAGE 'Invalid argument provided.' TYPE 'E'.
  CATCH /aws1/cx_frhserviceunavailex.
    MESSAGE 'Service temporarily unavailable.' TYPE 'E'.
ENDTRY.
```

- 有关 API 的详细信息，请参阅适用PutRecord于 S AP 的AWS SDK ABAP API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅将 Firehose 与 SDK 配合使用 AWS。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

`PutRecordBatch`与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 `PutRecordBatch`。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 将记录放入 Firehose

CLI

AWS CLI

将多条记录写入流中

以下 put-record-batch 示例将三条记录写入流中。数据以 Base64 格式编码。

```
aws firehose put-record-batch \
    --delivery-stream-name my-stream \
    --records file://records.json
```

myfile.json 的内容：

```
[
    {"Data": "Rmlyc3QgdGhpbmc="},
    {"Data": "U2Vjb25kIHRoaW5n"},
    {"Data": "VGhpcmQgdGhpbmc="}
]
```

输出：

```
{
    "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"
        }
```

```
    ]
}
```

有关更多信息，请参阅《Amazon Kinesis Data Firehose 开发人员指南》中的将数据发送到 Amazon Kinesis Data Firehose 传输流。

- 有关 API 的详细信息，请参阅AWS CLI 命令参考PutRecordBatch中的。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 还有更多相关信息 GitHub。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
   /**
    * 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);
        }
    }
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[PutRecordBatch](#)中的。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 还有更多相关信息 GitHub。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
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}")
```

- 有关 API 的详细信息，请参阅适用PutRecordBatch于 Python 的AWS SDK (Boto3) API 参考。

Rust

适用于 Rust 的 SDK

> **Note**
>
> 还有更多相关信息 GitHub。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
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
}
```

- 有关 API 的详细信息，请参阅适用PutRecordBatch于 Rust 的AWS SDK API 参考。

SAP ABAP

适用于 SAP ABAP 的 SDK

> **Note**
>
> 还有更多相关信息 GitHub。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    TRY.
        DATA(lo_result) = lo_frh->putrecordbatch(
          iv_deliverystreamname = iv_deliv_stream_name
          it_records            = it_records ).

        DATA(lv_failed_count) = lo_result->get_failedputcount( ).

        IF lv_failed_count > 0.
          MESSAGE |{ lv_failed_count } records failed to send.| TYPE 'I'.
        ELSE.
          MESSAGE 'All records sent successfully to Firehose delivery stream.'
TYPE 'I'.
        ENDIF.
      CATCH /aws1/cx_frhresourcenotfoundex.
        MESSAGE 'Delivery stream not found.' TYPE 'E'.
      CATCH /aws1/cx_frhinvalidargumentex.
        MESSAGE 'Invalid argument provided.' TYPE 'E'.
```

```
      CATCH /aws1/cx_frhserviceunavailex.
        MESSAGE 'Service temporarily unavailable.' TYPE 'E'.
    ENDTRY.
```

- 有关 API 的详细信息，请参阅适用PutRecordBatch于 S AP 的AWS SDK ABAP API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅将 Firehose 与 SDK 配合使用 AWS。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

使用 Firehose 的场景 AWS SDKs

以下代码示例向您展示了如何使用在 Firehose 中实现常见场景。 AWS SDKs这些场景向您展示了如何通过调用 Firehose 中的多个函数或与其他 AWS 服务结合来完成特定任务。每个场景都包含完整源代码的链接，您可以在其中找到有关如何设置和运行代码的说明。

场景以中等水平的经验为目标，可帮助您结合具体环境了解服务操作。

示例

- 使用 Amazon Data Firehose 处理单个记录和批量记录

使用 Amazon Data Firehose 处理单个记录和批量记录

以下代码示例演示如何使用 Firehose 处理单个记录和批量记录。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 还有更多相关信息 GitHub。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

此示例将单个记录和批量记录放入 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());
        }
    }
}
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API Reference》中的以下主题。
  - PutRecord
  - PutRecordBatch

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 还有更多相关信息 GitHub。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

此脚本将单个记录和批量记录放入 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()
```

此文件包含上述脚本的配置。

```
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()
```

- 有关 API 详细信息，请参阅《AWS SDK for Python (Boto3) API Reference》中的以下主题。
  - PutRecord
  - PutRecordBatch

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅将 Firehose 与 SDK 配合使用 AWS。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

对 Amazon Data Firehose 中的错误进行问题排查

如果在传输或处理数据时 Firehose 遇到错误，则它会一直重试，直至超过配置的重试持续时间。如果重试持续时间在成功传输数据之前结束，则 Firehose 会将数据备份到配置的 S3 备份存储桶。如果目的地是 Amazon S3，并且传输失败或传输到备份 S3 存储桶失败，Firehose 会不断重试，直到保留期结束。

有关使用追踪配送错误的信息 CloudWatch，请参阅the section called “使用 CloudWatch 日志进行监控”。

Direct PUT

对于 `DirectPut` Firehose 流，Firehose 会将记录保留 24 小时。对于其数据来源为 Kinesis 数据流的 Firehose 流，您可以按照更改数据留存期中的说明更改保留期。在这种情况下，Firehose 会无限地重试以下操作：`DescribeStream`、`GetRecords` 和 `GetShardIterator`。

如果 Firehose 流使用 `DirectPut`，请检查 `IncomingBytes` 和 `IncomingRecords` 指标，查看是否有传入流量。如果您正在使用 `PutRecord` 或 `PutRecordBatch`，请务必捕获异常并重试。我们建议使用带指数退避的重试策略，并且提供抖动和多次重试功能。此外，如果您使用 `PutRecordBatch` API，请确保您的代码检查响应FailedPutCount中的值，即使 API 调用成功也是如此。

Kinesis Data Stream

如果 Firehose 流使用 Kinesis 数据流作为源，请检查源数据流的 `IncomingBytes` 和 `IncomingRecords` 指标。此外，请确保为 Firehose 流发出 `DataReadFromKinesisStream.Bytes` 和 `DataReadFromKinesisStream.Records` 指标。

常见问题

以下是一些问题排查技巧，可帮助您解决使用 Firehose 流时遇到的常见问题。

Firehose 流不可用

Firehose 流不能用作 CloudWatch 日志、 CloudWatch 事件或物 AWS 联网操作的目标，因为某些 AWS 服务只能向相同的 Firehose 流发送消息和事件。 AWS 区域验证您的 Firehose 流是否与其他服务位于同一区域。

目的地没有数据

如果没有数据摄取问题，并且为 Firehose 流发出的指标看起来不错，但您在目的地中看不到数据，请检查读取器逻辑。确保您的读取器正确解析所有数据。

数据新鲜度指标增长或未发出数据新鲜度指标

数据新鲜度用于衡量您的数据在 Firehose 流中的最新程度。这是指 Firehose 流中最早数据记录的期限，从 Firehose 摄取数据到现在的时间计算。Firehose 提供了可用于监控数据新鲜度的指标。要确定指定目标的数据新鲜度指标，请参阅the section called “使用 CloudWatch 指标进行监控”。

如果您为所有事件或所有文档启用备份，请监控两个单独的数据新鲜度指标：一个用于主目标，另一个用于备份。

如果未发出数据新鲜度指标，则意味着 Firehose 流没有有效的传输。当完全阻止数据传输或没有任何传入数据时，会发生这种情况。

如果数据新鲜度指标在不断增加，则意味着数据传输落后。这可能是由于以下原因之一。

- 目标无法跟上传输速率。如果由于高流量而导致 Firehose 临时遇到错误，则传输可能会落后。这可能发生在亚马逊 S3 以外的目的地（ OpenSearch服务、Amazon Redshift 或 Splunk 也可能发生这种情况）。确保您的目标有足够的容量来处理传入流量。
- 目标非常慢。如果 Firehose 遇到高延迟，则数据传输可能会落后。监控目标的延迟指标。
- Lambda 函数运行缓慢。这可能导致数据传输速率要低于 Firehose 流的数据摄取速率。如果可能，请提高 Lambda 函数的效率。例如，如果函数处理网络 IO，请使用多个线程或异步 IO 来增加并行度。此外，考虑增加 Lambda 函数的内存大小，以便相应地增加 CPU 分配。这可能会加快 Lambda 的调用速度。有关配置 Lambda 函数的信息，请参阅配置 Lam bda AWS 函数。
- 在数据传输过程中出现故障。有关如何使用 Amazon Logs 监控错误 CloudWatch 的信息，请参阅the section called “使用 CloudWatch 日志进行监控”。
- 如果 Firehose 流的数据来源是 Kinesis 数据流，则可能会存在限制情况。检查 `ThrottledGetRecords`、`ThrottledGetShardIterator` 和 `ThrottledDescribeStream` 指标。如果有多个使用者附加到 Kinesis 数据流，请考虑以下事项：
  - 如果 `ThrottledGetRecords` 和 `ThrottledGetShardIterator` 指标非常高，我们建议您增加为数据流预配置的分片数。
  - 如果`ThrottledDescribeStream`为高，我们建议您为中配置的角色添加`kinesis:listshards`权限KinesisStreamSourceConfiguration。
- 目标缓冲提示较低。这可能会增加 Firehose 传输到目的地所需的往返次数，从而可能导致传输滞后。考虑增大缓冲提示的值。有关更多信息，请参阅 BufferingHints。

- 当错误频繁发生时，较长的重试持续时间可能会导致传输落后。考虑减小重试持续时间。此外，监控错误并尝试减少这些错误。有关如何使用 Amazon Logs 监控错误 CloudWatch 的信息，请参阅the section called “使用 CloudWatch 日志进行监控”。
- 如果目标为 Splunk 并且 `DeliveryToSplunk.DataFreshness` 非常高，但 `DeliveryToSplunk.Success` 看起来不错，则 Splunk 集群可能很忙。如果可能，请释放 Splunk 集群。或者，请联系 AWS Support，请求增加 Firehose 用来与 Splunk 集群通信的通道数量。

记录格式转换为 Apache Parquet 失败

如果您获取包含`Set`该类型的 DynamoDB 数据，通过 Lambda 将其流式传输到 Fire AWS Glue Data Catalog hose 流，然后使用将记录格式转换为 Apache Parquet，就会发生这种情况。

当 AWS Glue Crawler 为 DynamoDB 集数据类型`StringSet`（`NumberSet`、`BinarySet`和）编制索引时，它会将它们分别存储在数据目录中`SET<STRING>`，即`SET<BIGINT>`、和。`SET<BINARY>`但对于 Firehose，要将数据记录转换为 Apache Parquet 格式，需要 Apache Hive 数据类型。由于集类型并非有效的 Apache Hive 数据类型，所以转换会失败。要进行成功转换，请使用 Apache Hive 数据类型更新数据目录。为此，您可以将数据目录中的 `set` 更改为 `array`。

在中将一种或多种数据类型从**##**更改为**##** AWS Glue 数据目录

1. 登录 AWS 管理控制台 并打开 AWS Glue 控制台，网址为https://console.aws.amazon.com/glue/。
2. 在左侧窗格中的数据目录标题下，选择表。
3. 在表列表中，选择您需要修改一个或多个数据类型的表的名称。这会将您引导至该表的详细信息页面。
4. 选择详细信息页面右上角的编辑架构按钮。
5. 在数据类型列中，选择第一个 `set` 数据类型。
6. 在列类型下拉列表中，将该类型从 `set` 更改为 `array`。
7. 根据您的ArraySchema场景的相应数据类型`array<binary>`，在该字段中输`array<string>`入、或。`array<int>`
8. 选择更新。
9. 重复之前的步骤，将其他 `set` 类型转换为 `array` 类型。
10. 选择保存。

Lambda 的已转换对象缺少字段

当您使用 Lambda 数据转换将 JSON 数据更改为 Parquet 对象时，某些字段可能会在转换后缺失。如果您的 JSON 对象有大写字母并且区分大小写设置为 `false`，则会发生这种情况，这可能会导致数据转换后的 JSON 密钥不匹配，从而导致 S3 存储桶中生成的 Parquet 对象中缺少数据。

要解决此问题，请确保将 hose 配置将 `deserializationOption: case.insensitive` 设置为 `true`，以便 JSON 密钥在转换后匹配。

Amazon S3 故障排除

如果数据未传输到 Amazon Simple Storage Service（Amazon S3）存储桶，请检查以下各项。

- 检查 Firehose `IncomingBytes` 和 `IncomingRecords` 指标，确保数据成功发送到您的 Firehose 流。有关更多信息，请参阅 使用指标监控亚马逊数据 Firehose CloudWatch。
- 如果启用了使用 Lambda 进行的数据转换，请检查 Firehose `ExecuteProcessingSuccess` 指标，确保 Firehose 已尝试调用 Lambda 函数。有关更多信息，请参阅 使用指标监控亚马逊数据 Firehose CloudWatch。
- 检查 Firehose `DeliveryToS3.Success` 指标，确保 Firehose 已尝试将数据放入 Amazon S3 存储桶。有关更多信息，请参阅 使用指标监控亚马逊数据 Firehose CloudWatch。
- 如果尚未启用错误日志记录功能，则启用它并检查是否存在传输失败错误日志。有关更多信息，请参阅 使用日志监控亚马逊数据 Firehose CloudWatch。
- 如果你在日志中看到一条错误消息，上面写着 “在调用 Amazon S3 服务 InternalServerError 时遇到了 Firehos e。该操作将被重试；如果错误持续存在，请联系 S3 以进行解决。”，这可能是由于 S3 中单个分区的请求速率显著增加。您可以优化 S3 前缀设计模式以缓解此问题。有关更多信息，请参阅最佳实践设计模式：优化 Amazon S3 性能。如果这不能解决问题，请联系 Supp AWS ort 寻求进一步帮助。
- 确保在 Firehose 流中指定的 Amazon S3 存储桶仍然存在。
- 如果启用了使用 Lambda 进行的数据转换，确保在 Firehose 流中指定的 Lambda 函数仍然存在。
- 确保在 Firehose 流中指定的 IAM 角色有权限访问您的 S3 存储桶和 Lambda 函数（如果启用了数据转换）。此外，请确保 IAM 角色有权访问 CloudWatch 日志组和日志流以检查错误日志。有关更多信息，请参阅 授予 Firehose 访问 Amazon S3 目的地的权限。
- 如果使用数据转换，确保您的 Lambda 函数不会返回有效负载大小超过 6MB 的响应。有关更多信息，请参阅 Amazon 数据 FirehoseData 转换。

Amazon Redshift 问题排查

如果数据未传输到您的 Amazon Redshift 预置集群或 Amazon Redshift Serverless 工作组，请检查以下各项。

数据在加载到 Amazon Redshift 之前会先传输到 S3 存储桶。如果数据未传输至 S3 存储桶，请参阅Amazon S3 故障排除。

- 检查 Firehose `DeliveryToRedshift.Success` 指标，确保 Firehose 已尝试将数据从 S3 存储桶复制到 Amazon Redshift 预置集群或 Amazon Redshift Serverless 工作组。有关更多信息，请参阅使用指标监控亚马逊数据 Firehose CloudWatch。
- 如果尚未启用错误日志记录功能，则启用它并检查是否存在传输失败错误日志。有关更多信息，请参阅 使用日志监控亚马逊数据 Firehose CloudWatch。
- 查看 Amazon Redshift `STL_CONNECTION_LOG` 表，查看 Firehose 是否可以成功建立连接。在该表中，应该能够根据用户名查看连接及其状态。有关更多信息，请参阅《Amazon Redshift 数据库开发人员指南》中的 `STL_CONNECTION_LOG`。
- 如果前面的检查显示正在建立连接，请检查 Amazon Redshift `STL_LOAD_ERRORS` 表以验证 COPY 失败的原因。有关更多信息，请参阅《Amazon Redshift 数据库开发人员指南》中的 `STL_LOAD_ERRORS`。
- 确保 Firehose 流中的 Amazon Redshift 配置正确且有效。
- 确保在 Firehose 流中指定的 IAM 角色可以访问 S3 存储桶（Amazon Redshift 从该存储桶复制数据），以及用于数据转换的 Lambda 函数（如果启用了数据转换）。此外，请确保 IAM 角色有权访问 CloudWatch 日志组和日志流以检查错误日志。有关更多信息，请参阅 授予 Firehose 对 Amazon Redshift 目的地的访问权限。
- 如果您的 Amazon Redshift 预置集群或 Amazon Redshift Serverless 工作组位于虚拟私有云（VPC）中，确保该群集允许从 Firehose IP 地址访问。有关更多信息，请参阅 授予 Firehose 对 Amazon Redshift 目的地的访问权限。
- 确保 Amazon Redshift 预置集群或 Amazon Redshift Serverless 工作组公开可用。
- 如果使用数据转换，确保您的 Lambda 函数不会返回有效负载大小超过 6MB 的响应。有关更多信息，请参阅 Amazon 数据 FirehoseData 转换。

对亚马逊 OpenSearch 服务进行故障排除

如果数据未传送到您的 OpenSearch 服务域，请检查以下内容。

数据可以同时备份到 Amazon S3 存储桶。如果数据未传输至您的 S3 存储桶，请参阅Amazon S3 故障排除。

- 检查 Firehose `IncomingBytes` 和 `IncomingRecords` 指标，确保数据成功发送到您的 Firehose 流。有关更多信息，请参阅 使用指标监控亚马逊数据 Firehose CloudWatch。
- 如果启用了使用 Lambda 进行的数据转换，请检查 Firehose `ExecuteProcessingSuccess` 指标，确保 Firehose 已尝试调用 Lambda 函数。有关更多信息，请参阅 使用指标监控亚马逊数据 Firehose CloudWatch。
- 检查 Firehose `DeliveryToAmazonOpenSearchService.Success` 指标，确保 Firehose 已尝试将数据索引到服务集群。 OpenSearch 有关更多信息，请参阅 使用指标监控亚马逊数据 Firehose CloudWatch。
- 如果尚未启用错误日志记录功能，则启用它并检查是否存在传输失败错误日志。有关更多信息，请参阅 使用日志监控亚马逊数据 Firehose CloudWatch。
- 确保您的 Firehose 直播中的 OpenSearch 服务配置准确有效。
- 如果启用了使用 Lambda 进行的数据转换，确保在 Firehose 流中指定的 Lambda 函数仍然存在。此外，请确保 IAM 角色有权访问 CloudWatch 日志组和日志流以检查错误日志。有关更多信息，请参阅 FirehoseAccess 向公共 OpenSearch 服务目的地授权。
- 确保在您的 Firehose 流中指定的 IAM 角色可以访问您的 OpenSearch 服务集群、S3 备份存储桶和 Lambda 函数（如果启用了数据转换）。此外，请确保 IAM 角色有权访问 CloudWatch 日志组和日志流以检查错误日志。有关更多信息，请参阅 授予 Firehose 访问公共 OpenSearch 服务目标的权限。
- 如果使用数据转换，确保您的 Lambda 函数不会返回有效负载大小超过 6MB 的响应。有关更多信息，请参阅 Amazon 数据 FirehoseData 转换。
- Amazon Data Firehosecurtly 不支持将日志传送 CloudWatch 到亚马逊 OpenSearch 服务目标，因为亚马逊 CloudWatch 将多个日志事件合并到一个 Firehose 记录中，并且 OpenSearch 亚马逊服务无法在一条记录中接受多个日志事件。作为替代方案，您可以考虑在 CloudWatch 日志中使用亚马逊 OpenSearch 服务的订阅筛选条件。

Splunk 问题排查

如果数据未传输到您的 Splunk 终端节点，请检查以下各项。

- 如果您的 Splunk 平台在 VPC 中，请确保 Firehose 可以访问它。有关更多信息，请参阅 在 VPC 中访问 Splunk。

- 如果您使用 AWS 负载均衡器，请确保它是 Classic 负载均衡器或 Application 负载均衡器。此外，使用为经典负载均衡器禁用的 Cookie 有效期启用基于持续时间的粘接会话，应用程序负载均衡器的有效期设置为最大值（7 天）。有关如何执行此操作的信息，请参阅 C lassic Load Balancer 或 Application Load Bal ancer 的 Duration-Based 会话粘性。
- 检查 Splunk 平台要求。适用于 Firehose 的 Splunk 插件需要 6.6.X 或更高版本的 Splunk 平台。有关更多信息，请参阅适用于亚马逊 Kinesis Fireh Add-on ose 的 Splunk。
- 如果在 Firehose 和 HTTP 事件收集器（HEC）节点之间具有代理（Elastic Load Balancing 或其他代理），请务必启用粘性会话以支持 HEC 确认（ACK）。
- 确保您使用有效的 HEC 令牌。
- 确保启用了 HEC 令牌。
- 检查是否为发送到 Splunk 的数据正确设置格式。有关更多信息，请参阅为 HTTP 事件收集器的事件设置格式。
- 确保为 HEC 令牌和输入事件配置了有效索引。
- 如果由于 HEC 节点出现服务器错误而无法上传到 Splunk，将会自动重试该请求。如果所有重试都失败，数据将备份到 Amazon S3。检查您的数据是否出现在 Amazon S3 中，这种情况表明出现此类失败。
- 确保在您的 HEC 令牌上启用了索引器确认。
- 在 Firehose 流的 Splunk 目的地配置中增加 `HECAcknowledgmentTimeoutInSeconds` 值。
- 在 Firehose 流的 Splunk 目的地配置中的 `RetryOptions` 下增加 `DurationInSeconds` 值。
- 检查 HEC 的运行状况。启用运行状况检查是向 Splunk 传输数据的先决条件。
- 如果使用数据转换，确保您的 Lambda 函数不会返回有效负载大小超过 6MB 的响应。有关更多信息，请参阅 Amazon Data Firehose 数据转换。
- 确保名为 `ackIdleCleanup` 的 Splunk 参数设置为 `true`。默认情况下，它设置为 false。若要将此参数设置为 `true`，请执行以下操作：
  - 对于托管 Splunk 云部署，请使用 Splunk 支持门户提交案例。在此情况下，应请求 Splunk 支持人员启用 HTTP 事件收集器，在 `inputs.conf` 中将 `ackIdleCleanup` 设置为 `true`，并创建或修改要用于此插件的负载均衡器。
  - 对于分布式 Splunk Enterprise 部署，请将 `inputs.conf` 文件中的 `ackIdleCleanup` 参数设置为 true。对于 *nix 用户，此文件位于 `$SPLUNK_HOME/etc/apps/splunk_httpinput/local/` 下。对于 Windows 用户，它位于 `%SPLUNK_HOME%\etc\apps\splunk_httpinput\local\` 下。
  - 对于单实例 Splunk Enterprise 部署，请将 `inputs.conf` 文件中的 `ackIdleCleanup` 参数设置为 `true`。对于 *nix 用户，此文件位于 `$SPLUNK_HOME/etc/apps/splunk_httpinput/`

local/ 下。对于 Windows 用户，它位于 %SPLUNK_HOME%\etc\apps\splunk_httpinput\local\ 下。

- 确保在 Firehose 流中指定的 IAM 角色可以访问 S3 备份存储桶以及用于数据转换的 Lambda 函数（如果启用了数据转换）。此外，请确保 IAM 角色有权访问 CloudWatch 日志组和日志流以检查错误日志。有关更多信息，请参阅向 Splunk 目标授权 FirehoseAccess 。
- 要将传输到 S3 错误存储桶（S3 备份）的数据重新驱动回 Splunk，请按照 Splunk 文档中提到的步骤进行操作。
- 参见亚马逊 Kinesis Firehose Add-on 的 Splunk 疑难解答。

排查 Snowflake 问题

本节介绍了使用 Snowflake 作为目的地时常见的问题排查步骤

Firehose 流创建失败

如果为向 S PrivateLink-enabled nowflake 集群传送数据的流创建 Firehose 流失败，则表示 Firehos VPCE-ID e 无法访问。这可能是由以下原因之一导致的：

- 不正确 VPCE-ID。确认没有印刷错误。
- Firehose 在预览版中不支持无区域的 Snowflake URL。使用 Snowflake 账户定位器提供 URL。请参阅 Snowflake 文档了解更多详细信息。
- 确认 Firehose 直播是在与雪花 AWS 区域相同的区域中创建的。
- 如果问题仍然存在，请联系 AWS 支持人员。

传输失败

如果数据未传输到您的 Snowflake 表，请检查以下各项。Snowflake 传输失败的数据将连同与有效载荷对应的错误代码和错误消息一起传输到 S3 错误存储桶。以下是一些常见的错误场景。有关错误代码的完整列表，请参阅 Snowflake 数据传输错误。

- 错误代码: Snowflake.DefaultRoleMissing: 表示在创建 Firehose 直播时未配置雪花角色。如果未配置 Snowflake 角色，则请确保将默认角色设置为指定的 Snowflake 用户。
- 错误代码: Snowflake.ExtraColumns: 表示由于输入有效载荷中有多列，对 Snowflake 的插入被拒绝。不应指定表中不存在的列。请注意，Snowflake 列名称是区分大小写的。如果尽管表中存在列，但传输仍因此错误而失败，则请确保输入有效载荷中列名的大小写与表定义中声明的列名相匹配。

- 错误代码: Snowflake.MissingColumns: 表示由于输入负载中缺少列，对 Snowflake 的插入被拒绝。确保为所有非空列指定值。
- 错误代码: Snowflake.InvalidInput: 当 Firehose 无法将提供的输入有效负载解析为有效的 JSON 格式时，可能会发生这种情况。请确保 json 有效载荷格式正确，没有多余的双引号、引号、转义字符等。Firehose 目前仅支持将单个 JSON 项目作为记录有效载荷，并且不支持 JSON 数组。
- 错误代码: Snowflake.InvalidValue: 表示由于输入负载中的数据类型不正确，传送失败。确保输入有效载荷中指定的 JSON 值符合 Snowflake 表定义中声明的数据类型。
- 错误代码: Snowflake.InvalidTableType: 表示不支持在 Firehose 流中配置的表类型。有关支持的表、列和数据类型，请参阅 snowpipe 流的限制中提供的限制。

> **Note**
>
> 无论出于何种原因，如果在创建 Firehose 流后更改了 Snowflake 目的地上的表定义或角色权限，则 Firehose 可能需要几分钟才能检测到这些更改。如果您因此而看到传输错误，则请尝试删除并重新创建 Firehose 流。

排查 Firehose 端点的可达性问题

如果 Firehose API 遇到超时，则请执行以下步骤来测试端点的可达性：

- 检查 API 请求是否来自 VPC 中的主机。来自 VPC 的所有流量都需要设置 Firehose VPC 端点。如需了解更多信息，请参阅将 Firehose 与配合使用。 AWS PrivateLink
- 如果流量来自公共网络或在特定子网中设置了 Firehose VPC 端点的 VPC，则请从主机运行以下命令来检查网络连接。可以在 Firehose 端点和配额中找到 Firehose 端点。
  - 使用 traceroute 或 tcping 之类的工具来检查网络设置是否正确。如果失败，则请检查您的网络设置：

    例如：

    ```
    traceroute firehose.us-east-2.amazonaws.com
    ```

    或者

    ```
    tcping firehose.us-east-2.amazonaws.com 443
    ```

- 如果显示网络设置正确且以下命令失败，则请检查 Amazon CA（证书颁发机构）是否在信任链中。

  例如：

  ```
  curl firehose.us-east-2.amazonaws.com
  ```

如果上述命令成功，则请再次尝试 API 以查看 API 是否返回了响应。

HTTP 端点问题排查

本节介绍在处理 Amazon Data Firehose 向通用 HTTP 终端节点目标和合作伙伴目的地（包括 Datadog、Dynatrace、MongoDB、New Relic、Splunk 或 Sumo Logic）传输数据时的常见故障排除步骤。 LogicMonitor就本节而言，所有适用的目标均称为 HTTP 端点。确保在 Firehose 流中指定的 IAM 角色可以访问 S3 备份存储桶以及用于数据转换的 Lambda 函数（如果启用了数据转换）。此外，请确保 IAM 角色有权访问 CloudWatch 日志组和日志流以检查错误日志。有关更多信息，请参阅授予 Firehose 访问终端节点目标的 HTTP 权限。

> **Note**
>
> 本节中的信息不适用于以下目的地：Splunk、S OpenSearch ervice、S3 和 Redshift。

CloudWatch 日志

强烈建议您为启用CloudWatch 日志记录。只有在传输到目标出现错误时才会发布日志。

目标异常

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"
}
```

目标异常表示 Firehose 能够建立到端点的连接并发出 HTTP 请求，但未收到 200 响应代码。不是 200 的 2xx 响应也会导致目标异常。Amazon Data Firehose 将从配置的终端节点收到的响应代码和截断的响应负载记录到日志中。 CloudWatch 由于 Amazon Data Firehose 会在不进行修改或解释的情况下日志记录响应代码和有效负载，因此端点需要提供拒绝 Amazon Data Firehose 的 HTTP 传输请求的具体原因。以下是针对这些异常的最常见的问题排查建议：

- 400：表示由于 Amazon Data Firehose 配置错误，您发送的请求不正确。确保目标位置具有正确的 url、通用属性、内容编码、访问密钥和缓冲提示。有关所需配置，请参阅特定于目标的文档。
- 401：表示您为 Firehose 流配置的访问密钥不正确或丢失。
- 403：表示您为 Firehose 流配置的访问密钥无权将数据传输到配置的端点。
- 413：表示 Amazon Data Firehose 发送到端点的请求负载太大，端点无法处理。尝试降低缓冲提示，达到目标的建议大小。
- 429：表示 Amazon Data Firehose 发送请求的速率高于目标可以处理的速率。通过增加缓冲时间来微调缓冲提示， and/or 增加缓冲大小（但仍处于目标的限制范围内）。
- 5xx：表示目标存在问题。Amazon Data Firehose 服务仍在正常运行。

> **Important**
>
> 重要提示：虽然这些是常见的问题排查建议，但特定端点提供响应代码的原因可能不同，应首先遵循特定端点的建议。

无效响应

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"
}
```

无效响应异常表示 Amazon Data Firehose 从端点目标接收到无效响应。响应必须符合响应规范，否则 Amazon Data Firehose 将认为传输尝试失败，并将重新传输相同的数据，直到超过配置的重试持续时间。Amazon Data Firehose 会将不符合响应规范的响应视为失败，即使响应的状态为 200。如果您正在开发兼容 Amazon Data Firehose 的端点，请遵循响应规范，以确保数据成功传输。

以下是一些常见的无效响应类型以及解决方法：

- 无效 JSON 或意外字段：表示无法将响应正确解串为 JSON 或包含意外字段。确保响应未经过内容编码。
- 缺失 RequestId：表示响应不包含 requestID。
- RequestId 不匹配：表示响应中的 requestId 与传出的 requestId 不匹配。
- 缺少时间戳：表示响应不包含时间戳字段。时间戳字段必须是数字，而不是字符串。
- 缺少 Content-Type 标头：表示响应不包含 “内容类型： application/json” 标头。不接受其他 content-type。

> **Important**
>
> 重要提示：Amazon Data Firehose 只能将数据传输到遵循 Firehose 请求和响应规范的端点。如果您将目的地配置为第三方服务，请确保使用兼容 Amazon Data Firehose 的正确端点，该端点可能与公有摄取端点不同。例如，Datadog 的 Amazon Data Firehose 端点是 `https://aws-kinesis-http-intake.logs.datadoghq.com/`，而其公有端点是 `https://api.datadoghq.com/`。

其他常见错误

下面列出了其他错误代码和定义。

- 错误代码： HttpEndpoint.RequestTimeout-表示终端响应时间超过 3 分钟。如果您是目标的所有者，请缩短目标端点的响应时间。如果您不是目标的所有者，请联系所有者，询问是否可以采取任何措施来缩短响应时间（即减少缓冲提示，从而减少每次请求处理的数据）。
- 错误代码： HttpEndpoint.ResponseTooLarge-表示响应太大。包括标头在内的响应必须小于 1MiB。
- 错误代码： HttpEndpoint.ConnectionFailed-表示无法与配置的端点建立连接。这可能是由于配置的 url 中存在拼写错误，Amazon Data Firehose 无法访问端点，或者端点响应连接请求的时间过长。
- 错误代码： HttpEndpoint.ConnectionReset-表示连接已建立，但已被端点重置或过早关闭。
- 错误代码： HttpEndpoint.SSLHandshakeFailure-表示无法使用配置的端点成功完成 SSL 握手。

MSK As Source 问题排查

本节介绍了使用 MSK As Source 时常见的问题排查步骤

> **Note**
>
> 有关处理、转换或 S3 交付问题的排查，请参阅前面的部分

hose 创建失败

如果使用 MSK As Source 的 hose 创建失败，请检查以下各项：

- 检查源 MSK 集群是否处于活动状态。
- 如果您使用的是私有连接，请确保集群上的私有链接已打开。

  如果您使用的是公有连接，请确保集群上的公开访问已打开。
- 如果您使用的是私有连接，请确保添加基于资源的策略，该策略允许 Firehose 创建私有链接。另请参阅：MSK 跨账户权限。
- 确保源配置中的角色有权限从集群的主题中摄取数据。
- 确保您的 VPC 安全组允许传入流量经过集群引导服务器使用的端口。

hose 暂停

如果您的 hose 处于暂停状态，请检查以下各项

- 检查源 MSK 集群是否处于活动状态。

- 检查源主题是否存在。如果主题被删除并重新创建，则还必须删除并重新创建 Firehose 流。

hose 反压

当超过 BytesPerSecondLimit 每个分区或者正常的传送流程缓慢或停止时，的值 DataReadFromSource.Backpressured 将为 1。

- 如果你点击了， BytesPerSecondLimit 请检查 DataReadFromSource.Bytes 指标并申请提高限额。
- 检查日 CloudWatch 志、目标指标、数据转换指标和格式转换指标以确定瓶颈。

数据新鲜度不正确

数据新鲜度似乎不正确

- Firehose 根据使用记录的时间戳来计算数据新鲜度。为了确保在将生产者记录保留在 Kafka 的代理日志中时正确记录此时间戳，请将 Kafka 主题时间戳类型配置设置为 `message.timestamp.type=LogAppendTime`。

MSK 集群连接问题

以下过程说明了如何验证与 MSK 集群的连接。有关设置 Amazon MSK 客户端的详细信息，请参阅 Amazon Managed Streaming for Apache Kafka Developer Guide 中的 Getting started using Amazon MSK。

要验证与 MSK 集群的连接

1. 创建一个 Unix-based （最好是 AL2）亚马逊 EC2 实例。如果您在集群上仅启用了 VPC 连接，则请确保您的 EC2 实例在同一 VPC 中运行。一旦实例可用，SSH 连接到该实例。有关更多信息，请参阅《Amazon EC2 用户指南》中的本教程。
2. 通过运行以下命令，使用 Yum 软件包管理器安装 Java。有关更多信息，请参阅《Amazon Corretto 8 用户指南》中的安装说明。

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

3. 通过运行以下命令安装 AWS 客户端。

   ```
   curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
   ```

```
unzip awscliv2.zip
sudo ./aws/install
```

4. 运行以下命令下载 Apache Kafka 客户端 2.6* 版本。

```
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. 转到 kafka_2.12-2.6.2/libs 目录，然后运行以下命令以下载 Amazon MSK IAM JAR 文件。

```
wget https://github.com/aws/aws-msk-iam-auth/releases/download/v1.1.3/aws-msk-iam-
auth-1.1.3-all.jar
```

6. 在 Kafka bin 文件夹中创建 client.properties 文件。

7. 将 awsRoleArn 替换为您在 Firehose SourceConfiguration 中使用的角色 ARN，然后验证证书位置。允许您的 AWS 客户用户代入角色awsRoleArn。 AWS 客户端用户将尝试代入您在此处指定的角色。

```
[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. 运行以下 Kafka 命令以列出主题。如果您的连接是公共的，则请使用公共端点 Bootstrap 服务器。如果您的连接是私有的，则请使用私有端点 Bootstrap 服务器。

```
bin/kafka-topics.sh --list --bootstrap-server <bootstrap servers> --command-config
 bin/client.properties
```

如果请求成功，则您应该会看到类似于以下示例的输出。

```
[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. 如果您在运行之前的脚本时遇到任何问题，则请验证您所提供的引导服务器是否可以通过指定端口访问。为执行此操作，您可以下载并使用 telnet 或类似的实用程序，如以下命令所示。

```
sudo yum install telnet
telnet <bootstrap servers><port>
```

如果请求成功，您将获得以下输出。这意味着您可以连接到本地 VPC 中的 MSK 集群，且引导服务器在指定端口上运行正常。

```
Connected to ..
```

10. 如果请求不成功，则请检查您的 VPC 安全组上的入站规则。例如，您可以在入站规则上使用以下属性。

```
Type: All traffic
Port: Port used by the bootstrap server (e.g. 14001)
Source: 0.0.0.0/0
```

如上一步所示，重试 telnet 连接。如果您仍然无法连接或 Firehose 连接仍然失败，则请联系 AWS Support。

Amazon Data Firehose 配额

本节介绍 Amazon Data Firehose 中的当前配额（以前称为限制）。除非另行指定，否则每个限额将基于区域应用。

Service Quotas 控制台是一个中心位置，您可以在其中查看和管理 AWS 服务配额，并请求增加您使用的许多资源的配额。使用我们提供的配额信息来管理您的 AWS 基础架构。请根据需要请提前计划以请求提高配额。

有关更多信息，请参阅 Amazon Web Services 一般参考中的 Amazon Data Firehose endpoints and quotas。

下一节显示 Amazon Data Firehose 具有以下配额。

- 使用 Amazon MSK 作为 Firehose 流的来源，每个 Firehose 流的默认配额为每个分区 MB/sec 10 的读取吞吐量，最大记录大小为 10MB。
- 如果将 Amazon MSK 作为 Firehose 流的来源，则如果启用 AWS Lambda，则最大记录大小为 6 MB；如果禁用 Lambda，则最大记录大小为 10 MB。 AWS Lambda 将其传入记录上限为 6 MB，Amazon Data Firehose 将超过 6Mb 的记录转发到错误 S3 存储桶。如果禁用 Lambda，Firehose 会将其传入记录限制为 10 MB。如果 Amazon Data Firehose 从 Amazon MSK 收到超过 10 MB 的记录大小，则 Amazon Data Firehose 会将该记录传输到 S3 错误存储桶，并向您的账户发出 Cloudwatch 指标。有关 Lambda 限制的更多信息， AWS 请参阅 Lambda 配额。
- 在 Firehose 流上启用动态分区后，可以为该 Firehose 流创建 500 个活动分区的默认配额。活动分区计数是传输缓冲区内活动分区的总数。例如，如果动态分区查询每秒构造 3 个分区，并且您有一个缓冲区提示配置，每 60 秒触发一次传输，那么平均您将拥有 180 个活动分区。在分区中传输数据后，该分区将不再处于活动状态。如果您需要更多分区，则可以创建更多 Firehose 流，并在这些流之间分配活动分区。
- 在 Firehose 流上启用动态分区后，每个活动分区支持的最大吞吐量为每秒 1 GB。
- 每个账户对每个区域的 Firehose 流数量具有以下配额：
  - 美国东部（弗吉尼亚州北部）、美国东部（俄亥俄州）、美国西部（俄勒冈州）、欧洲地区（爱尔兰）、亚太地区（东京）：5,000 个 Firehose 流
  - 欧洲（法兰克福）、欧洲（伦敦）、亚太地区（新加坡）、亚太地区（悉尼）、亚太地区（首尔）、亚太地区（孟买）、（美国西部）、加拿大 AWS GovCloud （西部）、加拿大（中部）：2,000 个 Firehose 直播

- 欧洲（巴黎）、欧洲（米兰）、欧洲（斯德哥尔摩）、亚太地区（香港）、亚太地区（大阪）、南美洲（圣保罗）、中国（宁夏）、中国（北京）、中东（巴林）、（美国东部）、非洲（开普敦）AWS GovCloud ：500 个 Firehose 直播
- 欧洲（苏黎世）、欧洲（西班牙）、亚太地区（海得拉巴）、亚太地区（雅加达）、亚太地区（墨尔本）、中东（阿联酋）、以色列（特拉维夫）、加拿大西部（卡尔加里）、加拿大（中部）、亚太地区（马来西亚）、亚太地区（泰国）、墨西哥（中部）：100 个 Firehose 流
- 如果超出此数字，调用 CreateDeliveryStream 会导致 `LimitExceededException` 异常。要提升此配额，可以使用 Service Quotas（如果该服务在您的区域可用）。有关使用服务限额的信息，请参阅请求增加配额。

- 将 D irect PUT 配置为数据源时，每个 Firehose 流为PutRecord和PutRecordBatch请求提供以下组合配额：
  - 美国东部（弗吉尼亚北部）、美国西部（俄勒冈）和欧洲（爱尔兰）：500,000 records/second, 2,000 requests/second, and 5 MiB/second。
  - 对于其他 AWS 区域：100,000 records/second, 1,000 requests/second, and 1 MiB/second。

  如果由于较高的数据摄取量超过 Firehose 流的吞吐能力而导致直接 PUT 流受到节流，则 Amazon Data Firehose 会自动提高流的吞吐量限制，直到节流得到控制。根据吞吐量的增加和节流，Firehose 可能需要更长的时间才能将数据流的吞吐量提高到所需的水平。因此，请继续重试失败的数据摄取记录。如果您预计数据量会突然大幅增加，或者您的新数据流需要的吞吐量比默认吞吐量限制更高，则请求提高吞吐量限制。

  这三个配额成比例扩展。例如，如果您将美国东部（弗吉尼亚北部）、美国西部（俄勒冈）或欧洲（爱尔兰）的吞吐量配额增加到 10 MiB/second, the other two quota increase to 4,000 requests/second and 1,000,000 records/second。

  > **Note**
  >
  > - 请勿使用资源级别限制和配额来控制您对服务的使用。
  > - 当 Kinesis Data Streams 配置为数据来源时，此配额不适用，Amazon Data Firehose 可无限扩展和缩减。
  > - 如果增加的配额远高于运行的流量，这会导致传输到目标的批次很少。从而造成效率低下，并导致目标服务的成本高昂。请确保仅为满足当前运行的流量而增加配额，并在流量增加时进一步提高配额。
  > - 数据记录越少，成本越高。Firehose 摄取定价基于您发送到服务的数据记录数，乘以每条记录的大小，四舍五入到最接近的 5 KB（5120 字节）。因此，在输入数据量（字节）相同的情况下，如果输入记录的数量越多，产生的成本就会越高。例如，如果传入的总

数据量为 5MiB，则与使用 1,000 条记录发送相同数量的数据相比，发送超过 5,000 条记录的 5MiB 数据的成本更高。有关更多信息，请参阅 AWS Calculator 中的 Amazon Data Firehose。

- 每个 Firehose 流最多可存储 24 小时的数据记录，以防传送目的地不可用且源不可用。 DirectPut如果源是 Kinesis Data Streams（KDS），且目标位置不可用，则数据将根据您的 KDS 配置保留。
- 在进行 base64 编码之前，发送到 Amazon Data Firehose 的记录的最大大小为 1,000 KiB。
- PutRecordBatch 操作每次调用可处理 500 条记录或 4 MB，以较小者为准。无法更改此配额。
- 以下每项操作每秒最多可以提供 5 次调用，这是硬性限制。
  - `CreateDeliveryStream`
  - `DeleteDeliveryStream`
  - `DescribeDeliveryStream`
  - `ListDeliveryStreams`
  - `UpdateDestination`
  - `TagDeliveryStream`
  - `UntagDeliveryStream`
  - `ListTagsForDeliveryStream`
  - `StartDeliveryStreamEncryption`
  - `StopDeliveryStreamEncryption`
- 缓冲间隔提示范围：60 秒 - 900 秒。
- 对于从 Amazon Data Firehose 到 Amazon Redshift 的传输，仅支持可公开访问的 Amazon Redshift 集群。
- Amazon Red OpenSearch shift 和服务交付的重试持续时间范围从 0 秒到 7,200 秒不等。
- 当目标为亚马逊 S3、亚马逊 Redshift 或 OpenSearch 服务时，Amazon Data Firehose 允许每个分片最多 5 次未完成的 Lambda 调用。对于 Splunk，配额为每个分片 10 次未完成的 Lambda 调用。
- 您可以使用类型 `CUSTOMER_MANAGED_CMK` 的 CMK 最多加密 500 个 Firehose 流。

文档历史记录

下表介绍了 Amazon Data Firehose 文档的重要更改。

更改	描述	更改日期
已将作为来源的数据库移除（公开预览）	现已移除作为来源的数据库（公共预览）。	2025 年 9 月 24 日
增加了对 Glue 多目录层次结构的支持	这简化了 Firehose 与 Amazon S3 表的集成，而无需在默认数据目录和 `S3TablesCatalog` 之间建立资源链接。请参阅设置到 Amazon S3 表的 Firehose 流。	2025 年 5 月 14 日
已将数据库添加为源（公开预览）	现在，您可以将数据库更改复制到 Amazon S3 中的 Apache Iceberg 表。	2024 年 11 月 15 日
将 Apache Iceberg 表添加为目的地的正式发布（GA）版本	您可以创建 Firehose 流，将 Apache Iceberg 表作为目的地。请参阅将数据传输到 Apache Iceberg 表。	2024 年 9 月 30 日
添加了数据类型示例	添加了 Apache Iceberg 支持的数据类型的示例。请参阅了解支持的数据类型。	2024 年 8 月 22 日
新上线区域	Amazon Data Firehose 现已在亚太地区（马来西亚）推出。请参阅Amazon Data Firehose 配额。	2024 年 8 月 22 日
将 Apache Iceberg 表添加为目的地（公开预览）	您可以创建 Firehose 流，将 Apache Iceberg 表作为目的地。请参阅将数据传输到 Apache Iceberg 表。	2024 年 7 月 25 日
Snowflake 的缓冲提示	Snowflake 现在支持缓冲提示。请参阅the section called “为 Snowflake 配置目的地设置”。	2024 年 7 月 25 日
Snowflake 作为新区域的目的地	Snowflake 现已在亚太地区（新加坡）、亚太地区（首尔）和亚太地区（东京）和亚太地区（悉尼）作为目的	2024 年 7 月 25 日

更改	描述	更改日期
	地发布。请参阅the section called “为 Snowflake 配置目的地设置”。	
重构了用户指南部分	简化了用户指南中各部分的导航。请参阅向 Firehose 流发送数据和排查错误。	2024 年 7 月 5 日
Amazon Data Firehose 与 AWS Secrets Manager	现在，您可以使用 Secrets Manager 安全地访问您的密钥，并自动轮换凭证。请参阅the section called “使用 AWS Secrets Manager进行身份验证”。	2024 年 6 月 6 日
增加了对摄取 Dynatrace 日志的支持	现在，您可以将日志和事件发送到 Dynatrace，以供进一步分析。请参阅the section called “为 Dynatrace 配置目的地设置”。	2024 年 4 月 18 日
Snowflake 作为目的地的正式发布（GA）版本	Snowflake 现可作为目的地正式发布。请参阅the section called “为 Snowflake 配置目的地设置”。	2024 年 4 月 17 日
Amazon Kinesis Data Firehose 现在被称为 Amazon Data Firehose	Amazon Kinesis Data Firehose 已更名为 Amazon Data Firehose。请参阅 什么是 Amazon Data Firehose。	2024 年 2 月 9 日
已将 Snowflake 添加为目的地（公开预览）	您可以创建将 Snowflake 作为目的地的 Firehose 流。请参阅the section called “为 Snowflake 配置目的地设置”。	2024 年 1 月 19 日
增加了日志的自动解压功能 CloudWatch	您可以对新的或现有流启用解压缩，将解压缩后的 CloudWatch 日志数据发送到 Firehose 目的地。请参阅the section called “向 Firehose 发送 CloudWatch 日志”。	2023 年 12 月 15 日
增加了 Splunk Observability Cloud 作为目标	您可以创建一个以 Splunk Observability Cloud 为目的地的 Firehose 流。请参阅the section called “为 Splunk Observability Cloud 配置目的地设置”。	2023 年 10 月 3 日

更改	描述	更改日期
增加了 Amazon Managed Streaming for Apache Kafka 作为数据来源	现在，您可以配置 Amazon MSK 将信息发送到 Firehose 流。请参阅the section called “为 Amazon MSK 配置源设置”。	2023 年 9 月 26 日
增加了对服务目标的 documentID 类型的支持 OpenSearch	如果 OpenSearch 服务是你的 Firehose 直播的目标，则 documentID 类型表示设置文档 ID 的方法。支持的方法有 Firehose 生成的文档 ID 和 OpenSearch 服务生成的文档 ID。请参阅the section called “配置目的地设置”。	2023 年 5 月 10 日
增加了对动态分区的支持	增加了对 Amazon Data Firehose 中流数据的连续动态分区的支持。请参阅对流数据进行分区。	2021 年 8 月 31 日
增加了有关自定义前缀的主题。	增加了一个主题，介绍在为传输到 Amazon S3 的数据构建自定义前缀时可以使用的表达式。请参阅the section called “了解 Amazon S3 对象的自定义前缀”。	2018 年 12 月 20 日
增加了新的 Amazon Data Firehose 教程	增加了一个教程，演示如何通过 Amazon Data Firehose 将 Amazon VPC 流日志发送到 Splunk。请参阅使用 Amazon Data Firehose 将 VPC 流日志摄取到 Splunk。	2018 年 10 月 30 日
增加了四个新的 Amazon Data Firehose 区域	增加了巴黎、孟买、圣保罗和伦敦。有关更多信息，请参阅 Amazon Data Firehose 配额。	2018 年 6 月 27 日
增加了两个新的 Amazon Data Firehose 区域	增加了首尔和蒙特利尔。有关更多信息，请参阅 Amazon Data Firehose 配额。	2018 年 13 月 6 日
新增了将 Kinesis Streams 作为源的功能	已将 Kinesis Streams 作为 Firehose 流的记录的潜在源添加。有关更多信息，请参阅 为 Firehose 流选择来源和目的地。	2017 年 8 月 18 日

更改	描述	更改日期
更新了控制台文档	更新了 Firehose 流创建向导。有关更多信息，请参阅教程：从控制台创建 Firehose 流。	2017 年 7 月 19 日
新数据转换	您可以配置 Amazon Data Firehose 在数据传输之前转换数据。有关更多信息，请参阅 在 Amazon Data Firehose 中转换源数据。	2016 年 12 月 19 日
新的 Amazon Redshift COPY 重试	您可以配置 Amazon Data Firehose，在对 Amazon Redshift 集群执行 COPY 命令失败时进行重试。有关更多信息，请参阅 教程：从控制台创建 Firehose 流、了解 Amazon Data Firehose 中的数据传输 和 Amazon Data Firehose 配额。	2016 年 5 月 18 日
全新 Amazon Data Firehose 目的地，亚马逊服务 OpenSearch	您可以创建以亚马逊 OpenSearch 服务为目标的 Firehose 直播。有关更多信息，请参阅 教程：从控制台创建 Firehose 流、了解 Amazon Data Firehose 中的数据传输 和 授予 Firehose 访问公共 OpenSearch 服务目标的权限。	2016 年 4 月 19 日
新的增强 CloudWatch 指标和故障排除功能	更新了监控 Amazon Data Firehose和 对 Amazon Data Firehose 中的错误进行问题排查。	2016 年 4 月 19 日
新的增强型 Kinesis 代理	已更新 配置 Kinesis 代理以发送数据。	2016 年 4 月 11 日
新的 Kinesis 代理	增加了 配置 Kinesis 代理以发送数据。	2015 年 10 月 2 日
初始版本	首次发布 Amazon Data Firehose《开发人员指南》。	2015 年 10 月 4 日