Implementation Guide

Streaming Data Solution for Amazon Kinesis

Streaming Data Solution for Amazon Kinesis: Implementation Guide

Table of Contents

Solution overview

Publication date: *August 2020 (last update: June 2024)*

The Streaming Data Solution for Amazon Kinesis allows you to capture, store, process, and deliver real-time streaming data. By automatically configuring the included AWS services, this solution helps you address real-time streaming use cases, for example:

- Capture high volume application log files
- Analyze website clickstreams
- Process database event streams
- Track financial transactions
- Aggregate social media feeds
- Collect IT log files
- Continuously deliver to a data lake

This solution helps accelerate your development lifecycle by minimizing or eliminating the need to model and provision resources using AWS CloudFormation, setup preconfigured Amazon CloudWatch alarms set to recommended thresholds, dashboards, and logging, and manually implement streaming data best practices. This solution is data and logic agnostic, meaning that you can start with boilerplate code and then customize it to your needs.

The solution uses templates where data flows through producers, streaming storage, consumers, and destinations. Producers continuously generate data and send it to streaming storage where it is durably captured and made available for processing by a data consumer. Data consumers process the data and then send it to a destination.

To support multiple use cases and business needs, this solution offers four AWS CloudFormation templates. You can use this solution to test new service combinations as the basis for your production environment, and to improve existing applications. All templates are configured to apply best practices to monitor functionality using dashboards and alarms, and to secure data.

This implementation guide describes architectural considerations and configuration steps for deploying Streaming Data Solution for Amazon Kinesis in the Amazon Web Services (AWS) Cloud. It includes links to AWS CloudFormation templates that launch and configure the AWS services required to deploy this solution using AWS best practices for security and availability.

The guide is intended for IT architects, developers, and DevOps professionals who want to get started quickly with the core streaming services available in the AWS Cloud.

This solution is a demo. We do not recommend using this to handle regulated data such as PII, HIPAA, and GPDR when deployed in production.

Features and benefits

The Streaming Data Solution for Amazon Kinesis provides the following features:

Automated configuration

Automatically configure the AWS services necessary to easily capture, store, process, and deliver streaming data.

Four template options

You can choose from four AWS CloudFormation template options. You can more quickly test new service combinations for your production environment and improve existing applications.

Real-time use cases

You can capture high-volume application logs, analyze clickstream data, continuously deliver to a data lake, and more.

Preconfigured Amazon CloudWatch alarms, dashboards, and logging

This solution comes with Amazon CloudWatch alarms set to recommended thresholds, dashboards for viewing performance metrics, and logging to make it easier for you to monitor the overall performance of the solution.

Customizable source code

Customize the solution's boilerplate code, and then use the monitoring capabilities to quickly transition from testing to production.

Integration with AWS Service Catalog AppRegistry and Application Manager, a capability of AWS Systems Manager

This solution includes a AWS Service Catalog AppRegistry resource to register the solution's CloudFormation template and its underlying resources as an application in both Service Catalog

AppRegistry and Application Manager. With this integration, you can centrally manage the solution’s resources.

Use cases

Option 1: Capture data from non-AWS environments such as mobile clients

This option uses an Amazon API Gateway as a layer of abstraction, which allows you to implement custom authentication approaches for data producers, control quotas for specific producers, and change the target Kinesis stream. This template uses AWS Lambda as the data consumer, which is best suited for use cases that don’t require internal state like filtering, business event processing, and data cleansing. AWS Lambda offers a small surface area for error scenarios and is simple to scale and operate.

Option 2: Capture and analyze streaming data

This option is intended for use cases such as streaming extract-transform-load (ETL), real-time analytics, predictive analytics, and machine learning. It uses Apache Flink and provides a fully managed service to handle backups for snapshots, a Kinesis Data Analytics implementation of an Apache Flink *Savepoint*, automatically. This option also supports the Amazon Kinesis Producer Library (KPL), which is best suited when you control the code that is written to the Kinesis Data Streams. This control allows you to implement cost optimizations through buffering on the data producer and has fewer resources to manage compared to alternatives. Amazon Kinesis Data Analytics for Apache Flink is used as the data consumer, which is best suited when you require capabilities such as durable application and exactly-once processing, that are very efficient processes for high volume data streams with low latency and high availability.

Option 3: Backup streaming data to S3

This option uses Amazon Kinesis Data Firehose. Use this option when you want a simple way to back up incoming streaming data with minimal administration for the processing layer and ability to send data into Amazon Simple Storage Service (Amazon S3), among other destinations in near real time. Kinesis Data Firehose takes care of compression and encryption, minimizing the amount of storage used at the destination and increasing security.

Option 4: Capture, analyze, and forward streaming data to an external endpoint

This option uses Apache Flink, and showcases how to asynchronously invoke an external endpoint in a streaming application, for example, when you want to enrich or filter incoming events. The external API can be any integration supported by API Gateway, such as a Lambda function or an Amazon SageMaker endpoint.

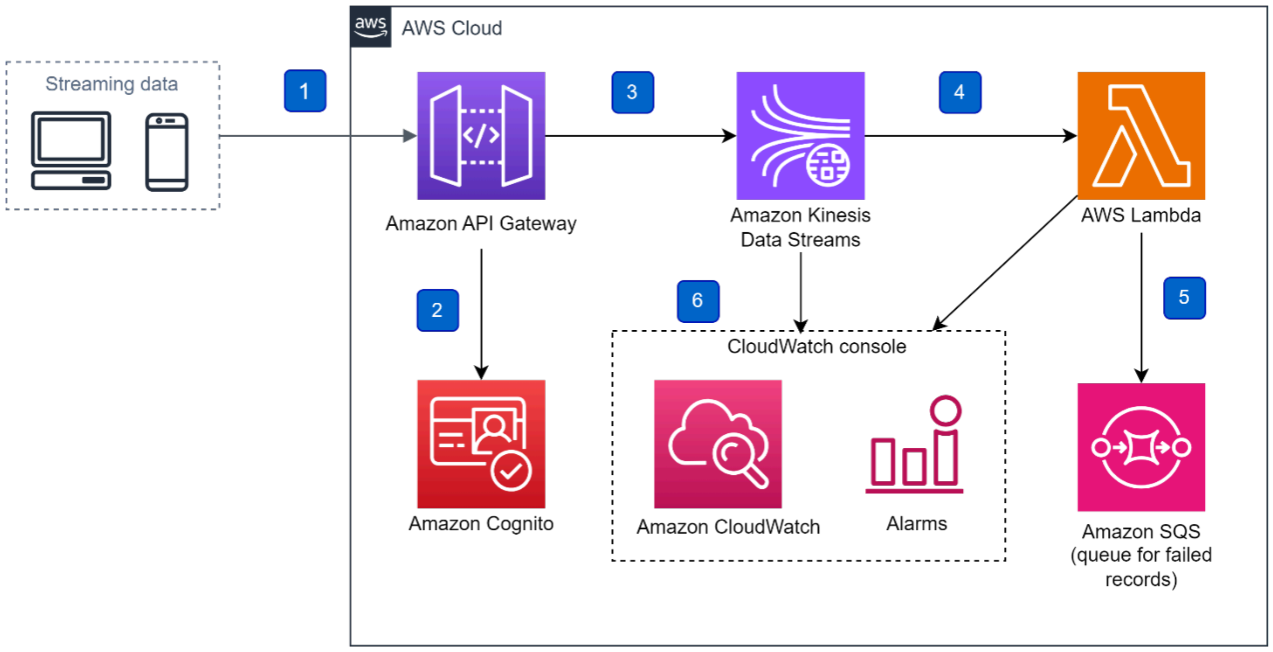
Architecture overview

This solution automatically configures the core AWS services necessary to capture, store, process, and deliver streaming data. Four reference architecture options are available. Each option includes customizable demo code and sample applications.

All AWS CloudFormation resources were created using AWS Solutions Constructs.

Option 1: Deploy the AWS CloudFormation template using API Gateway, Kinesis Data Streams, and Lambda

Deploying the `aws-streaming-data-solution-for-kinesis-using-api-gateway-and-lambda` AWS CloudFormation template builds the following environment in the AWS Cloud.



AWS CloudFormation template using Amazon API Gateway, Kinesis Data Streams, and AWS Lambda reference architecture

This AWS CloudFormation template deploys a reference architecture that includes the following:

1. An API Gateway REST API that acts as a proxy to Amazon Kinesis Data Streams, adding either an individual data record or a list of data records.
2. An Amazon Cognito user pool is used to control who can invoke REST API methods.

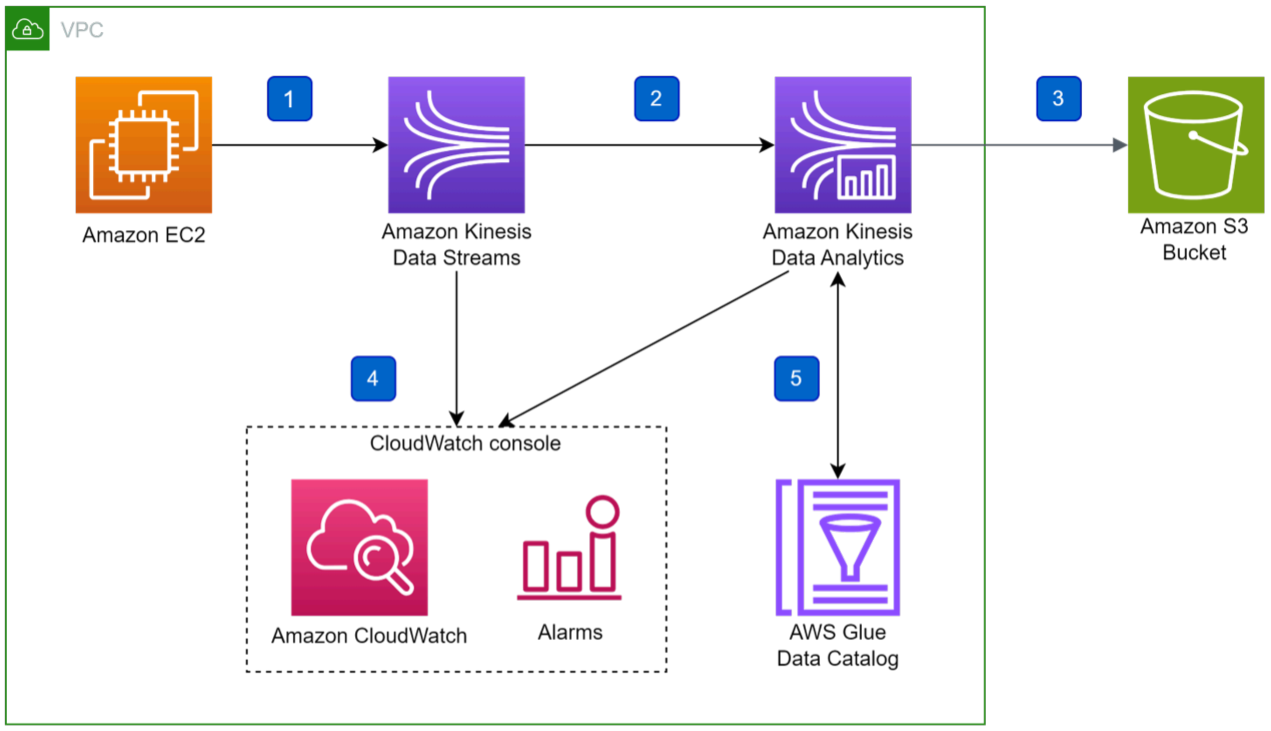
3. Kinesis Data Streams to store the incoming streaming data.
4. An AWS Lambda function processes the records from the data stream.
5. Errors and failed records that occur during AWS Lambda processing are annotated, and the events are stored in Amazon Simple Queue Service (Amazon SQS).
6. An Amazon CloudWatch dashboard monitors application health, progress, resource utilization, events, and errors.

Note

For information about how AWS services handle errors, refer to Error handling in the *AWS Lambda Developer Guide* and AWS Lambda Supports Failure-Handling Features.

Option 2: Deploy the AWS CloudFormation template using Amazon EC2, KPL, Kinesis Data Streams, Managed Service for Apache Flink, and CloudWatch

Deploying the `aws-streaming-data-solution-for-kinesis-using-kpl-and-kinesis-data-analytics` AWS CloudFormation template builds the following environment in the AWS Cloud.



AWS CloudFormation template using Amazon EC2, KPL, Kinesis Data Streams, Managed Service for Apache Flink, and CloudWatch reference architecture

This AWS CloudFormation template deploys a reference architecture that includes the following:

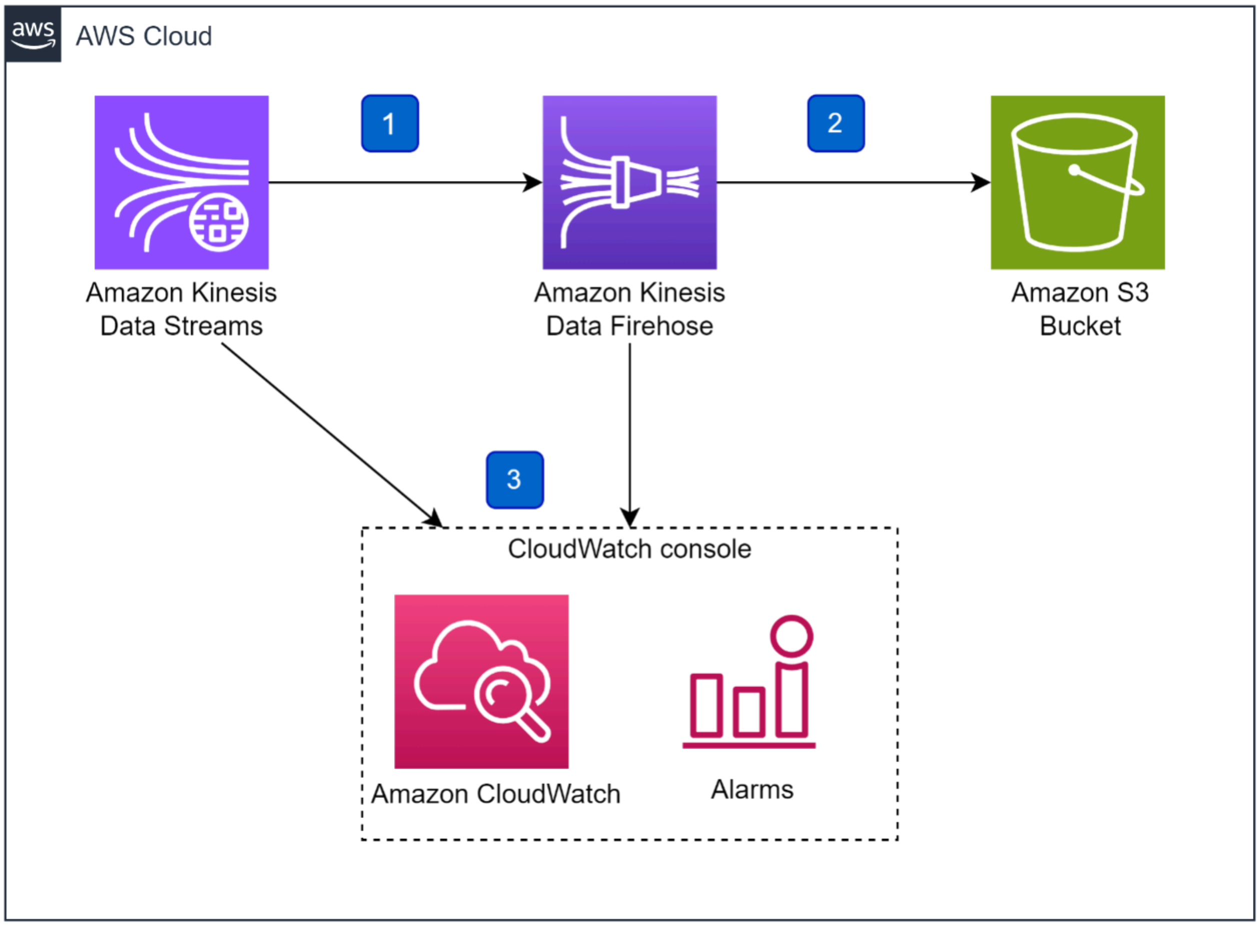
1. An Amazon Elastic Compute Cloud (Amazon EC2) instance that uses the Amazon Kinesis Producer Library (KPL) to generate data.
2. Kinesis Data Streams to store the incoming streaming data.
3. Managed Service for Apache Flink Studio processes the incoming records and saves the processed data in an Amazon Simple Storage Service (Amazon S3) bucket.
4. An Amazon CloudWatch dashboard monitors application health, progress, resource utilization, events, and errors. For information on essential metrics including recommended alarms, refer to Using CloudWatch Alarms with Amazon Managed Service for Apache Flink for Apache Flink in the *Managed Service for Apache Flink Developer Guide*.
5. Kinesis Data Analytics Studio uses an AWS Glue Data Catalog to store metadata tables representing data stores.

Note

You can configure the solution to use an existing Amazon Virtual Private Cloud (Amazon VPC) to allow access to private resources (for example, private databases). By default, Managed Service for Apache Flink does not launch into your VPC, but you can allow your applications to access Amazon Relational Database Service (Amazon RDS) using this solution. For information about connecting to a virtual private cloud in your account, refer to Configuring Managed Service for Apache Flink in the *Managed Service for Apache Flink Developer Guide*.

Option 3: Deploy the AWS CloudFormation template using Kinesis Data Streams, Firehose, and Amazon S3

Deploying the `aws-streaming-data-solution-for-kinesis-using-kinesis-data-firehose-and-amazon-s3` AWS CloudFormation template builds the following environment in the AWS Cloud.



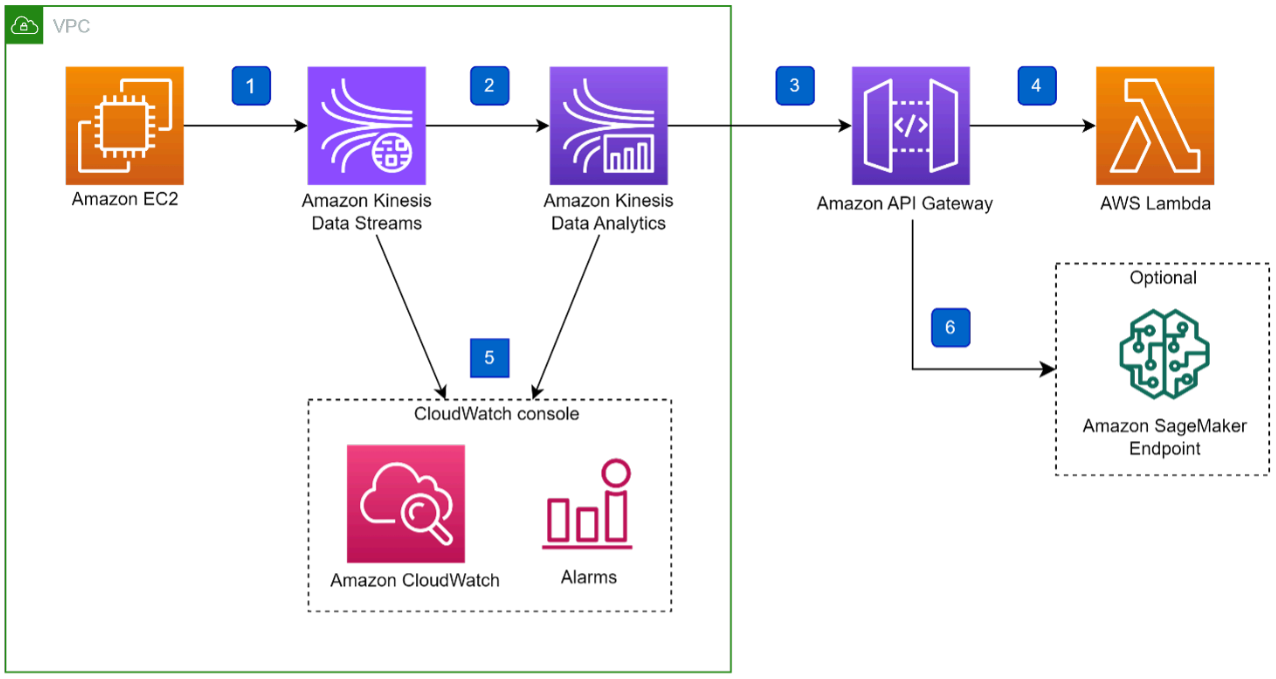
AWS CloudFormation template using Kinesis Data Streams, Kinesis Data Firehose, and S3 reference architecture

This AWS CloudFormation template deploys a reference architecture that includes the following:

1. Kinesis Data Streams stores the incoming streaming data.
2. Kinesis Data Firehose buffers the data before delivering the output to an Amazon S3 bucket. It is a fully managed service that automatically scales to match the throughput of your data and requires no ongoing administration.
3. An Amazon CloudWatch dashboard monitors the data ingestion and buffering. CloudWatch alarms are set on essential metrics for Kinesis Data Firehose. For information on essential metrics (including recommended alarms), refer to Monitoring Kinesis Data Firehose Using CloudWatch Metrics in the *Kinesis Data Firehose Developer Guide*.

Option 4: Deploy the AWS CloudFormation template using Kinesis Data Streams, Managed Service for Apache Flink, and API Gateway

Deploying the `aws-streaming-data-solution-for-kinesis-using-kinesis-data-analytics-and-api-gateway` AWS CloudFormation template builds the following environment in the AWS Cloud.



AWS CloudFormation template using Kinesis Data Streams, Managed Service for Apache Flink, and API Gateway reference architecture

This AWS CloudFormation template deploys a reference architecture similar to Option 2 and includes the following:

1. An Amazon Elastic Compute Cloud (Amazon EC2) instance that uses the Amazon Kinesis Producer Library (KPL) to generate data.
2. Kinesis Data Streams stores the incoming streaming data.
3. Managed Service for Apache Flink processes the incoming records and asynchronously invokes an external endpoint.
4. The demo application invokes a Lambda function.

5. An Amazon CloudWatch dashboard monitors application health, progress, resource utilization, events, and errors. For information on essential metrics including recommended alarms, refer to Using CloudWatch Alarms with Amazon Managed Service for Apache Flink for Apache Flink in the *Amazon Managed Service for Apache Flink Developer Guide*.
6. The external API can be any integration supported by API Gateway (for example, an Amazon SageMaker endpoint).

Note

You can configure the solution to use an existing Amazon Virtual Private Cloud (Amazon VPC) allowing access to private resources (for example, private databases). By default, Managed Service for Apache Flink does not launch into your VPC, but you can allow your applications to access Amazon Relational Database Service (Amazon RDS) using this solution. For information about connecting to a virtual private cloud in your account, refer to Configuring Managed Service for Apache Flink in the *Managed Service for Apache Flink Developer Guide*.

AWS Well-Architected design considerations

This solution was designed with best practices from the AWS Well-Architected Framework which helps customers design and operate reliable, secure, efficient, and cost-effective workloads in the cloud.

This section describes how the design principles and best practices of the Well-Architected Framework were applied when building this solution.

Ingesting and processing real-time streaming data requires scalability and low latency to support a variety of applications such as activity tracking, transaction order processing, click-stream analysis, data cleansing, metrics generation, log filtering, indexing, social media analysis, and IoT device data telemetry and metering. These applications are often spiky and process thousands of events per second.

Operational excellence

This section describes how we architected this solution using the principles and best practices of the operational excellence pillar.

The Streaming Data Solution for Amazon Kinesis solution pushes metrics to Amazon CloudWatch to provide observability into the infrastructure; AWS Lambda functions, Amazon Kinesis Streams, Kinesis Data Analytics, Kinesis Data Firehose, S3 buckets, and the rest of the solution components. Errors and failed records that occur during processing are annotated, and the events are stored in Amazon Simple Queue Service.

Security

This section describes how we architected this solution using the principles and best practices of the security pillar.

- All data storage including Amazon S3 buckets have encryption at rest.
- All inter-service communications use AWS IAM roles.
- Communications between end user and Amazon API Gateway uses Bearer token generated and handed by Amazon Cognito.
- All roles used by the solution follows least-privilege access. That is, it only contains minimum permissions required so the service can function properly.

Reliability

This section describes how we architected this solution using the principles and best practices of the reliability pillar.

The Streaming Data Solution for Amazon Kinesis solution uses AWS Serverless Services wherever possible (examples include AWS Lambda, Amazon API Gateway, S3, Kinesis Data Stream) to ensure high availability and recovery from service failure.

Performance efficiency

This section describes how we architected this solution using the principles and best practices of the performance efficiency pillar.

- Using serverless architecture throughout this solution.
- Automatic scaling is turned on by default in Kinesis Data Analytics.
- The ability to launch this solution in any region that supports AWS services in this solution such as: Amazon API Gateway, Kinesis Data Stream, Kinesis Data Analytics, Kinesis Data Firehose, EC2, S3 Bucket, CloudWatch, and AWS Lambda.

- Multiple options are available to quickly carry out comparative testing using different types of service configurations.

Cost optimization

This section describes how we architected this solution using the principles and best practices of the cost optimization pillar.

- Using serverless architecture so that customers only get charged for what they use.
- Providing an option to the user on whether or not to enable enhanced monitoring (shard-level) for Amazon Kinesis Data Streams. This option is turned off by default to reduce the cost for users who don't need shard-level data monitoring.

Sustainability

This section describes how we architected this solution using the principles and best practices of the sustainability pillar.

The solution utilizes managed and serverless services, to minimize the environmental impact of the backend services. The solution Serverless design (using Lambda, SQS, API Gateway, and S3) and the use of managed services (such as Kinesis Data Streams) are aimed at reducing carbon footprint compared to the footprint of continually operating on-premises servers.

Architecture details

Architecture and component details for all templates.

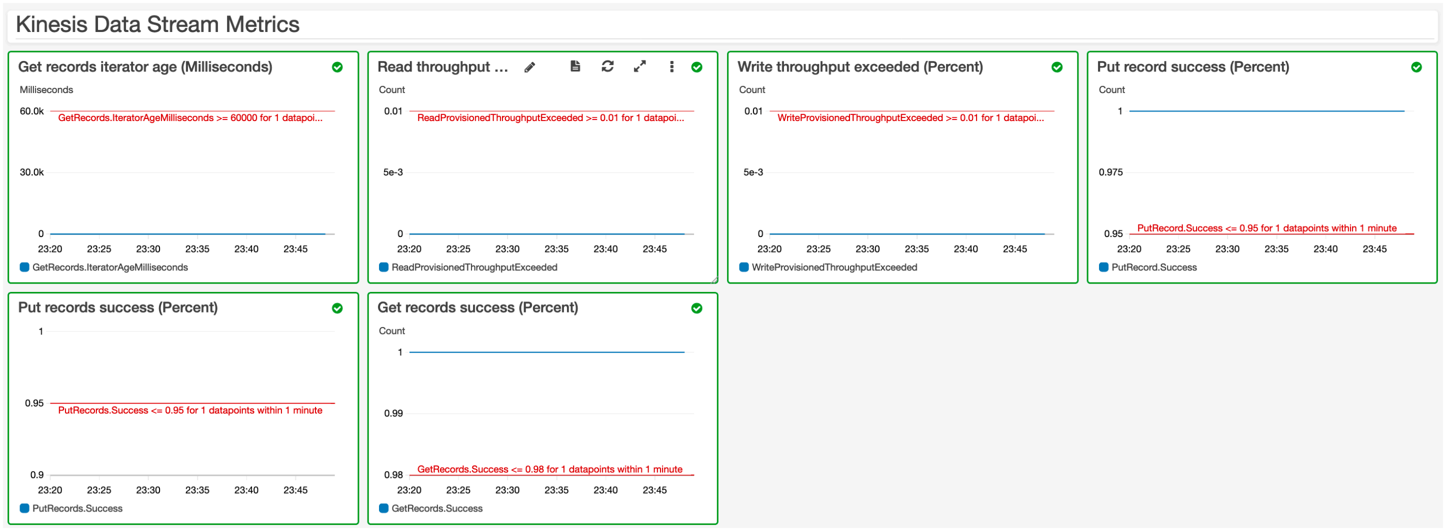
Components for option 1: API Gateway, Kinesis Data Streams, and Lambda

Demo consumer application

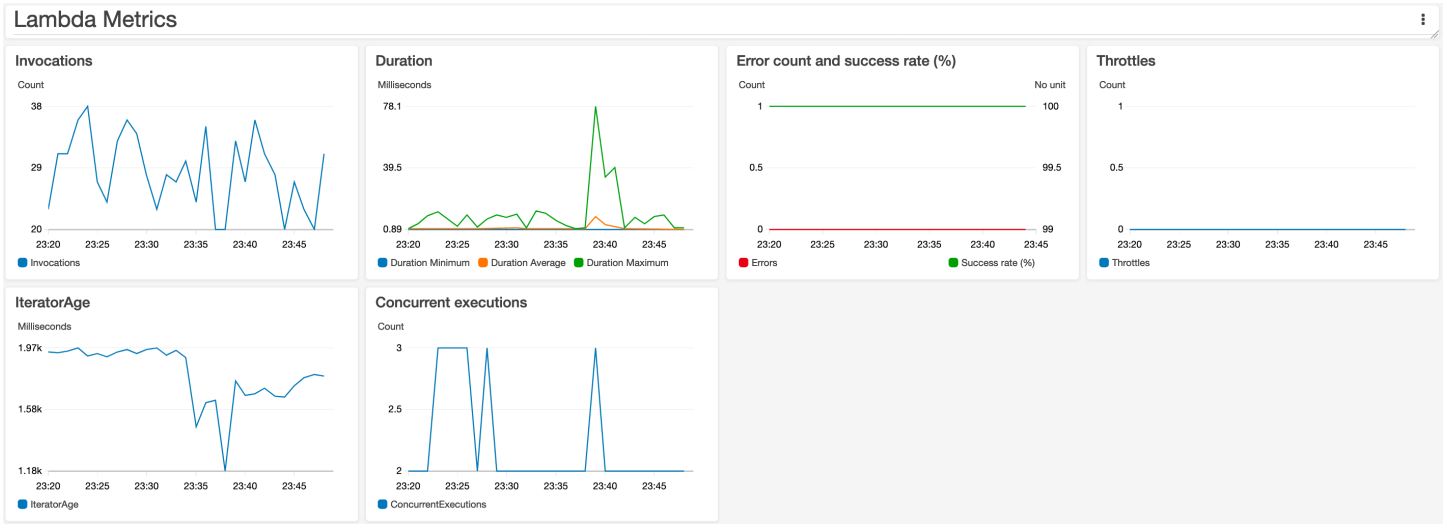
A *consumer* is an application that processes data from a Kinesis data stream. The `aws-streaming-data-solution-for-kinesis-using-api-gateway-and-lambda` template includes a demo consumer application, which is a Node.js function that logs the data being published. The customizable source code is available from the solution's GitHub repository and can be customized to your business needs.

CloudWatch dashboards and alerts

This template deploys an Amazon CloudWatch dashboard that monitors the health of the data stream. You can customize the dashboards and alerts using Amazon CloudWatch or the source code from the solution's GitHub repository.


Kinesis Data Stream Metrics
Get records iterator age (Milliseconds)
Milliseconds
GetRecords.IteratorAgeMilliseconds >= 60000 for 1 datapoi...
GetRecords.IteratorAgeMilliseconds
Read throughput ...
Count
ReadProvisionedThroughputExceeded >= 0.01 for 1 datapoi...
ReadProvisionedThroughputExceeded
Write throughput exceeded (Percent)
Count
WriteProvisionedThroughputExceeded >= 0.01 for 1 datapoi...
WriteProvisionedThroughputExceeded
Put record success (Percent)
Count
PutRecord.Success <= 0.95 for 1 datapoints within 1 minute
PutRecord.Success
Put records success (Percent)
PutRecords.Success <= 0.95 for 1 datapoints within 1 minute
PutRecords.Success
Get records success (Percent)
Count
GetRecords.Success <= 0.98 for 1 datapoints within 1 minute
GetRecords.Success
23:20 23:25 23:30 23:35 23:40 23:45

Kinesis Data Streams metrics on the CloudWatch dashboard



AWS Lambda metrics on the CloudWatch dashboard

Components for option 2: Amazon EC2, KPL, Kinesis Data Streams, and Managed Service for Apache Flink

Demo producer application

A *producer* is an application that puts user data records into a Kinesis data stream (also called data ingestion). The `aws-streaming-data-solution-for-kinesis-using-kpl-and-kinesis-data-analytics` AWS CloudFormation template includes a demo producer application, which is implemented using the Amazon Kinesis Producer Library (KPL). The demo producer application is configured to write 100 records per second to the data stream. The customizable source code is available from the solution's GitHub repository. For information about the customizing the demo producer application, or replacing it with your own application, refer to the KPL demo README.md file in the GitHub repository.

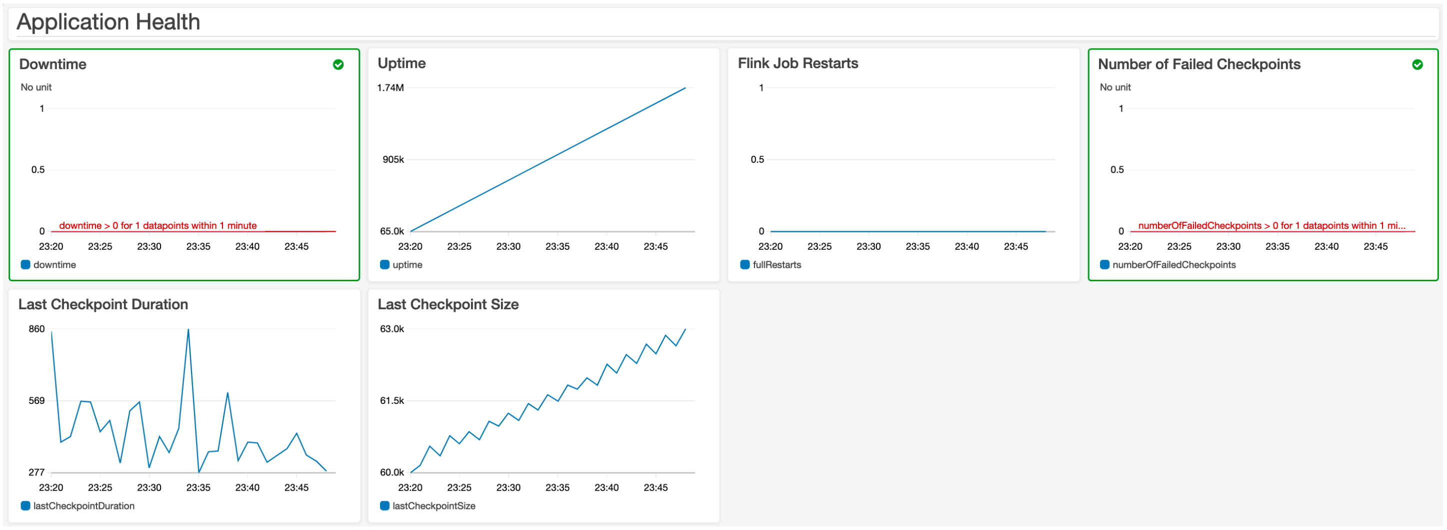
Note

This demo producer application uses the same schema that is provided in Getting Started with Amazon Managed Service for Apache Flink (DataStream API) in the *Amazon Managed Service for Apache Flink Developer Guide*.

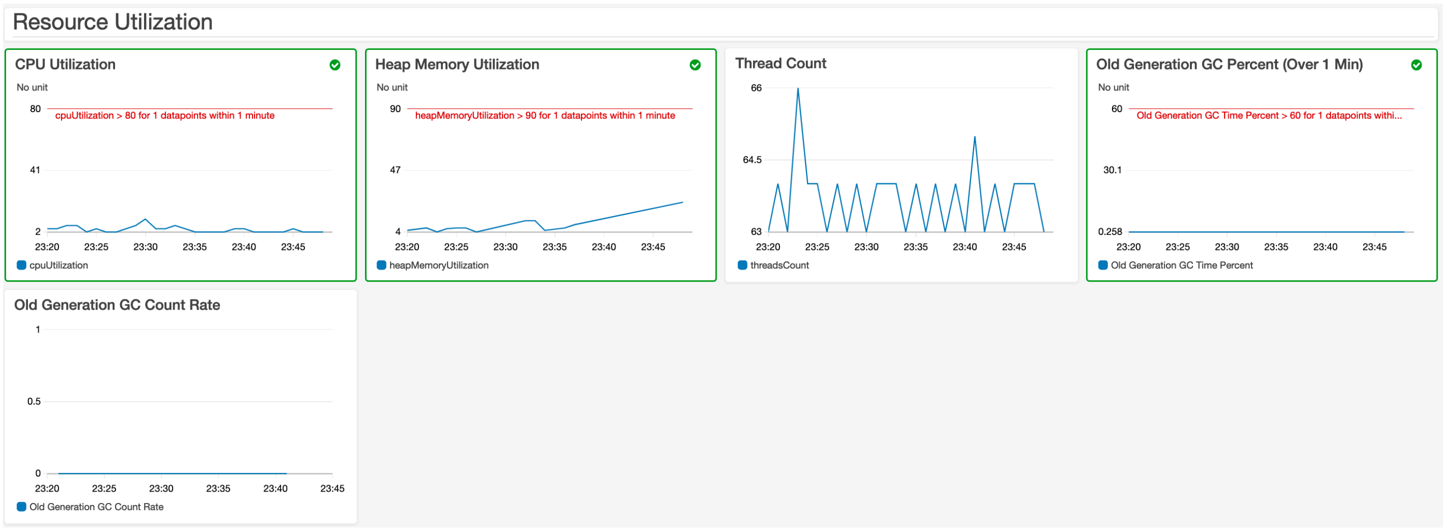
CloudWatch dashbards and alerts

This template deploys an Amazon CloudWatch dashboard that monitors the health of the data stream. You can also view Managed Service for Apache Flink Log Insights statistics.

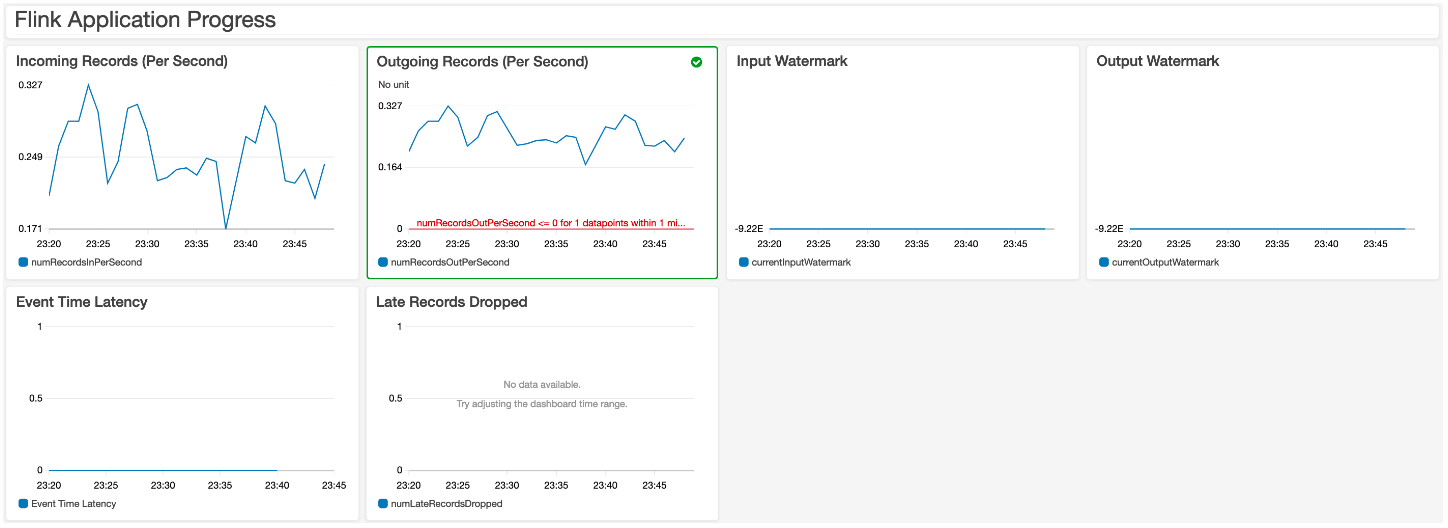
You can customize the dashboards and alerts using Amazon CloudWatch or the source code from the solution’s [GitHub repository](#).



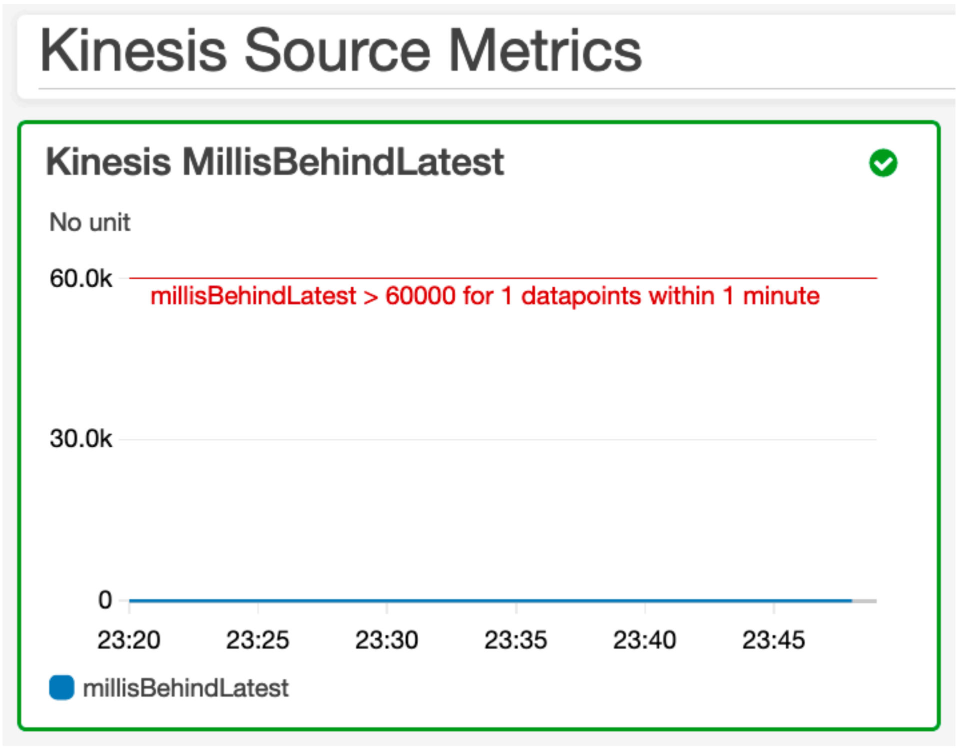
Application Health on the CloudWatch dashboard


Resource Utilization
CPU Utilization
No unit
cpuUtilization > 80 for 1 datapoints within 1 minute
cpuUtilization
Heap Memory Utilization
No unit
heapMemoryUtilization > 90 for 1 datapoints within 1 minute
heapMemoryUtilization
Thread Count
threadsCount
Old Generation GC Percent (Over 1 Min)
No unit
Old Generation GC Time Percent > 60 for 1 datapoints withi...
Old Generation GC Time Percent
Old Generation GC Count Rate
Old Generation GC Count Rate

Resource Utilization on the CloudWatch dashboard



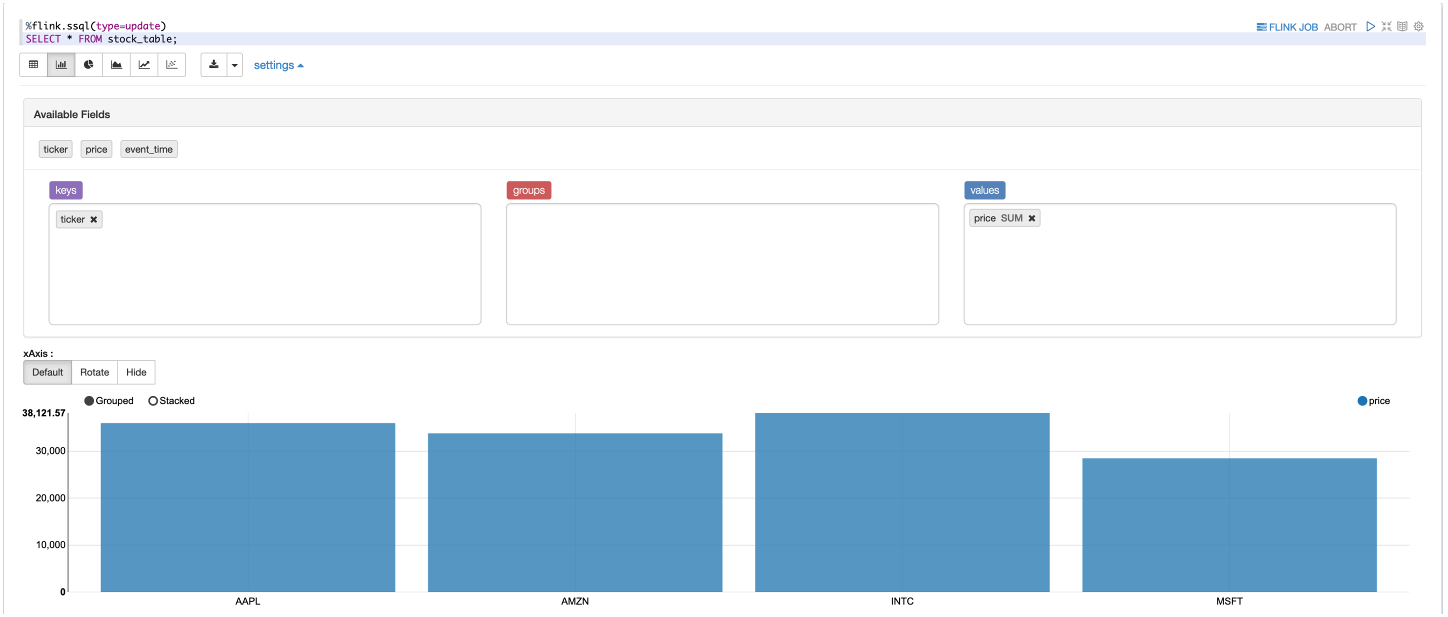
Flink Application Progress on the CloudWatch dashboard



Kinesis Source Metrics on the CloudWatch dashboard

Studio notebook

This template deploys an Amazon Managed Service for Apache Flink Studio notebook powered by Apache Zeppelin and Apache Flink to interactively analyze streaming data.

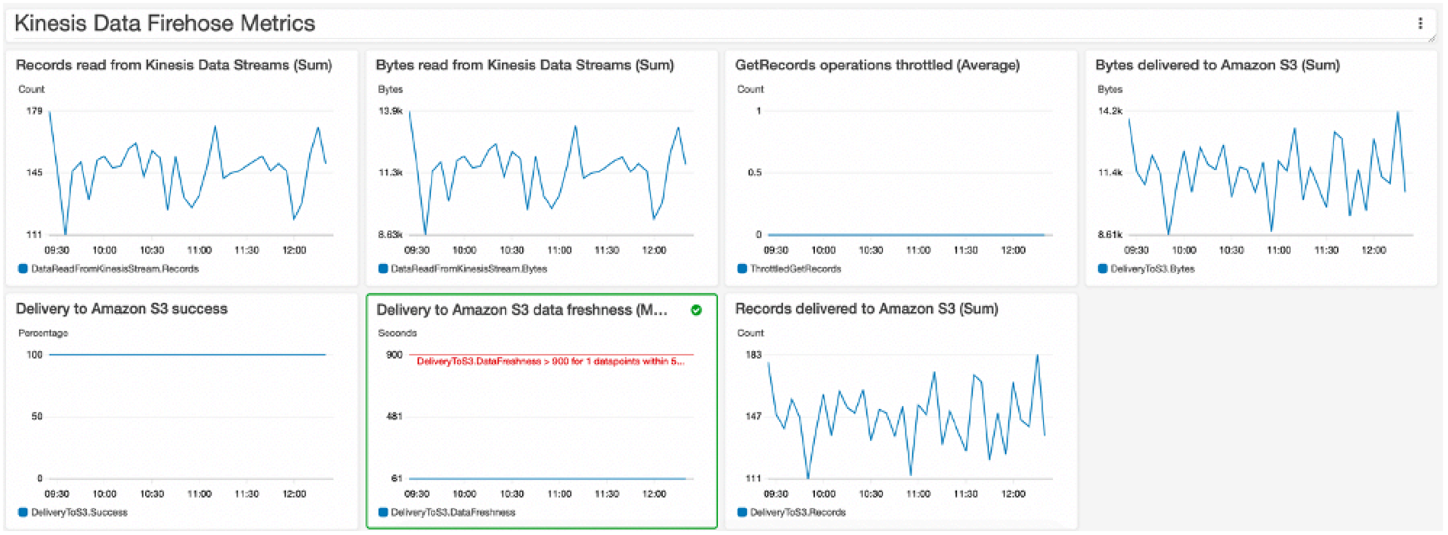

```
%flink.ssql(type=update)
SELECT * FROM stock_table;
```

Example query on the Studio notebook

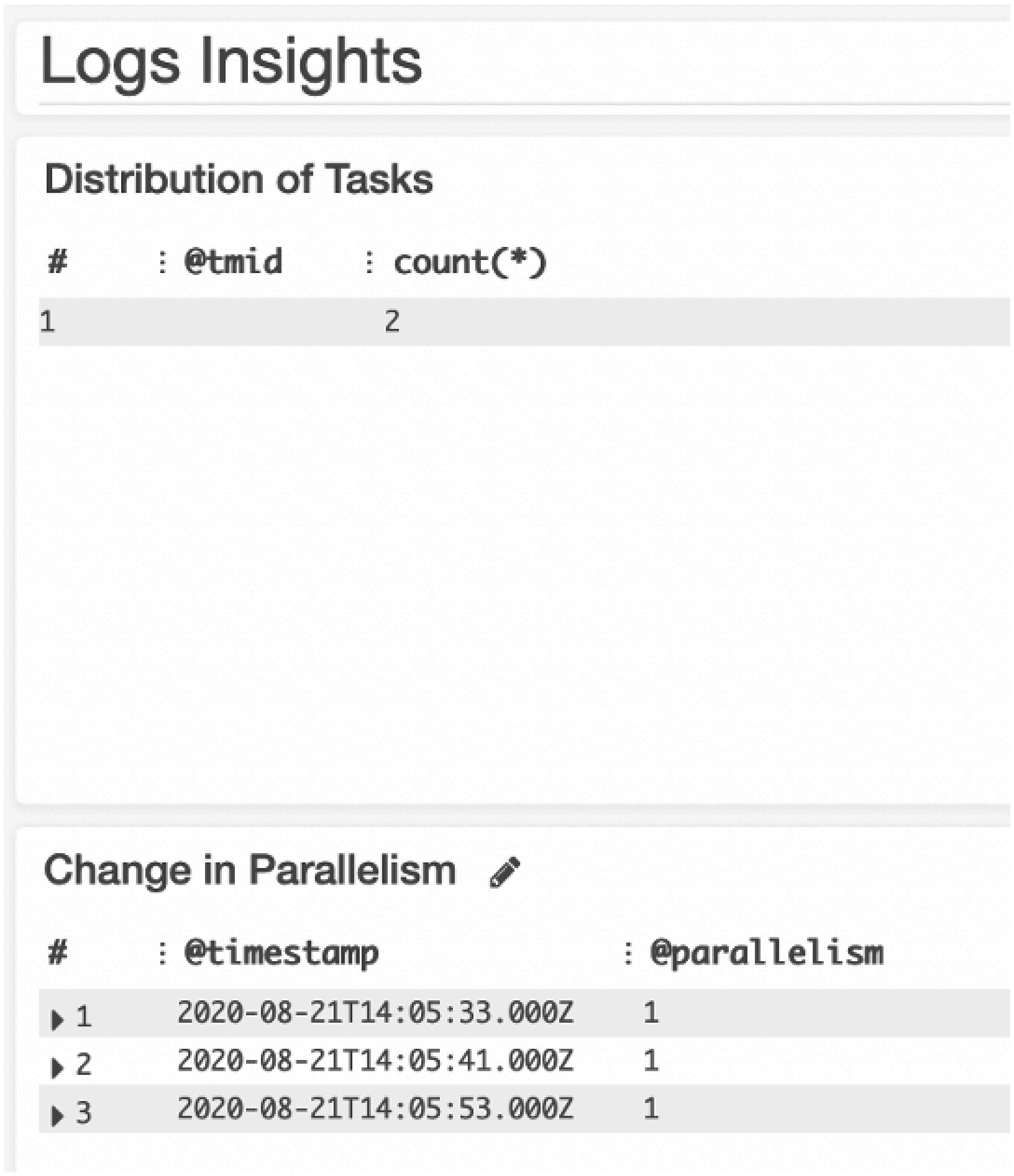
Components for option 3: Kinesis Data Streams, Firehose, and Amazon S3

CloudWatch dashboards and alerts

This template deploys an Amazon CloudWatch dashboard to monitor the health of the data ingestion and buffering. You can customize the dashboards and alerts using Amazon CloudWatch or the source code from the solution's GitHub repository.


Kinesis Data Firehose Metrics
Records read from Kinesis Data Streams (Sum)
Count
DataReadFromKinesisStream.Records
Bytes read from Kinesis Data Streams (Sum)
Bytes
DataReadFromKinesisStream.Bytes
GetRecords operations throttled (Average)
Count
ThrottledGetRecords
Bytes delivered to Amazon S3 (Sum)
Bytes
DeliveryToS3.Bytes
Delivery to Amazon S3 success
Percentage
DeliveryToS3.Success
Delivery to Amazon S3 data freshness (M...
Seconds
DeliveryToS3.DataFreshness > 900 for 1 datapoints within 5...
DeliveryToS3.DataFreshness
Records delivered to Amazon S3 (Sum)
Count
DeliveryToS3.Records
09:30
10:00
10:30
11:00
11:30
12:00

Kinesis Data Firehose metrics on the Amazon CloudWatch dashboard



Logs Insights

Distribution of Tasks

#	@tmid	count(*)
1		2

Change in Parallelism

#	@timestamp	@parallelism
1	2020-08-21T14:05:33.000Z	1
2	2020-08-21T14:05:41.000Z	1
3	2020-08-21T14:05:53.000Z	1

Logs Insights on the CloudWatch dashboard

Components for option 4: Kinesis Data Streams, Managed Service for Apache Flink, and API Gateway

Demo producer application

The `aws-streaming-data-solution-for-kinesis-using-kinesis-data-analytics-and-api-gateway` AWS CloudFormation template includes a demo producer application, that is implemented using the Amazon Kinesis Replay project. By default, the application replays an historic dataset of New York City taxi trips derived from the public data set available from the Registry of Open Data on AWS.

Demo consumer application

This template also includes a demo consumer application, which is a Java application for Amazon Managed Service for Apache Flink. This application demonstrates how to invoke an external API in your streaming application.

CloudWatch dashboards and alerts

This solution deploys an Amazon CloudWatch dashboard to monitor the health, progress, resource utilization, and specific events and errors of the data streaming activities. You can also view Log Insights statistics on the CloudWatch dashboard, which provides information on Managed Service for Apache Flink application logs.

This dashboard visualizations for this template are identical to the `aws-streaming-data-solution-for-kinesis-using-kpl-and-kinesis-data-analytics` template (Option 2).

Custom resources

The AWS CloudFormation templates provided in this solution support enhanced monitoring for Amazon Kinesis Data Streams. When enhanced monitoring is turned on, Kinesis Data Streams sends shard-level data to Amazon CloudWatch. Additional costs may apply.

The `aws-streaming-data-solution-for-kinesis-using-kpl-and-kinesis-data-analytics` and `aws-streaming-data-solution-for-kinesis-using-kinesis-data-analytics-and-api-gateway` AWS CloudFormation templates use the VPC configuration capability in Managed Service for Apache Flink.

Because these features are not currently supported in Amazon Kinesis services, this solution provides AWS Lambda functions that implement custom resources that activate these features.

Cost

You are responsible for the cost of the AWS services used while running this solution. As of this revision, the monthly cost for running this solution with either provided AWS CloudFormation template options, while publishing 100 records per second in the US East (N. Virginia) Region, is described in the following tables.

Prices are subject to change. For full details, refer to the pricing webpage for each AWS service used in this solution.

We recommend creating a budget through AWS Cost Explorer to help manage costs. Prices are subject to change. For full details, refer to the pricing webpage for each AWS service used in this solution.

Sample cost tables

Option 1: Deploy the AWS CloudFormation template using API Gateway, Amazon Kinesis Data Streams, and AWS Lambda

The following table provides a cost estimate to deploy the `aws-streaming-data-solution-for-kinesis-using-api-gateway-and-lambda` AWS CloudFormation template that uses Amazon API Gateway, Amazon Kinesis Data Streams (with enhanced monitoring turned off), and AWS Lambda.

Table for Option 1: Cost estimate for running the solution using the CloudFormation template that deploys API Gateway, Kinesis Data Streams, and AWS Lambda

AWS service	Dimensions	Cost [USD]
Amazon Kinesis Data Streams	1 shard	$10.95
	100 records (4 KB) / second	$3.68
	168 hours data retention	$14.60
Amazon API Gateway	2,678,400 requests / month (1/sec)	$10.50
Amazon Cognito	100 users / month	$5.00

AWS service	Dimensions	Cost [USD]
	Advanced security features added	
AWS Lambda	2,678,400 requests / month (1/sec)	$3.33
	TOTAL:	**$48.06 per month**

Option 2: Deploy the AWS CloudFormation template using Amazon EC2, KPL, Kinesis Data Streams, Managed Service for Apache Flink, and CloudWatch

The following table provides a cost estimate to deploy the `aws-streaming-data-solution-for-kinesis-using-kpl-and-kinesis-data-analytics` AWS CloudFormation template that uses Amazon Elastic Compute Cloud (Amazon EC2), KPL, Kinesis Data Streams, Managed Service for Apache Flink, and Amazon CloudWatch.

Table for Option 2: Cost estimate for running the solution using the CloudFormation template that deploys Amazon EC2, KPL, Kinesis Data Streams, Managed Service for Apache Flink, and CloudWatch

AWS service	Dimensions	Cost [USD]
Amazon Kinesis Producer Library (KPL)	EC2 instance (t3.small) 730 hours / month	$15.18
Kinesis Data Streams	1 shard 100 records (4 KB) / second 168 hours data retention	$10.95 $3.68 $14.60
Managed Service for Apache Flink	1 processing unit	$80.30 $5.00

AWS service	Dimensions	Cost [USD]
	50 GB running application storage	
Amazon S3	1 GB storage (Amazon S3 Standard)	$0.02
	TOTAL:	**$129.73 per month**

Option 3: Deploy the AWS CloudFormation template using Kinesis Data Streams, Firehose, and Amazon S3

The following table provides a cost estimate to deploy the `aws-streaming-data-solution-for-kinesis-using-kinesis-data-firehose-and-amazon-s3` AWS CloudFormation template that uses Amazon Kinesis Data Streams (with enhanced monitoring turned off), Amazon Data Firehose (with data transformation and dynamic partitioning turned off), and Amazon S3.

Table for Option 3: Cost estimate for running the solution using the AWS CloudFormation template that deploys Amazon Kinesis Data Streams, Amazon Data Firehose, and Amazon S3

AWS service	Dimensions	Cost [USD]
Kinesis Data Streams	1 shard	$10.95
	100 records (4KB) / second	$3.68
	168 hours data retention	$14.60
Firehose	100 records (4 KB) / second	$36.34
Amazon S3	1 GB storage (Amazon S3 Standard)	$0.02
	TOTAL:	**$65.59 per month**

Option 4: Deploy the AWS CloudFormation template using Kinesis Data Streams, Managed Service for Apache Flink, and API Gateway

The following table provides a cost estimate to deploy the AWS CloudFormation template that uses Amazon Kinesis Data Streams, Managed Service for Apache Flink, and API Gateway.

Table for Option 4: Cost estimate for running the solution using the AWS CloudFormation template that deploys Amazon Kinesis Data Streams, Managed Service for Apache Flink, and API Gateway

AWS service	Dimensions	Cost [USD]
Amazon Kinesis Replay	EC2 instance (t3.small) 730 hours / month	$15.18
Kinesis Data Streams	1 shard 100 records (4KB) / second 168 hours data retention	$10.95 $3.68 $14.60
Managed Service for Apache Flink	1 processing unit 50 GB running application storage	$80.30 $5.00
API Gateway	2,678,400 requests / month (1/second)	$10.50
	TOTAL:	**$140.21 per month**

Security

When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS. This shared model can reduce your operational burden as AWS operates, manages, and controls the components from the host operating system and virtualization layer down to the physical security of the facilities in which the services operate. For more information about security on AWS, refer to AWS Cloud Security.

IAM roles

AWS Identity and Access Management (IAM) roles enable customers to assign granular access policies and permissions to services and users in the AWS Cloud. This solution creates IAM roles for communication between services. For more information, refer to Providing Access to an AWS Service in the *IAM User Guide*.

Security groups

This solution creates a security group for the Amazon Kinesis Producer Library (KPL) instance so that it can communicate with the Amazon Kinesis endpoint. This security group does not allow any inbound traffic, and the instance can only be accessed via AWS Systems Manager Session Manager.

Auditing

Each AWS service included in this solution is integrated with AWS CloudTrail, which captures all API calls. For more details, refer to the following documentation.

- Logging Managed Service for Apache Flink API Calls with AWS CloudTrail
- Logging calls to Amazon API Gateway APIs with AWS CloudTrail
- Logging AWS Lambda API calls with AWS CloudTrail
- Logging Amazon Kinesis Data Streams API Calls with AWS CloudTrail
- Logging Amazon Data Firehose API Calls with AWS CloudTrail

AWS CloudFormation templates

This solution uses AWS CloudFormation to automate the deployment of the Streaming Data Solution for Amazon Amazon Kinesis in the AWS Cloud. It includes the following CloudFormation templates, which you can download before deployment and customize to meet your needs:

View template

Option

1: streaming-data-solution-for-kinesis-using-api-gateway-and-lambda.template - Use this template to launch this solution using Amazon API Gateway, Amazon Kinesis Data Streams, and AWS Lambda. The default configuration deploys API Gateway REST APIs, Kinesis Video Streams, AWS Lambda functions, and an Amazon Simple Queue Service queue. You can also customize the template based on your specific needs.

View template

Option

2: streaming-data-solution-for-kinesis-using-kpl-and-kinesis-data-analytics.template - Use this template to launch this solution using Kinesis Producer Library, Amazon Elastic Compute Cloud (Amazon EC2), and Amazon Managed Service for Apache Flink. The default configuration deploys an Amazon EC2 instance, Kinesis Data Streams, Managed Service for Apache Flink, Amazon CloudWatch, and an Amazon Simple Storage Service bucket. You can also customize the template based on your specific needs.

View template

Option

3: streaming-data-solution-for-kinesis-using-kinesis-data-firehose-and-amazon-s3.template - Use this template to launch the solution using Kinesis Data Streams, Kinesis Data Firehose, and S3. You can also customize the template for your specific needs.

View template

Option

4: streaming-data-solution-for-kinesis-using-kinesis-data-analytics-and-api-gateway.template - Use this template to launch the solution using Kinesis Data Streams, Managed Service for Apache Flink, and API Gateway. You can also customize the template for your specific needs.

Deploy the solution

Prerequisites

Choose one of the following AWS CloudFormation templates to deploy, then follow the step-by-step instructions for your selected template:

- **Option 1:** Deploy the `streaming-data-solution-for-kinesis-using-api-gateway-and-lambda.template` AWS CloudFormation template using Amazon API Gateway, Amazon Kinesis Data Streams, and AWS Lambda
- **Option 2:** Deploy the `streaming-data-solution-for-kinesis-using-kpl-and-kinesis-data-analytics.template` AWS CloudFormation template using Amazon Elastic Compute Cloud (Amazon EC2), Amazon Kinesis Producer Library (KPL), Kinesis Data Streams, Amazon Managed Service for Apache Flink, and Amazon CloudWatch
- **Option 3:** Deploy the `streaming-data-solution-for-kinesis-using-kinesis-data-firehose-and-amazon-s3.template` AWS CloudFormation template using Kinesis Data Streams, Kinesis Data Firehose, and S3
- **Option 4:** Deploy the `streaming-data-solution-for-kinesis-using-kinesis-data-analytics-and-api-gateway.template` AWS CloudFormation template using Kinesis Data Streams, Managed Service for Apache Flink, and Amazon API Gateway

Option 1: Deploy the aws-streaming-data-solution-for-kinesis-using-api-gateway-and-lambda CloudFormation template

Before you launch this template, review the architecture and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy the solution into your account.

Time to deploy: Approximately five minutes

Launch the Stack

Note

You are responsible for the cost of the AWS services used while running this solution. Refer to the Cost section for more details. For full details, refer to the pricing webpage for each AWS service used in this solution.

1. Sign in to the AWS Management Console and use the button below to launch the `streaming-data-solution-for-kinesis-using-api-gateway-and-lambda` AWS CloudFormation template.

   Launch solution

   Alternatively, you can download the template as a starting point for your own implementation.
2. The template launches in the US East (N. Virginia) Region by default. To launch this solution in a different AWS Region, use the Region selector in the console navigation bar.
3. On the **Create stack** page, verify that the correct template URL shows in the **Amazon S3 URL** text box and choose **Next**.
4. On the **Specify stack details** page, assign a name to your solution stack. For informationabout naming character limitations, refer to IAM and STS Limits in the *AWS Identity and Access Management User Guide*.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following default values.

Parameter	Default	Description
Amazon API Gateway configuration		
Steady-state requests per second **(ThrottlingRateLimit)**	`100`	The number of steady-state requests per second that the proxy API Gateway permits.

Parameter	Default	Description
		The allowed range is from 1 to `10000`.
Burst requests per second **(ThrottlingBurstLimit)**	`50`	The number of burst requests per second that the proxy API Gateway permits. The allowed range is from `0` to `5000`.
Amazon Kinesis Data Streams configuration		
Number of open shards **(ShardCount)**	`2`	The number of shards that the stream uses. The allowed range is from `1` to `100` shards.
Data retention period (hours) **(RetentionHours)**	`24`	The number of hours that data records stored in shards will remain accessible. The allowed range is from `24` to `8760` hours.
Enable enhanced (shard-level) metrics **(EnableEnhancedMonitoring)**	`false`	Choose whether to activate enhanced monitoring for shard-level metrics. This function is turned off by default.
AWS Lambda consumer configuration		
Largest number of records that will be read from the stream at once **(BatchSize)**	`100`	The maximum number of records to retrieve in a single batch. The allowed range is from `1` to `10000`.

Parameter	Default	Description
Number of batches to process from each shard concurrently **(ParallelizationFactor)**	1	The number of batches to process from each shard concurrently. The allowed range is from `1` to `10`.
Maximum number of times to retry when the function returns an error **(MaxRetryAttempts)**	3	The maximum number of retries when the AWS Lambda function returns an error.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review** page, review and confirm the settings. Check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.
9. Choose **Create stack** to deploy the stack.

   You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should receive a **CREATE_COMPLETE** status in approximately five minutes.

You can customize or replace the demo application that is included with this solution can be customized or replaced to meet your business needs. The source code is available from the solution's GitHub repository. For information about the demo producer application and customizing the demo application or replacing it with your own application, refer to the README.md file in the GitHub repository.

Note

This solution includes the `solution-helper` Lambda function, which runs only during initial configuration or when resources are updated or deleted. When you run this solution, you will notice the Lambda functions in the AWS Management Console. While it may not appear active, do not delete the `solution-helper` function because it is necessary to manage associated resources.

Option 2: Deploy the aws-streaming-data-solution-for-kinesis-using-kpl-and-kinesis-data-analytics CloudFormation template

Before you launch this template, review the architecture and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy the solution into your account.

Time to deploy: Approximately 10 minutes

Deployment overview

Use the following steps to deploy this solution on AWS. For detailed instructions, follow the links for each step.

Step 1. Launch the Stack

- Launch the AWS CloudFormation template into your AWS account.
- Review the other template parameters, and adjust if necessary.

Step 2. Post-configuration steps

Step 1. Launch the Stack

Note

You are responsible for the cost of the AWS services used while running this solution. Refer to the Cost section for more details. For full details, refer to the pricing webpage for each AWS service used in this solution.

1. Sign in to the AWS Management Console and use the button below to launch the `streaming-data-solution-for-kinesis-using-kpl-and-kinesis-data-analytics` AWS CloudFormation template.

Launch solution

Alternatively, you can download the template as a starting point for your own implementation.

2. The template launches in the US East (N. Virginia) Region by default. To launch this solution in a different AWS Region, use the Region selector in the console navigation bar.

> **Note**
>
> This template uses Amazon Managed Service for Apache Flink, which is not currently available in all AWS Regions. You must launch this solution in an AWS Region where Managed Service for Apache Flink is available. For the most current availability by Region, refer to the AWS Service Region Table.

3. On the **Create stack** page, verify that the correct template URL shows in the **Amazon S3 URL** text box and choose **Next**.

4. On the **Specify stack details** page, assign a name to your solution stack. For information about naming character limitations, refer to IAM and STS Limits in the *AWS Identity and Access Management User Guide*.

5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following default values.

Parameter	Default	Description
Amazon Kinesis Producer Library (KPL) configuration		
VPC where the KPL instance should be launched **(ProducerVpcId)**	*`<Requires input>`*	VPC where the KPL instance is launched.
Subnet where the KPL instance should be launched **(ProducerSubnetId)**	*`<Requires input>`*	Subnet where the KPL instance is launched (the subnet requires access to Kinesis Data Streams, either via IGW or NAT).

Parameter	Default	Description
Amazon Machine Image for the KPL instance **(ProducerAmId)**	`/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2`	Amazon Machine Image (AMI) ID for the KPL instance.
Amazon Kinesis Data Streams configuration		
Number of open shards **(ShardCount)**	`2`	The number of shards that the stream uses. The allowed range is from `1` to `100` shards.
Data retention period (hours) **(RetentionHours)**	`24`	The number of hours that data records stored in shards will remain accessible. The allowed range is from `24` to `8760` hours.
Enable enhanced (shard-level) metrics **(EnableEnhancedMonitoring)**	`false`	Choose whether to activate enhanced monitoring for shard-level metrics. This function is turned off by default.
Amazon Managed Service for Apache Flink configuration		

Parameter	Default	Description
Monitoring log level **(LogLevel)**	`INFO`	The level of detail of the CloudWatch Logs for an application. The available options include `DEBUG`, `ERROR`, `INFO`, and `WARN`. For information about choosing a log level, refer to Application Monitoring Levels in the *Amazon Managed Service for Apache Flink Developer Guide*.
Comma-separated list of subnet ids for VPC connectivity **(ApplicationSubnetIds)**	<Optional input>	If subnet IDs are provided, then security groups must also be included.
Comma-separated list of security groups ids for VPC connectivity **(ApplicationSecurityGroupIds)**	<Optional input>	If security group IDs are provided, then subnets must also be included.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review** page, review and confirm the settings. Check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.
9. Choose **Create stack** to deploy the stack.

   You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should receive a **CREATE_COMPLETE** status in approximately 10 minutes.

Note

This solution includes the `solution-helper` Lambda function, which runs only during initial configuration. This function is only created if you start the collection of operational metrics.

Step 2. Post-configuration steps

By default, the demo producer and Studio notebook will not run after the stacks are created. Follow these steps to enable them.

1. Sign in to the Amazon Kinesis console and, from the left menu pane, select **Analytics applications**.
2. On the **Amazon Managed Service for Apache Flink** page, go to the **Studio** tab, and select **Kda*<studio-notebook-name>***.
3. Choose **Actions** then choose **Run application**.
4. Navigate to the AWS Systems Manager console and, from the left menu pane under **Instances and Nodes**, select **Session Manager**.
5. On the **AWS Systems Manager** page, choose **Start session**.
6. On the **Start a session** page, select the *ec2-instance-id>* for the KPL instance and choose **Start session**.

   Refer to the AWS CloudFormation **Outputs** tab for the Amazon EC2 instance ID.
7. In the console window, run the following command to start the demo producer application. (Replace *<stream-name>*, *<aws-region>*, and *<seconds>* with your specific information).

```
sudo java -jar /tmp/aws-kpl-demo.jar <stream-name> <aws-region> <seconds>
```

You can customize or replace the demo application that is included with this solution can be customized or replaced to meet your business needs. The source code is available from the solution's GitHub repository. For information about the demo producer application and customizing the demo application or replacing it with your own application, refer to the `README.md` file in the GitHub repository.

Option 3: Deploy the aws-streaming-data-solution-for-kinesis-using-kinesis-data-firehose-and-amazon-s3 CloudFormation template

Before you launch this template, review the architecture and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy the solution into your account.

Time to deploy: Approximately five minutes

Launch the Stack

Note

You are responsible for the cost of the AWS services used while running this solution. Refer to the Cost section for more details. For full details, refer to the pricing webpage for each AWS service used in this solution.

1. Sign in to the AWS Management Console and use the button below to launch the `streaming-data-solution-for-kinesis-using-kinesis-data-firehose-and-amazon-s3` AWS CloudFormation template.

   Launch solution

   Alternatively, you can download the template as a starting point for your own implementation.
2. The template launches in the US East (N. Virginia) Region by default. To launch this solution in a different AWS Region, use the Region selector in the console navigation bar.
3. On the **Create stack** page, verify that the correct template URL shows in the **Amazon S3 URL** text box and choose **Next**.
4. On the **Specify stack details** page, assign a name to your solution stack. For information about naming character limitations, refer to IAM and STS Limits in the *AWS Identity and Access Management User Guide*.

5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following default values.

Parameter	Default	Description
Amazon Kinesis Data Streams configuration		
Number of open shards **(ShardCount)**	`2`	The number of shards that the stream uses. The allowed range is from `1` to `100` shards.
Data retention period **(RetentionHours)**	`24`	The number of hours that data records stored in shards will remain accessible. The allowed range is from `24` to `8760` hours.
Enable enhanced (shard-level) metrics **(EnableEnhancedMonitoring)**	`false`	Choose whether to activate enhanced monitoring for shard-level metrics. This function is turned off by default.
Amazon Kinesis Data Firehose configuration		
Size of the buffer (in MBs) that incoming data is buffered before delivery **(BufferingSize)**	`5`	The size to buffer incoming data before delivering to S3. The allowed range is from `1` to `128`.
Length of time (in seconds) that incoming data is buffered before delivery **(BufferingInterval)**	`300`	The amount of time to buffer incoming data before delivering to S3. The allowed range is from `60` to `900`.

Parameter	Default	Description
Compression format for delivered data in Amazon S3 **(CompressionFormat)**	`GZIP`	The format of data once it's delivered to S3. Allowed values are GZIP, HADOOP_SNAPPY, Snappy, UNCOMPRESSED, and ZIP.
Data prefix for delivered data in S3 **(DataPrefix)**	*`<Optional input>`*	Prefix to be appended to the data delivered to S3 (if dynamic partitioning is activated, you can also specify the `partition KeyFromQuery` namespace).
Prefix for failed records that cannot be delivered **(ErrorsPrefix)**	*`<Optional input>`*	Prefix to be used for errors when delivering data (if dynamic partitioning is activated, this parameter is required).
Dynamic partitioning configuration		
Whether data on Amazon S3 will be partitioned **(DynamicPartitioning)**	`Disabled`	Whether dynamic partition ing should be activated (once activated, this feature cannot be deactivated). Allowed values are Disabled and Enabled.
New line delimiter **(NewLineDelimiter)**	`Disabled`	Whether to add a new line delimiter between records. Allowed values are Disabled and Enabled.

Parameter	Default	Description
JQ expression **(JqExpression)**	`<Optional input>`	JQ expression to be used for inline parsing (for instance, "{ ticker: .ticker }").
Retry duration **(RetryDurationSec)**	`300`	Total amount of time (in seconds) that should be spent on retries. The allowed range is from `0` to `7200`.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review** page, review and confirm the settings. Check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.
9. Choose **Create stack** to deploy the stack.

   You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should receive a **CREATE_COMPLETE** status in approximately five minutes.

Note

This solution includes the `solution-helper` Lambda function, which runs only during initial configuration or when resources are updated or deleted. When you run this solution, you will notice the Lambda functions in the AWS Management Console. While it may not appear active, do not delete the `solution-helper` function because it is necessary to manage associated resources.

Option 4: Deploy the aws-streaming-data-solution-for-kinesis-using-kinesis-data-analytics-and-api-gateway CloudFormation template

Before you launch this template, review the architecture and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy the solution into your account.

Time to deploy: Approximately 10 minutes

Deployment overview

Use the following steps to deploy this solution on AWS. For detailed instructions, follow the links for each step.

Step 1. Launch the Stack

- Launch the AWS CloudFormation template into your AWS account.
- Review the other template parameters, and adjust if needed.

Step 2. Post-configuration steps

Step 1. Launch the Stack

Note

You are responsible for the cost of the AWS services used while running this solution. Refer to the Cost section for more details. For full details, refer to the pricing webpage for each AWS service used in this solution.

1. Sign in to the AWS Management Console and use the button below to launch the `streaming-data-solution-for-kinesis-using-kinesis-data-analytics-and-api-gateway` AWS CloudFormation template.

   Launch solution

Alternatively, you can download the template as a starting point for your own implementation.

2. The template launches in the US East (N. Virginia) Region by default. To launch this solution in a different AWS Region, use the Region selector in the console navigation bar.

> **Note**
>
> This template uses Amazon Managed Service for Apache Flink, which is not currently available in all AWS Regions. You must launch this solution in an AWS Region where Managed Service for Apache Flink is available. For the most current availability by Region, refer to the AWS Service Region Table.

3. On the **Create stack** page, verify that the correct template URL shows in the **Amazon S3 URL** text box and choose **Next**.

4. On the **Specify stack details** page, assign a name to your solution stack. For information about naming character limitations, refer to IAM and STS Limits in the *AWS Identity and Access Management User Guide*.

5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following default values.

Parameter	Default	Description
Amazon Kinesis Replay configuration		
VPC where the KPL instance should be launched **(ProducerVpcId)**	*<Requires input>*	VPC where the KPL instance is launched.
Subnet where the KPL instance should be launched **(ProducerSubnetId)**	*<Requires input>*	Subnet where the KPL instance is launched (the subnet requires access to Kinesis Data Streams, either via IGW or NAT).

Parameter	Default	Description
Amazon Machine Image for the KPL instance **(ProducerAmId)**	`/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2`	Amazon Machine Image (AMI) ID for the KPL instance.
Amazon Kinesis Data Streams configuration		
Number of open shards **(ShardCount)**	`2`	The number of shards that the stream uses. The allowed range is from `1` to `100` shards.
Data retention period (hours) **(RetentionHours)**	`24`	The number of hours that data records stored in shards will remain accessible. The allowed range is from `24` to `8760` hours.
Enable enhanced (shard-level) metrics **(EnableEnhancedMonitoring)**	`false`	Choose whether to activate enhanced monitoring for shard-level metrics. This function is deactivated by default.
Amazon Managed Service for Apache Flink configuration		
Monitoring log level **(LogLevel)**	`INFO`	The level of detail of the CloudWatch Logs for an application. The available options include `DEBUG`, `ERROR`, `INFO`, and `WARN`. For information about choosing a log level, refer to Application Monitoring Levels in the *Amazon Kinesis Data Analytics Developer Guide*.

Parameter	Default	Description
Comma-separated list of subnet ids for VPC connectivity **(ApplicationSubnetIds)**	<Optional input>	If subnet IDs are provided, then security groups must also be included.
Comma-separated list of security groups ids for VPC connectivity **(ApplicationSecuri tyGroupIds)**	<Optional input>	If security group IDs are provided, then subnets must also be included.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review** page, review and confirm the settings. Check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.
9. Choose **Create stack** to deploy the stack.

   You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should receive a **CREATE_COMPLETE** status in approximately 10 minutes.

Note

This solution includes the `solution-helper` Lambda function, which runs only during initial configuration or when resources are updated or deleted. When you run this solution, you will notice the Lambda functions in the AWS Management Console. While it may not appear active, do not delete the `solution-helper` function because it is necessary to manage associated resources.

Step 2. Post-configuration steps

By default, the demo producer and demo consumer applications will not run after the stacks are created. Use the following process to start them.

1. Sign in to the Amazon Kinesis console and, from the left menu pane, select **Analytics applications**.
2. On the **Amazon Managed Service for Apache Flink** page, select **Kda** *<application-name>*.
3. Choose **Actions** then choose **Run application**.
4. Navigate to the AWS Systems Manager console and, from the left menu pane under **Instances and Nodes**, select **Session Manager**.
5. On the **AWS Systems Manager** page, choose **Start session**.
6. On the **Start a session** page, select the ***<ec2-instance-id>*** for the KPL instance and choose **Start session**.

   Refer to the AWS CloudFormation **Outputs** tab for the Amazon EC2 instance ID.
7. In the console window, run the following command to start the demo producer application.

   (Replace *<stream-name>*, *<aws-region>*, and *<seconds>* with your information.

```
sudo java -jar /tmp/amazon-kinesis-replay-0.1.0.jar -streamName <stream-name> -
streamRegion <region> -noWatermark -objectPrefix artifacts/kinesis-analytics-taxi-
consumer/taxi-trips-partitioned.json.lz4/dropoff_year=2018/
```

   Note

   The demo application uses an updated schema of the taxi dataset. You must specify a custom objectPrefix when running the preceeding command.

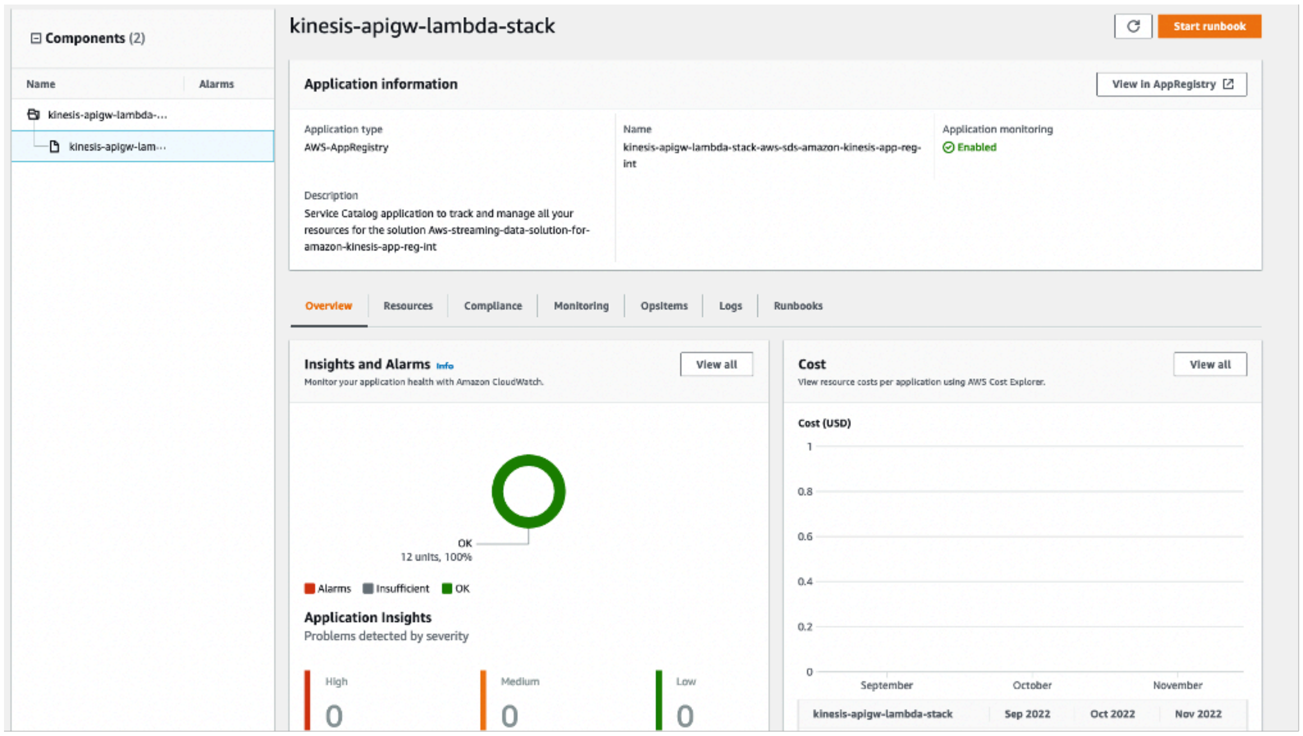
Monitoring this solution with AWS Service Catalog AppRegistry

AWS Streaming Data Solution for Amazon Kinesis includes a Service Catalog AppRegistry resource to register the CloudFormation template and underlying resources as an application in both AWS Service Catalog AppRegistry and AWS Systems Manager Application Manager.

AWS Systems ManagerApplication Manager gives you an application-level view into this solution and its resources so that you can:

- Monitor its resources, costs for the deployed resources across stacks and AWS accounts, and logs associated with this solution from a central location.
- View operations data for the resources of this solution in the context of an application, such as deployment status, CloudWatch alarms, resource configurations, and operational issues.

The following figure depicts an example of the application view for the AWS Streaming Data Solution for Amazon Kinesis stack in Application Manager.



Application type
AWS-AppRegistry

Name
kinesis-apigw-lambda-stack-aws-sds-amazon-kinesis-app-reg-int

Application monitoring
Enabled

Description
Service Catalog application to track and manage all your resources for the solution Aws-streaming-data-solution-for-amazon-kinesis-app-reg-int

kinesis-apigw-lambda-stack	Sep 2022	Oct 2022	Nov 2022

AWS Streaming Data Solution for Amazon Kinesis Application Manager

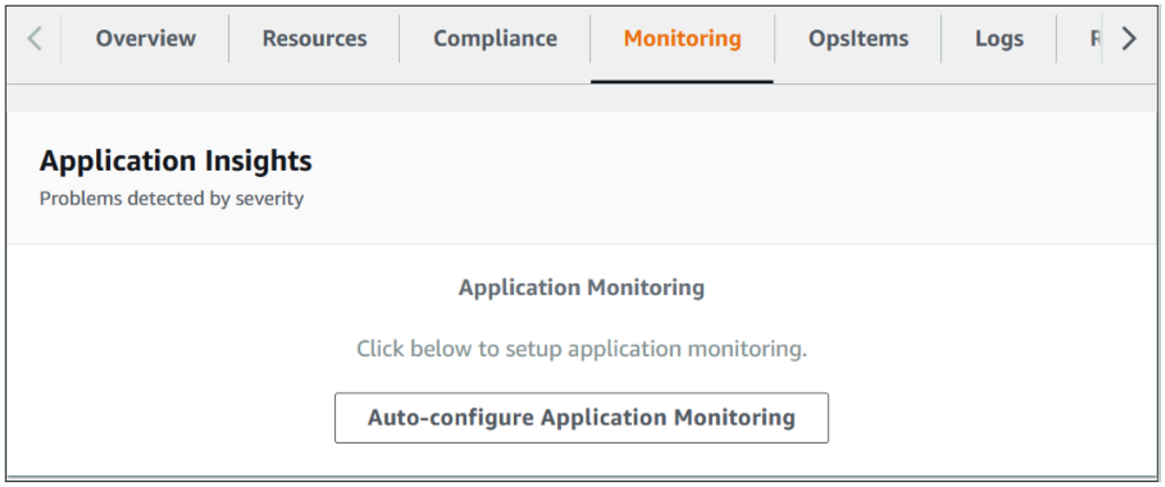
Note

AWS Cost Explorer must be activated. It is not activated by default.

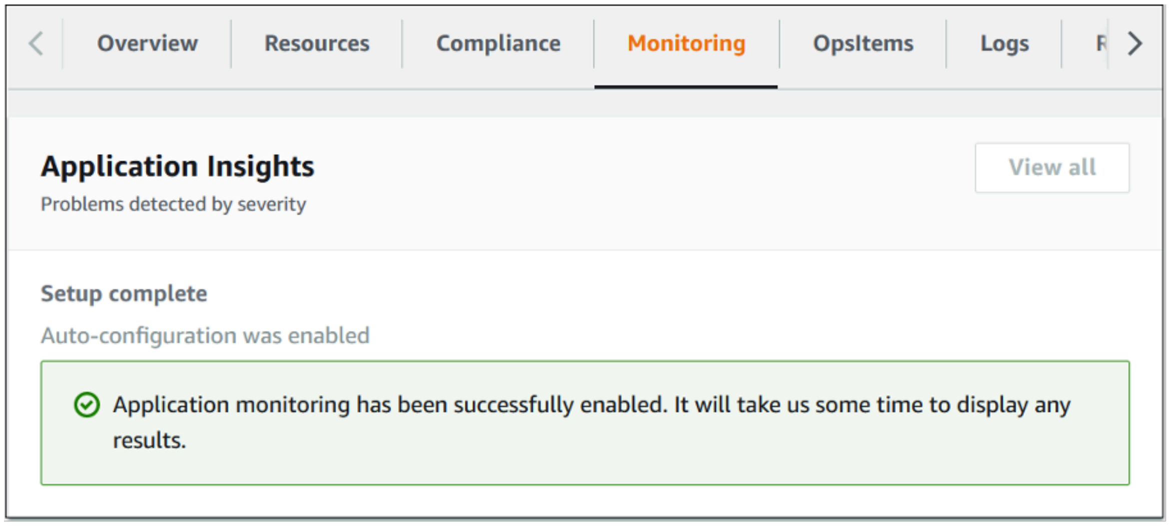
Activate CloudWatch Application Insights

1. Sign in to the Systems Manager console.
2. In the navigation pane, choose **Application Manager**.
3. In **Applications**, choose **AppRegistry applications**.
4. In **AppRegistry applications**, search for the application name for this solution and select it.

   The next time you open Application Manager, you can find the new application for your solution in the **AppRegistry application** category.
5. In the **Components** tree, choose the application stack you want to activate.
6. In the **Monitoring** tab, in **Application Insights**, select **Auto-configure Application Monitoring**.


Overview
Resources
Compliance
Monitoring
OpsItems
Logs
Application Insights
Problems detected by severity
Application Monitoring
Click below to setup application monitoring.
Auto-configure Application Monitoring

Monitoring for your applications is now activated and the following status box appears:



Application Insights

Problems detected by severity

Setup complete

Auto-configuration was enabled

Application monitoring has been successfully enabled. It will take us some time to display any results.

Activate AWS Cost Explorer

Note

This section is only required if AWS Cost Explorer is not already enabled for your AWS Account.

You can see the overview of the costs associated with the application and application components within the Application Manager console through integration with AWS Cost Explorer which must be first activated. Cost Explorer helps you manage costs by providing a view of your AWS resource costs and usage over time. To activate Cost Explorer for the solution:

1. Sign in to the AWS Cost Management console.
2. In the navigation pane, select **Cost Explorer**.
3. On the **Welcome to Cost Explorer** page, choose **Launch Cost Explorer**.

The activation process can take up to 24 hours to complete. Once activated, you can open the Cost Explorer user interface to further analyze cost data for the solution.

Activate cost allocation tags associated with the solution

After you activate Cost Explorer, you must activate a cost allocation tag to see the costs for this solution. The cost allocation tags can only be activated from the management account for the organization. To activate cost allocation tags:

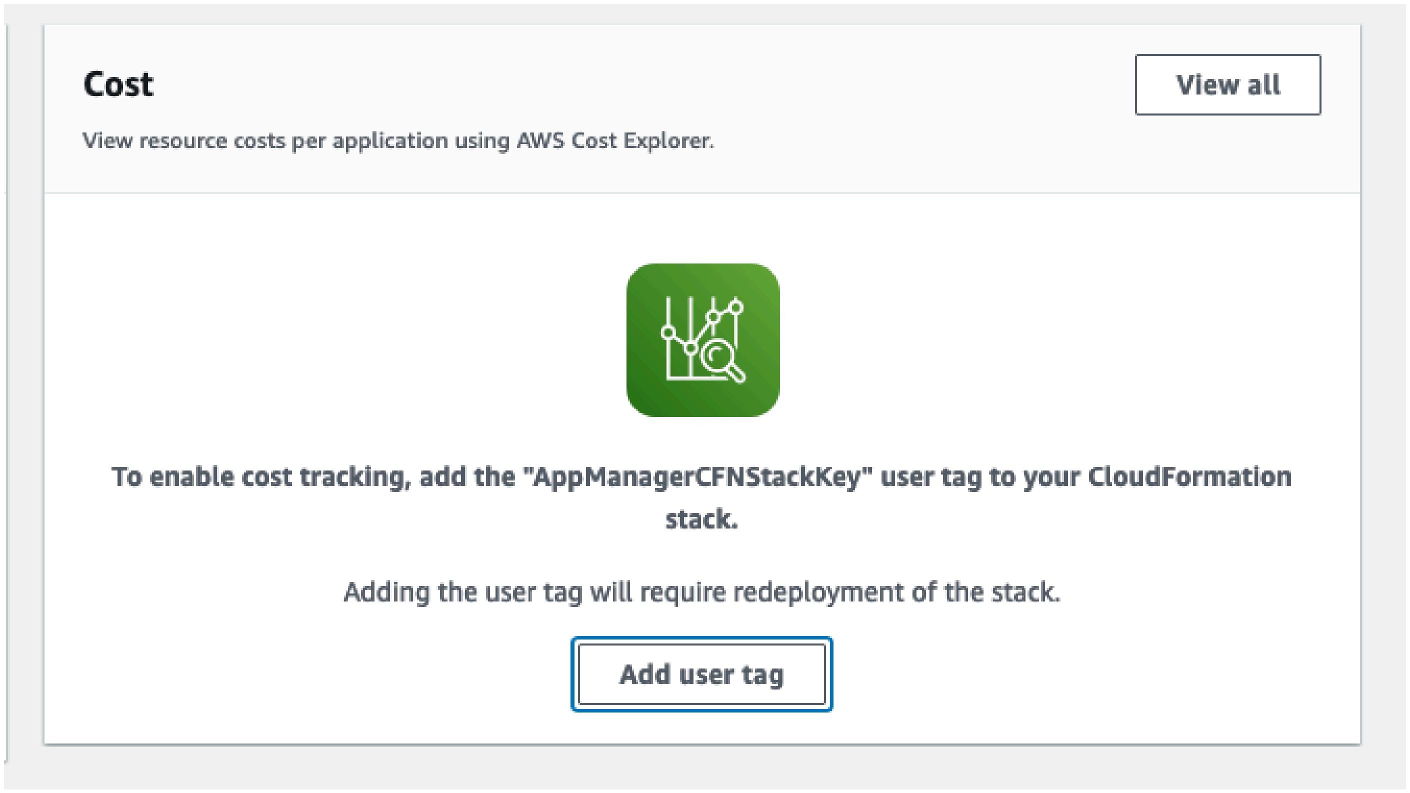
1. Sign in to the AWS Billing and Cost Management console.
2. In the navigation pane, select **Cost Allocation Tags**.
3. On the Cost allocation tags page, filter for the `AppManagerCFNStackKey` tag, then select the tag from the results shown.
4. Choose **Activate**

The activation process can take up to 24 hours to complete and the tag data to appear.

Confirm cost tags associated with the solution

After you activate cost allocation tags associated with the solution, you must confirm the cost allocation tags to see the costs for this solution. To confirm cost allocation tags:

1. Sign in to the Systems Manager console.
2. In the navigation pane, choose **Application Manager**.
3. In **Applications**, choose the application name for this solution and select it.
4. In the **Overview** tab, in **Cost**, select **Add user tag**.



Cost

View resource costs per application using AWS Cost Explorer.

To enable cost tracking, add the "AppManagerCFNStackKey" user tag to your CloudFormation stack.

Adding the user tag will require redeployment of the stack.

5. On the **Add user tag** page, enter `confirm`, then select **Add user tag**.

The activation process can take up to 24 hours to complete and the tag data to appear.

Additional resources

- Amazon API Gateway
- Amazon CloudWatch
- Amazon Elastic Compute Cloud
- Amazon Managed Service for Apache Flink
- Amazon Kinesis Data Streams
- Amazon Data Firehose
- Amazon Simple Storage Service
- AWS CloudFormation
- AWS Identity and Access Management
- AWS Lambda
- Amazon Simple Queue Service
- AWS Systems Manager

AWS documentation

Developing Producers Using the Amazon Kinesis Producer Library

Best practices for monitoring and data protection:

- Viewing Managed Service for Apache Flink Metrics and Dimensions
- Using CloudWatch Alarms with Amazon Managed Service for Apache Flink for Apache Flink
- Security in Amazon Managed Service for Apache Flink
- Security in Amazon Kinesis Data Streams
- Using AWS Lambda with Amazon Kinesis
- Monitoring Firehose Using CloudWatch Metrics
- Security in Amazon Data Firehose

Kinesis Producer Library

This solution's CloudFormation resources are created from AWS CDK components, and the resources contain the template code that you can use to customize this solution. For more information about setting up your project and customizing this solution, refer to the README.md file in GitHub.

The Amazon Kinesis Producer Library (KPL) is used in the AWS CloudFormation template using Amazon Elastic Compute Cloud (Amazon EC2), Amazon Managed Service for Apache Flink, and Amazon CloudWatch. The KPL is an easy-to-use, highly configurable library that helps you write to Amazon Kinesis Data Streams. It acts as an intermediary between the producer application code and the Kinesis Data Streams API actions. It also integrates seamlessly with the Kinesis Client Library (KCL) and submits Amazon CloudWatch metrics on your behalf to provide visibility into producer performance.

The sample producer application included in this solution uses the following configuration:

Parameter	Value	Description
MaxConnections	1	The maximum number of open connections to the backend. HTTP requests are sent in parallel over multiple connections. The allowed range is from 1 to 256.
RequestTimeout	`60000`	The maximum elapsed time (in milliseconds) from when an HTTP request initiates and when all responses are received. The request times out if the elapsed time is exceeded.
RecordMaxBufferedTime	`2000`	The maximum amount of buffer time (in milliseconds)

Parameter	Value	Description
		that a record is allowed to be buffered before it is sent.
AggregationEnabled	`false`	Choose whether to enable aggregation. With aggregati on, multiple user records are packed into a single **KinesisRecord**. If disabled, each user record is sent in its own **KinesisRecord**.

For more information about configuration parameters, refer to the sample properties file on the Amazon Kinesis Producer GitHub repository.

Uninstall the solution

You can uninstall the Streaming Data Solution for Amazon Kinesis using the AWS Management Console or the AWS Command Line Interface (AWS CLI). However, the Amazon Simple Storage Service (Amazon S3) bucket and Amazon CloudWatch Logs created by this solution must be manually deleted.

Using the AWS Management Console

1. Sign in to the AWS CloudFormation console.
2. On the **Stacks** page, select the solution stack.
3. Choose **Delete**.

Using AWS Command Line Interface

Determine whether AWS Command Line Interface (AWS CLI) is available in your environment. For installation instructions, see What Is the AWS Command Line Interface? in the *AWS CLI User Guide*. After confirming the AWS CLI is available, run the following command.

```
$ aws cloudformation delete-stack --stack-name <cloudformation-stack-name>
```

Replace *<cloudformation-stack-name>* with the name of your CloudFormation stack.

Deleting the Amazon S3 buckets

This solution is configured to retain the Amazon S3 buckets if you choose to delete the AWS CloudFormation stack to prevent against accidental data loss. After uninstalling the solution, you can manually delete the S3 buckets if you do not need to retain the data. Use the following procedure to delete the Amazon S3 buckets.

1. Sign in to the Amazon S3 console.
2. Choose **Buckets** from the left navigation pane.
3. Locate the *<stack-name>* S3 buckets.
4. Select one of the S3 buckets and choose **Delete**.

Repeat the steps until you have deleted all the `<stack-name>` S3 buckets.

Alternatively, you can configure the AWS CloudFormation template to delete the Amazon S3 buckets automatically. Before deleting the stack, change the deletion behavior in the AWS CloudFormation DeletionPolicy attribute.

Deleting the CloudWatch Logs

This solution retains the CloudWatch Logs if you decide to delete the AWS CloudFormation stack to prevent against accidental data loss. After uninstalling the solution, you can manually delete the logs if you do not need to retain the data. Use the following procedure to delete the CloudWatch Logs.

1. Sign in to the Amazon CloudWatch console.
2. Choose **Log Groups** from the left navigation pane.
3. Locate the log groups created by the solution.
4. Select one of the log groups.
5. Choose **Actions** and then choose **Delete**.

Repeat the steps until you have deleted all the solution log groups.

Alternatively, you can configure the AWS CloudFormation template to delete the CloudWatch Logs automatically. Before deleting the stack, change the deletion behavior in the AWS CloudFormation DeletionPolicy attribute.

Deleting the Amazon Cognito User Pool

This solution retains the Cognito user pool if you decide to delete the AWS CloudFormation stack to prevent against accidental data loss. After uninstalling the solution, you can manually delete the user pool if you do not need to retain the data by using the following procedure:

1. Sign in to the Amazon Cognito console.
2. Choose **Manage User Pools**.
3. Locate the user pool created by the solution.
4. Choose **Delete pool**.

Alternatively, you can configure the AWS CloudFormation template to delete the CloudWatch Logs automatically. Before deleting the stack, change the deletion behavior in the AWS CloudFormation DeletionPolicy attribute.

Developer guide

This section provides the source code for the solution.

Source code

Visit our GitHub repository to download the source files for this solution and to share your customizations with others.

The AWS Cloud Development Kit (AWS CDK) generates the Streaming Data Solution for Amazon Kinesis templates. See the README.md file for additional information.

Reference

This section includes information about an optional feature for collecting unique metrics for this solution and a list of builders who contributed to this solution.

Anonymized data collection

This solution includes an option to send anonymized operational metrics to AWS. We use this data to better understand how customers use this solution and related services and products. When enabled, the following information is collected and sent to AWS:

- **Solution ID** - The AWS solution identifier
- **Unique ID (UUID)** - Randomly generated, unique identifier for each solution deployment
- **Timestamp** - The UTC formatted timestamp of when the event occurred
- **Data** - The Region where the stack launched, request type (whether the stack was created, updated, or deleted), and details about the option chosen (for example, shard count, whether enhanced monitoring was enabled, buffering size, etc.). For example:

```
{'Pattern': 'KdsKdfS3', 'RetentionHours': '24',
'CompressionFormat': 'GZIP', 'BufferingInterval': '300',
'ShardCount': '2', 'EnhancedMonitoring': 'false', 'Version': 'v1.2.0',
'BufferingSize': '5', 'Region': 'us-east-1', 'RequestType': 'Create'}
```

Note that AWS owns the data gathered through this survey. Data collection is subject to the AWS Privacy Policy. To opt out of this feature, modify the AWS CloudFormation template mapping section:

1. Download the AWS CloudFormation template to your local hard drive.
2. Open the AWS CloudFormation template with a text editor.
3. Modify the AWS CloudFormation template mapping section from:

```
"Send" : {
"AnonymousUsage" : { "Data" : "Yes" }
},
```

to:

```
"Send" : {
"AnonymousUsage" : { "Data" : "No" }
},
```

4. Sign in to the AWS CloudFormation console.
5. Select **Create stack**.
6. On the **Create stack** page, **Specify template** section, select **Upload a template file**.
7. Under **Upload a template file**, choose **Choose file** and select the edited template from your local drive.
8. Choose **Next** and follow the steps in Launch the stack in the Automated deployment section of this guide.

Contributors

- Mukit Bin Momin
- Tarek Abdunabi
- Daniel Pinheiro
- Morris Estepa
- Abhay Joshi

Document revisions

Date	Change
August 2020	Initial release
September 2020	Release v1.1.0: Refactored the AWS CloudFormation template to use an AWS Solutions Construct; updated command to start the demo producer application in Step 2 of Automated Deployment; for more information about changes for v1.1.0, refer to the CHANGELOG.md file in the Github repository.
October 2020	Release v1.2.0: Added additional template options. For more information about changes for v1.2.0, refer to the CHANGELOG.md file in the Github repository.
January 2021	Release v1.3.0: Added support for Apache Flink 1.11.1. For more information about changes for v1.2.0, refer to the CHANGELOG.md file in the Github repository.
April 2021	Release v1.4.0: Updated demo applications to use the Node.js 14x runtime. For more information, refer to the CHANGELOG .md file file in the GitHub repository.
May 2021	Release v1.4.1: Added Support for Apache Kafka versions 2.8.0 and 2.6.2. For more information, refer to the CHANGELOG.md file in the GitHub repository.
July 2021	Release v1.5.0: Updated authentication for Option 1 to use Amazon Cognito user pools. For more information, refer to the CHANGELOG.md file in the GitHub repository.
November 2021	Release v1.6.0: Added support for Apache Flink 1.13.2; Added support for Amazon Kinesis Data Firehose dynamic partition ing; Updated option 2 to use Amazon Kinesis Data Analytics Studio. For more information, refer to the CHANGELOG.md file in the GitHub repository.

Date	Change
July 2022	Release v1.6.1: Security updates for the Gson package and the minimist and vm2 npm packages. For more information, refer to the CHANGELOG.md file in the GitHub repository.
September 2022	Release v1.6.2: Security patch for vm2 npm package. For more information, refer to the CHANGELOG.md file in the GitHub repository.
November 2022	Release v1.7.0. This release includes integration of this solution with AWS Systems Manager Application Manager and library updates to the AWS CDK and AWS SDK. For more information, refer to the CHANGELOG.md file in the GitHub repository.
December 2022	Release v1.7.1: Removed "AWS" prefix from Service Catalog AppRegistry and Attribute Group Name to correct a common error . For more information, refer to the CHANGELOG.md file in the GitHub repository.
January 2023	Release v1.7.2: Security patch for npm packages. For more information, refer to the CHANGELOG.md file in the GitHub repository.
April 2023	Release v1.7.3: Security patch for npm packages. For more information, refer to the CHANGELOG.md file in the GitHub repository.
April 2023	Release v1.7.4: Mitigated impact caused by new default settings for S3 Object Ownership (ACLs disabled) for all new S3 buckets. For more information, refer to the CHANGELOG.md file in the GitHub repository.
April 2023	Release v1.7.5: Security patch for npm packages. For more information, refer to the CHANGELOG.md file in the GitHub repository.

Date	Change
May 2023	Release v1.7.6: Minor updates and bug fixes. For more information, refer to the CHANGELOG.md file in the GitHub repository.
June 2023	Release v1.7.7: Security patching for npm packages. Update logical ID of AppRegistry resources to prevent CloudForm ation stack update failures. For more information, refer to the CHANGELOG.md file in the GitHub repository.
July 2023	Release v1.7.8: Security patching for Python packages. For more information, refer to the CHANGELOG.md file in the GitHub repository.
September 2023	Release v1.8.0: Migrate to new SDKs. Upgrade Lambda runtimes. Security patches. For more information, refer to the CHANGELOG.md file in the GitHub repository.
October 2023	Release v1.8.1: Security patch. For more information, refer to the CHANGELOG.md file in the GitHub repository.
November 2023	Documentation update: Added Confirm cost tags associate d with the solution to the Monitoring the solution with AWS Service Catalog AppRegistry section.
February 2024	Release v1.9.0: Upgrade Apache Flink. Encrypt data at-rest in Glue Data Catalog. Upgrade Lambda runtimes. Security patches. For more information, refer to the CHANGELOG.md file in the GitHub repository.
June 2024	Release v1.9.1: Onboarded to CloudFormation Guard scanning. Upgraded and patched dependencies. For more information, refer to the CHANGELOG.md file in the GitHub repository.

Notices

AWS Glossary

For the latest AWS terminology, see the AWS glossary in the *AWS Glossary Reference*.