Implementation Guide

Centralized Logging with OpenSearch

Centralized Logging with OpenSearch: Implementation Guide

Table of Contents

Build your own centralized log analytics platform with Amazon OpenSearch Service in 20 minutes

Important

This AWS Solution will retire in December 2026. We encourage customers to explore using Amazon CloudWatch's new unified data management and analytics capabilities. Learn more about AWS CloudWatch unified data and telemetry and give it a try in the AWS CloudWatch console.

The Centralized Logging with OpenSearch solution provides comprehensive log management and analysis functions to help you simplify the build of log analytics pipelines. Built on top of Amazon OpenSearch Service, the solution helps you to streamline log ingestion, log processing, and log visualization. You can use the solution in multiple use cases, such as to abide by security and compliance regulations, achieve refined business operations, and enhance IT troubleshooting and maintenance.

Important

Centralized Logging with OpenSearch supports Amazon OpenSearch Service with OpenSearch 1.3 or later.

Use this navigation table to quickly find responses to these questions:

If you want to ...	Read...
Know the cost for running this solution	Cost
Understand the security considerations for this solution	Security
Know which AWS Regions are supported for this solution	Supported AWS Regions

If you want to ...	Read...
Get started with the solution quickly to import an Amazon OpenSearch Service domain, build a log analytics pipeline, and access the built-in dashboard	Getting started
Learn the operations related to Amazon OpenSearch Service domains	Domain management
Walk through the processes of building log analytics pipelines	AWS Services logs and Application logs
Encountering issues when using the solution	Troubleshooting
Go through a hands-on workshop designed for this solution	Workshop

This implementation guide describes architectural considerations and configuration steps for deploying the Centralized Logging with OpenSearch solution in the AWS Cloud. It includes links to 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, DevOps, and data engineers with practical experience architecting on the AWS Cloud.

Features and benefits

The solution has the following features:

All-in-one log ingestion

Provides a single web console to ingest both application logs and AWS service logs into log analytics engines. For supported AWS service logs, see AWS service logs. For supported application logs, refer to Application logs.

Codeless log processor

Supports log processor plugins developed by AWS. You can enrich the raw log data through a few steps on the web console.

Out-of-the-box dashboard template

Offers a collection of reference designs of visualization templates, for both commonly used software such as NGINX and Apache HTTP Server, and AWS services such as Amazon S3 and AWS CloudTrail.

Use cases

The solution can be applied to the following use cases:

Security and compliance regulations

Comply with regulatory requirements such as GDPR, PCI DSS, MLPS, and HIPAA. Easily store equipment, network, and application logs in a centralized place for log auditing and threat detection.

Business operations and data analysis

Identify trends and patterns in minutes, and build interactive and intuitive visualization. Derive business insights from logs and inform business decisions with data.

Application and infrastructure troubleshooting

Monitor both application and cloud infrastructure logs with ease, understand and resolve the root cause of issues quickly. Improve the observability of your workloads and achieve better business stability.

Concepts

This section describes key concepts and defines terminology specific to this solution:

Log Analytics Engine

A log analytics engine is a sophisticated tool designed to process, analyze, and visualize vast amounts of log data from diverse systems, applications, and devices. Our solution primarily uses the **Amazon OpenSearch Service** as the default log analytics engine, complemented by a **Light Engine** specifically optimized for structured, infrequent logs.

OpenSearch Engine

The OpenSearch Engine in this solution refers to the Amazon OpenSearch Service, a distributed, community-driven, Apache 2.0-licensed, 100% open source search and analytics suite used for a broad set of use cases like real-time application monitoring, log analytics, and website search.

Light Engine

The Light Engine is a serverless log analytics engine that uses AWS services like Athena, Glue, Lambda, and Step Functions. Designed to analyze structured and infrequent logs, it offers up to a 90% cost reduction compared to the OpenSearch Engine.

Log Analytics Pipeline

A Log Analytics Pipeline, or Log Pipeline, represents the entire data flow from the source to the log analytics engines. It typically encompasses the stages of shipping, buffering, processing, filtering, enriching, and storing logs. Centralized Logging with OpenSearch supports two types of Log Analytics Pipelines: the Service Log Pipeline, tailored for ingesting logs generated by AWS Services, and the Application Log Pipeline, designed for ingesting logs from custom applications.

Log Source

A Log Config defines the metadata of your logs, specifying the log type, format, sample logs, filters, and the schema needed to map raw log data into the structured format used by the log analytics engine. A Log Source refers to the location where logs are generated or stored. Centralized Logging with OpenSearch supports ingesting logs from diverse sources, encompassing both application logs and logs from AWS services. For supported AWS service logs, refer to AWS Service Logs. For supported application logs, refer to Application Logs.

Log Agent

A log agent is a program that reads logs from one location and sends them to another location (for example, OpenSearch). Currently, Centralized Logging with OpenSearch only supports the Fluent Bit 1.9 log agent, which is installed automatically. The Fluent Bit agent has a dependency of OpenSSL 1.1. To learn how to install OpenSSL on Linux instances, refer to OpenSSL installation. To find the platforms supported by Fluent Bit, see Supported Platforms.

Log Config

A Log Config defines the metadata of your logs, specifying the log type, format, sample logs, filters, and the schema needed to map raw log data into the structured format used by the log analytics engine.

Log Buffer

Log Buffer is a buffer layer between the Log Agent and OpenSearch clusters. The agent uploads logs into the buffer layer before being processed and delivered into the log analytics engine. A buffer layer is a way to protect the log analytics engine from being overwhelmed. For AWS service logs, a log buffer is automatically configured if needed. For Application logs, this solution provides the following types of buffer layers.

- **Amazon S3**. Use this option if you can bear minutes-level latency for log ingestion. The log agent periodically uploads logs to an Amazon S3 bucket. The frequency of data delivery to Amazon S3 is determined by Buffer size (default value is 50 MiB) and Buffer interval (default value is 60 seconds) values that you configured when creating the application log analytics pipelines. The condition satisfied first starts data delivery to Amazon S3.
- **Amazon Kinesis Data Streams**. Use this option if you need real-time log ingestion. The log agent uploads logs to Amazon Kinesis Data Stream in seconds. The frequency of data delivery to Kinesis Data Streams is determined by Buffer size (10 MiB) and Buffer interval (5 seconds). The condition satisfied first triggers data delivery to Kinesis Data Streams.

Log Buffer is optional when creating an application log analytics pipeline. For all types of application logs, you can use this solution to ingest logs without any buffer layers. However, we only recommend this option when you have small log volume, and you are confident that the logs will not exceed the thresholds at the OpenSearch side.

Instance Group

An Instance Group represents a group of EC2 instances, which enables the solution to associate a Log Config with multiple EC2 instances quickly. Centralized Logging with OpenSearch uses Systems Manager Agent (SSM Agent) to install/configure Fluent Bit agent, and sends log data to Kinesis Data Streams. Instance Group is one of the supported Log Sources in this solution.

Main Account

An AWS account where the Centralized Logging with OpenSearch console is deployed. The Log Analytics Engines must also reside in the same account.

Member Account

Another AWS account from which you want to ingest AWS Service logs or application logs. Logs are sent from Member Accounts to Main Accounts, where they are analyzed using resources in the Main Account.

Access Proxy

An Access Proxy serves as an intermediary for accessing Amazon OpenSearch Service domains from the internet securely. By default, an Amazon OpenSearch Service domain within a VPC is not accessible from the internet. The Centralized Logging with OpenSearch solution implements a Nginx-based proxy stack architecture to enable internet access to OpenSearch Dashboards. This allows users to interact conveniently with the dashboards from anywhere with internet connectivity.

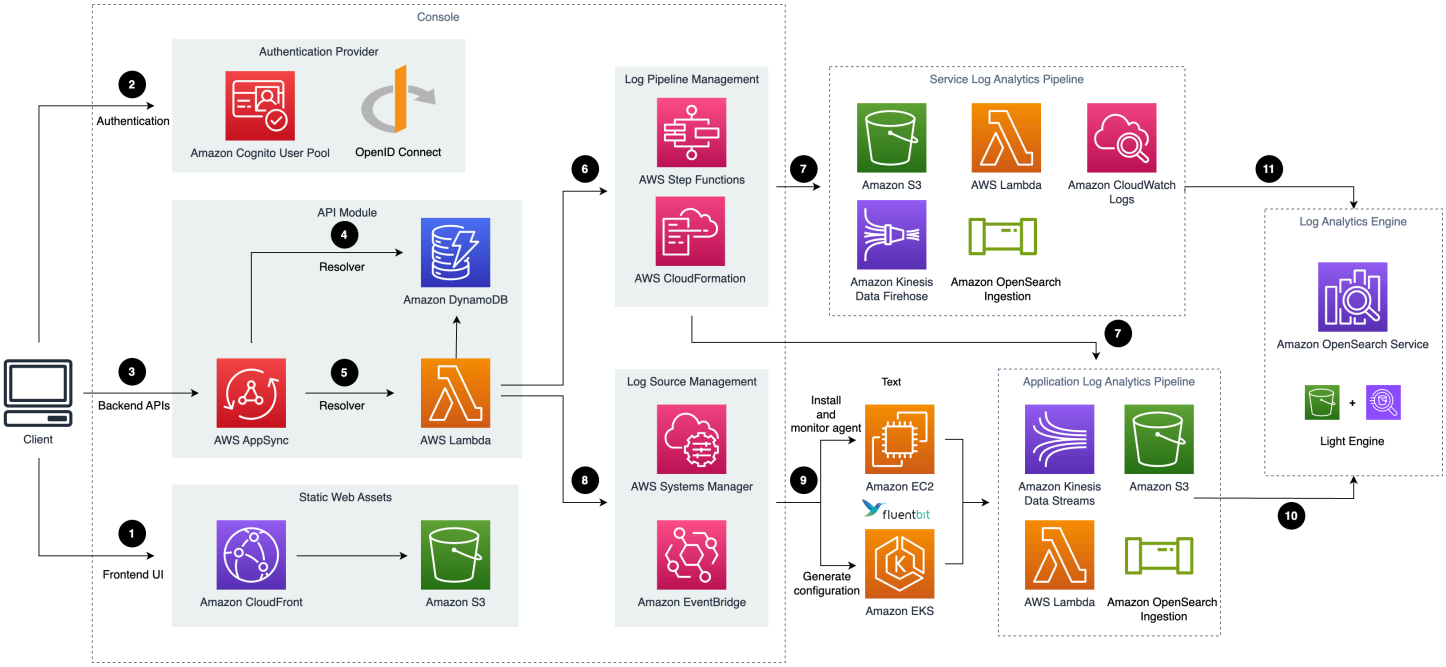
Architecture overview

This section provides a reference implementation architecture diagram with description and AWS Well-Architected design considerations.

Architecture diagram

Deploying this solution with the default parameters builds the following environment in the AWS Cloud.

Architecture diagram, as described in text that follows.



This solution deploys the AWS CloudFormation template in your AWS Cloud account and completes the following settings.

1. Amazon CloudFront distributes the frontend web UI assets hosted in an Amazon S3 bucket.
2. Amazon Cognito user pool or OpenID Connector (OIDC) can be used for authentication.
3. AWS AppSync provides the backend GraphQL APIs.
4. Amazon DynamoDB stores the solution-related information as the backend database.
5. AWS Lambda interacts with other AWS Services to process the core logic of managing log pipeline, log agents, and obtains information updated in DynamoDB tables.

6. AWS Step Functions orchestrates the on-demand AWS CloudFormation deployment of a set of predefined stacks for log pipeline management. The log pipeline stacks deploy separate AWS resources and are used to collect and process logs and ingest them into Amazon OpenSearch Service for further analysis and visualization.
7. Service Log Pipeline or Application Log Pipeline is provisioned on demand via Centralized Logging with the OpenSearch console.
8. AWS Systems Manager and Amazon EventBridge manage log agents for collecting logs from application servers, such as installing log agents (Fluent Bit) for application servers and monitoring the health status of the agents.
9. Amazon EC2 or Amazon EKS installs Fluent Bit agents and uploads log data to the application log pipeline.
10. Application log pipelines read, parse, process application logs, and ingest them into Amazon OpenSearch Service domains or Light Engine.
11. Service log pipelines read, parse, process AWS service logs and ingest them into Amazon OpenSearch Service domains or Light Engine.

> **Note**
>
> After deploying the solution, you can use AWS WAF to protect CloudFront or AWS AppSync. Moreover, you can follow this guide to configure your AWS WAF settings to prevent GraphQL schema introspection.

This solution supports two types of log pipelines: **Service Log Analytics Pipeline** and **Application Log Analytics Pipeline**, and two types of log analytics engines: **OpenSearch Engine** and **Light Engine**. Architecture details for pipelines and Light Engine are described in:

- Service Log Analytics Pipeline
- Application Log Analytics Pipeline
- Light Engine

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 benefit this solution.

Operational excellence

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

- The solution pushes metrics, logs, and traces to Amazon CloudWatch at various stages to provide observability into the infrastructure, Elastic Load Balancing, Amazon ECS cluster, Lambda functions, Step Function workflow, and the rest of the solution components. This solution also creates the CloudWatch dashboards for each pipeline monitoring.

Security

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

- The web console users are authenticated and authorized with Amazon Cognito or OpenID Connect.
- All inter-service communications use AWS IAM roles.
- All roles used by the solution follow least privilege access. That is, it only contains the 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.

- Using AWS serverless services wherever possible (for example, AWS AppSync, Amazon DynamoDB, AWS Lambda, AWS Step Functions, Amazon S3, and Amazon SQS) for high availability and recovery from service failure.

- Configuration management content of the solution is stored in Amazon DynamoDB, all of your data is stored on solid-state disks (SSDs) and is automatically replicated across multiple Availability Zones in an AWS Region, providing built-in high availability and data durability.

Performance efficiency

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

- The ability to launch this solution in any Region that supports AWS services in this solution such as: Amazon S3, Amazon ECS, and Elastic Load Balancing.
- Using serverless architecture removes the need for you to run and maintain physical servers for traditional compute activities.
- Automatically testing and deploying this solution daily. Reviewing this solution by solutions architects and subject matter experts for areas to experiment and improve.

Cost optimization

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

- Using Auto Scaling groups so that the compute costs are only related to how much data is ingested and processed.
- Using serverless services such as Amazon S3, Amazon DynamoDB, and AWS Lambda so that customers only get charged for what they use.

Sustainability

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

- The solution's serverless design (using Amazon Kinesis Data Streams, Amazon S3, AWS Lambda) and the use of managed services (such as Amazon ECS) are aimed at reducing carbon footprint compared to the footprint of continually operating on-premises servers.

Architecture details

This section describes the components and AWS services that make up this solution and the architecture details on how these components work together.

AWS services in this solution

The following AWS services are included in this solution:

AWS service	Description
Amazon CloudFront	To distribute the frontend web UI assets.
Amazon S3	To store the static web assets (frontend user interface), and also uses it as a data buffer for log shipping.
Amazon Cognito	To authenticate users (in AWS Regions).
AWS AppSync	To provide the backend GraphQL APIs.
Amazon DynamoDB	To store the solution related information as a backend database.
AWS Lambda	To interact with other AWS services to process the core logic of managing log pipelines or log agents, and obtain information updated in DynamoDB tables.
AWS Step Functions	To orchestrate on-demand AWS CloudForm ation deployment of a set of predefined stacks for log pipeline management.
AWS CloudFormation	To provision the AWS resources for the modules of pipelines and the solution web console.

AWS service	Description
AWS Systems Manager	To manage log agents for collecting logs from application servers, such as installing log agents (Fluent Bit) for application servers.
Amazon Kinesis Data Streams	To subscribe to logs from a CloudWatch Log Group or as a data buffer for log shipping, and then initiate the Log Processor Lambda to run.
Amazon Data Firehose	To subscribe the logs from CloudWatch Log Group and then put the logs into Amazon S3.
Amazon SQS	To receive Amazon S3 Event Notifications and then initiate the Log Processor Lambda to run.
Amazon Athena	To build the Light Engine.
AWS Glue	To build the Light Engine.

Service log analytics pipeline

Centralized Logging with OpenSearch supports log analysis for AWS services, such as Amazon S3 Access Logs, and Application Load Balancer access logs. For a complete list of supported AWS services, refer to Supported AWS Services.

This solution ingests different AWS service logs using different workflows.

> **Note**
>
> Centralized Logging with OpenSearch supports cross-account log ingestion. If you want to ingest logs from the same account, the resources in the **Sources** group will be in the same account as your Centralized Logging with OpenSearch account. Otherwise, they will be in another AWS account.

Logs through Amazon S3

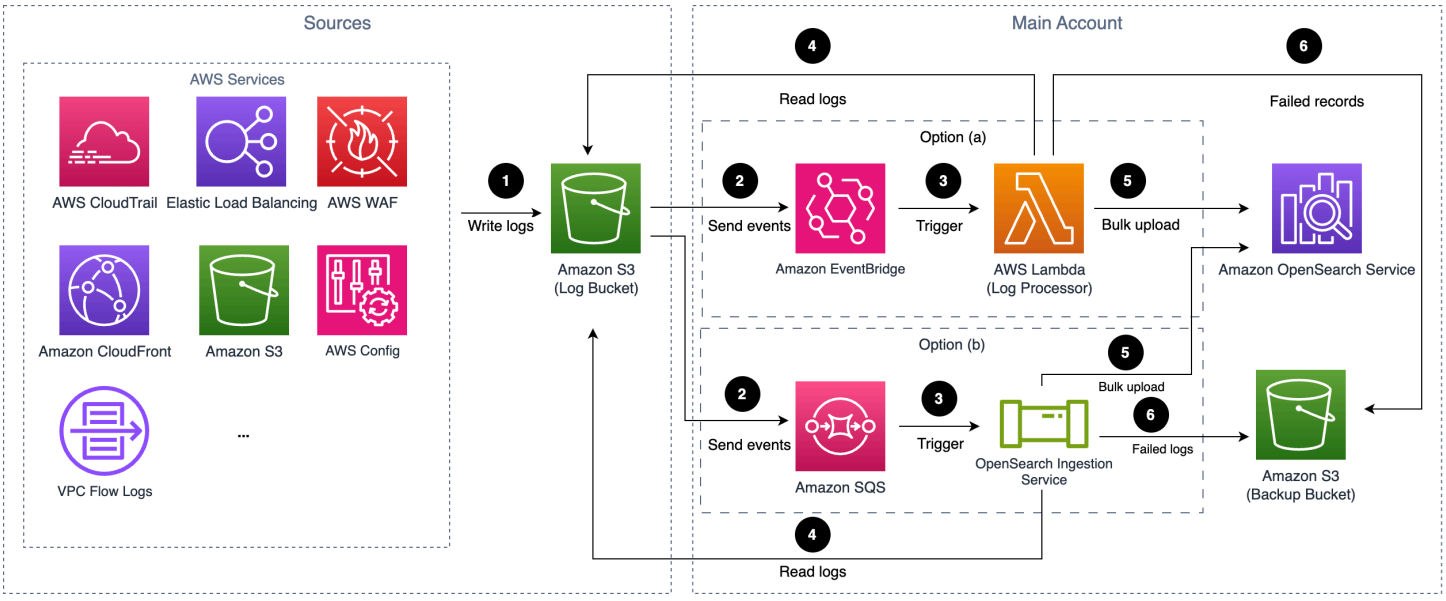
Many AWS services support delivering logs to Amazon S3 directly, or through other services. The workflow supports three scenarios:

Scenario 1: Logs to Amazon S3 directly (OpenSearch Engine)

In this scenario, the service directly delivers logs to Amazon S3. This architecture is applicable to the following log sources:

- AWS CloudTrail logs (delivers to Amazon S3)
- Application Load Balancer access logs
- AWS WAF logs
- Amazon CloudFront standard logs
- Amazon S3 Access Logs
- AWS Config logs
- VPC Flow Logs (delivers to Amazon S3)

Amazon S3 based service log pipeline architecture.



The log pipeline runs the following workflow:

1. AWS services are configured to deliver logs to Amazon S3 bucket (Log Bucket).
2. (Option A) An event notification is sent to Amazon EventBridge when a new log file is created.

(Option B) An event notification is sent to Amazon SQS using S3 Event Notifications when a new log file is created.

3. (Option A) Amazon EventBridge initiates the *Log Processor* Lambda function.

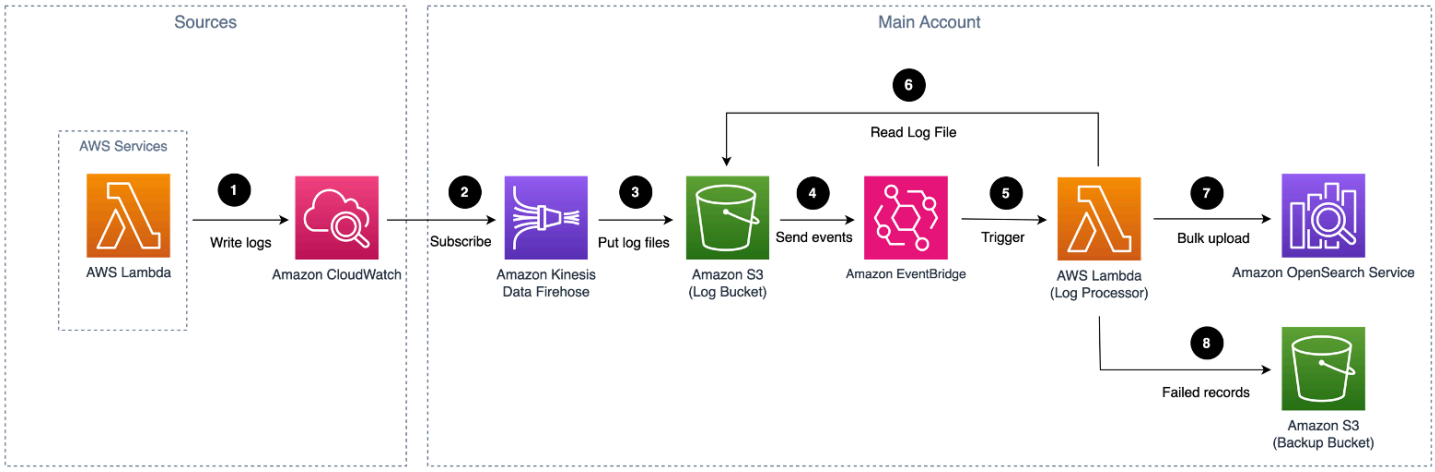
   (Option B) OpenSearch Ingestion Service consumes the SQS messages.
4. The Log Processor Lambda function reads and processes the log files.
5. The Log Processor Lambda function ingests the logs into the Amazon OpenSearch Service.
6. Logs that fail to be processed are exported to Amazon S3 bucket (Backup Bucket).

Scenario 2: Logs to Amazon S3 via Kinesis Data Firehose (OpenSearch Engine)

In this scenario, the service cannot deliver their logs to Amazon S3 directly. The logs are sent to Amazon CloudWatch, and Amazon Data Firehose is used to subscribe the logs from CloudWatch Log Group and then redeliver the logs to Amazon S3. This architecture is applicable to the following log sources:

- AWS Lambda logs

Depicts Amazon S3 (via Firehose) based service log pipeline architecture



The log pipeline runs the following workflow:

1. AWS services logs are configured to deliver logs to Amazon CloudWatch Logs, and Amazon Data Firehose are used to subscribe and store logs in Amazon S3 bucket (Log Bucket).
2. An event notification is sent to Amazon SQS using S3 Event Notifications when a new log file is created.

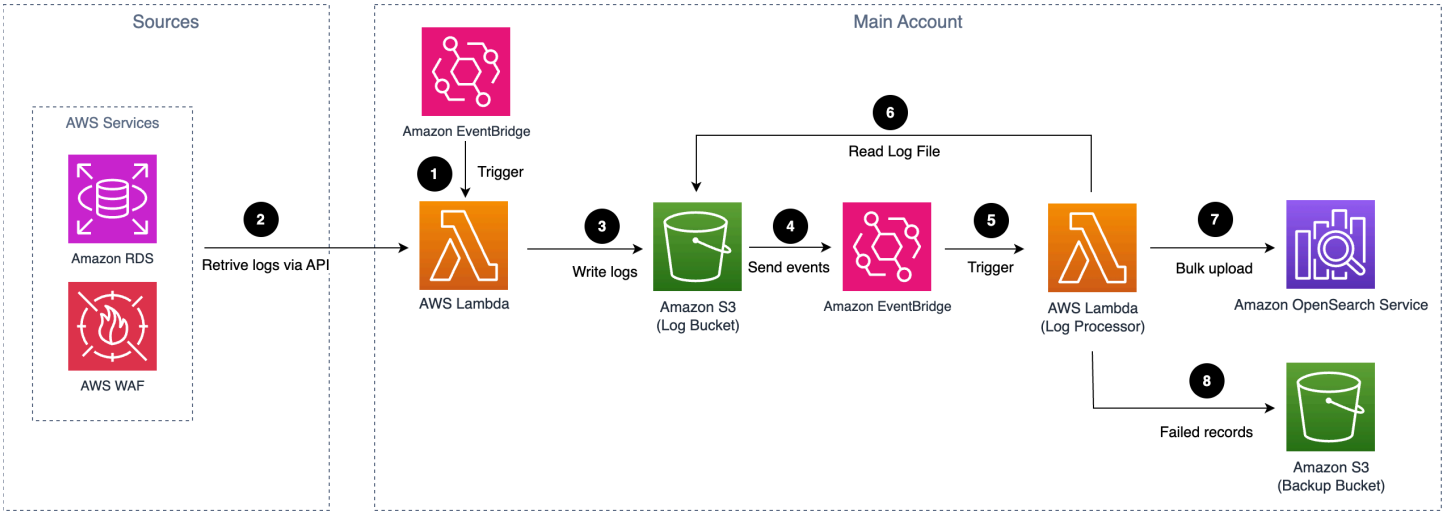
3. Amazon SQS initiates the Log Processor Lambda function to run.
4. The Log Processor Lambda function reads and processes the log files.
5. The Log Processor Lambda function ingests the logs into the Amazon OpenSearch Service.
6. Logs that fail to be processed are exported to Amazon S3 bucket (Backup Bucket).

Scenario 3: Logs to Amazon S3 by calling AWS service API (OpenSearch Engine)

In this scenario, a helper Lambda function periodically invokes AWS service APIs to download log files and save them to an S3 bucket. This architecture is applicable to the following log sources:

- Amazon RDS
- AWS WAF (Sampled logs)

Depicts API call-based service log pipeline architecture



The log pipeline runs the following workflow:

1. Amazon EventBridge periodically triggers the AWS Lambda function. Every 5 mins by default.
2. The AWS Lambda function invokes the appropriate AWS service API to download the log files.
3. The downloaded log files are stored in an Amazon S3 bucket.
4. An event notification is sent to Amazon EventBridge when a new log file is created.
5. Upon receiving the notification, Amazon EventBridge triggers the Log Processor Lambda function.
6. The Log Processor Lambda function reads and processes the log files.

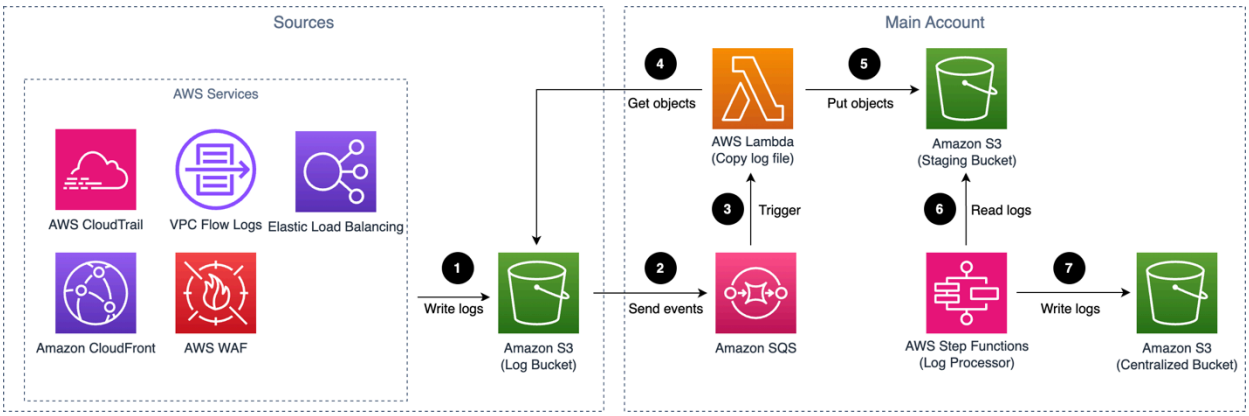
7. The Log Processor Lambda function ingests the processed logs into the Amazon OpenSearch Service for analysis.
8. The Log Processor Lambda function exports any logs that fail to process to a designated Amazon S3 bucket (Backup Bucket) for later review.

Scenario 4: Logs to Amazon S3 directly (Light Engine)

In this scenario, the service directly sends logs to Amazon S3. This architecture is applicable to the following log sources:

- Amazon CloudFront standard logs
- AWS CloudTrail logs (delivers to Amazon S3)
- Application Load Balancer access logs
- AWS WAF logs
- VPC Flow Logs (delivers to Amazon S3)

Amazon S3 based service log pipeline architecture.



The log pipeline runs the following workflow:

1. AWS services are configured to deliver logs to the Amazon S3 bucket (Log Bucket).
2. An event notification is sent to Amazon SQS using S3 Event Notifications when a new log file is created.
3. Amazon SQS initiates the Log Processor Lambda function to run.
4. AWS Lambda retrieves the log file from the Log Bucket.
5. AWS Lambda uploads the log file to the Staging Bucket.

6. The Log Processor Lambda function, AWS Step Functions, processes raw log files stored in the staging bucket in batches.
7. The Log Processor Lambda function converts raw log files to Apache Parquet format, automatically partitions all incoming data based on criteria including time and Region, calculates relevant metrics, and stores the results in the Centralized Bucket.

Logs through Amazon Kinesis Data Streams

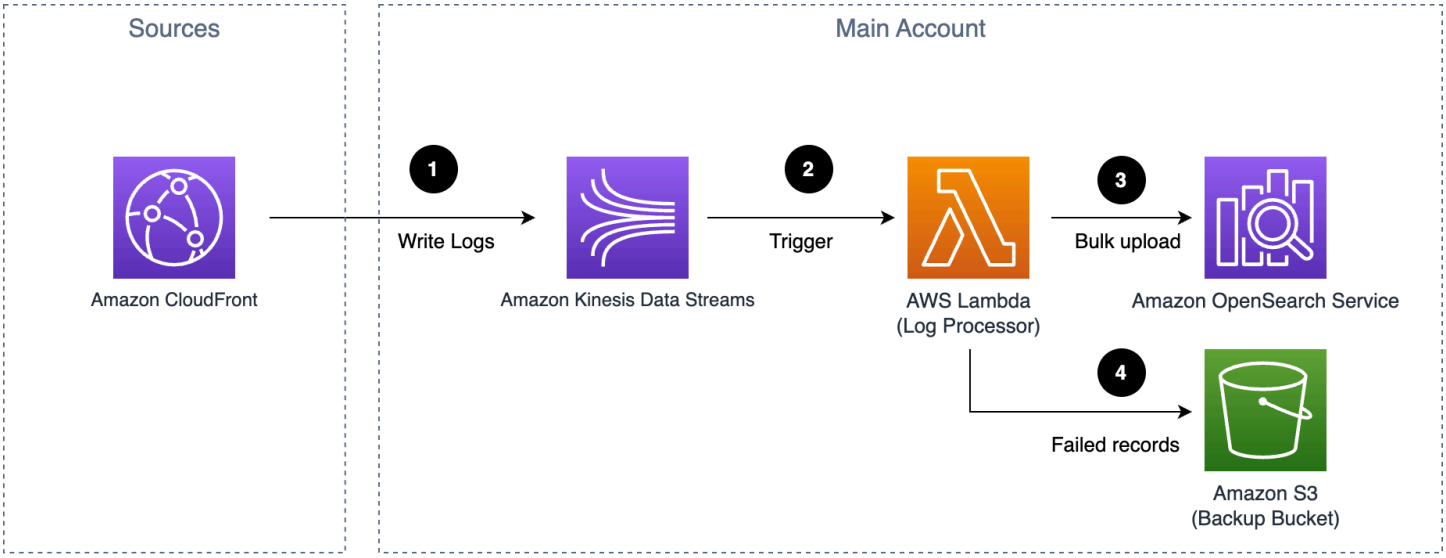
Some AWS services support delivering logs to Amazon Kinesis Data Streams. The workflow supports two scenarios:

Scenario 1: Logs to Kinesis Data Streams directly (OpenSearch Engine)

In this scenario, the service directly delivers logs to Amazon Kinesis Data Streams. This architecture is applicable to the following log sources:

- Amazon CloudFront real-time logs

Amazon Kinesis Data Streams based service log pipeline architecture.



> **Warning**
>
> This solution does not support cross-account ingestion for CloudFront real-time logs.

The log pipeline runs the following workflow:

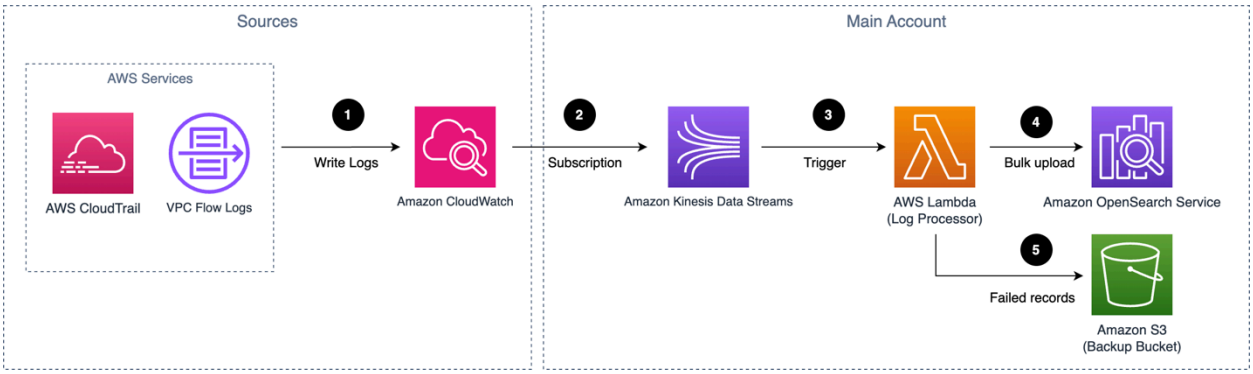
1. AWS services are configured to deliver logs to Amazon Kinesis Data Streams.
2. Amazon Kinesis Data Streams trigger AWS Lambda to execute.
3. AWS Lambda read, parse, and process logs from Kinesis Data Streams, and upload to Amazon OpenSearch Service.
4. Logs that fail to be processed are exported to Amazon S3 bucket (Backup Bucket).

Scenario 2: Logs to Kinesis Data Streams via CloudWatch Logs (OpenSearch Engine)

In this scenario, the service delivers the logs to CloudWatch Logs, and then CloudWatch Logs redeliver the logs in real-time to Kinesis Data Streams using subscription. This architecture is applicable to the following log sources:

- AWS CloudTrail logs (delivers to CloudWatch Logs)
- VPC Flow Logs (delivers to CloudWatch Logs)

Amazon Kinesis Data Streams (via CloudWatch Logs) based service log pipeline architecture.



The log pipeline runs the following workflow:

1. AWS Services logs are configured to write logs Amazon CloudWatch Logs.
2. Logs are redelivered to Kinesis Data Streams via CloudWatch Logs subscription.
3. Kinesis Data Streams initiates the AWS Lambda (Log Processor) to execute.
4. The Log Processor processes and ingests the logs into the Amazon OpenSearch Service.
5. Logs that fail to be processed are exported to Amazon S3 bucket (Backup Bucket).

For cross-account ingestion, the AWS Services store logs on Amazon CloudWatch log group in the member account, and other resources remain in the main account.

Application log analytics pipeline

Centralized Logging with OpenSearch supports log analysis for application logs, such as NGINX/ Apache HTTP Server logs or custom application logs.

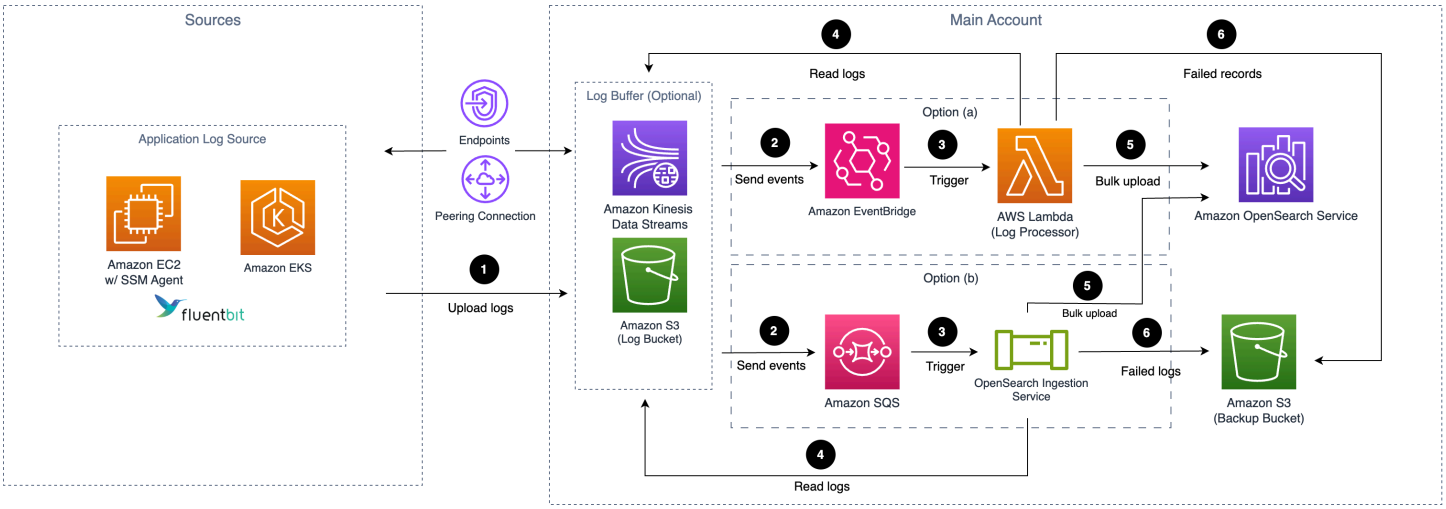
> **Note**
>
> Centralized Logging with OpenSearch supports cross-account log ingestion. If you want to ingest logs from the same account, the resources in the **Sources** group will be in the same account as your Centralized Logging with OpenSearch account. Otherwise, they will be in another AWS account.

Logs from Amazon EC2 / Amazon EKS

Centralized Logging with OpenSearch supports collecting logs from Amazon EC2 instances or Amazon EKS clusters. The workflow supports two scenarios.

Scenario 1: Using OpenSearch Engine

Application log pipeline architecture for EC2/EKS.



The log pipeline runs the following workflow:

1. Fluent Bit works as the underlying log agent to collect logs from application servers and send them to an optional Log Buffer, or ingest into OpenSearch domain directly.
2. (Option A) The Log Buffer sends events to Amazon EventBridge.

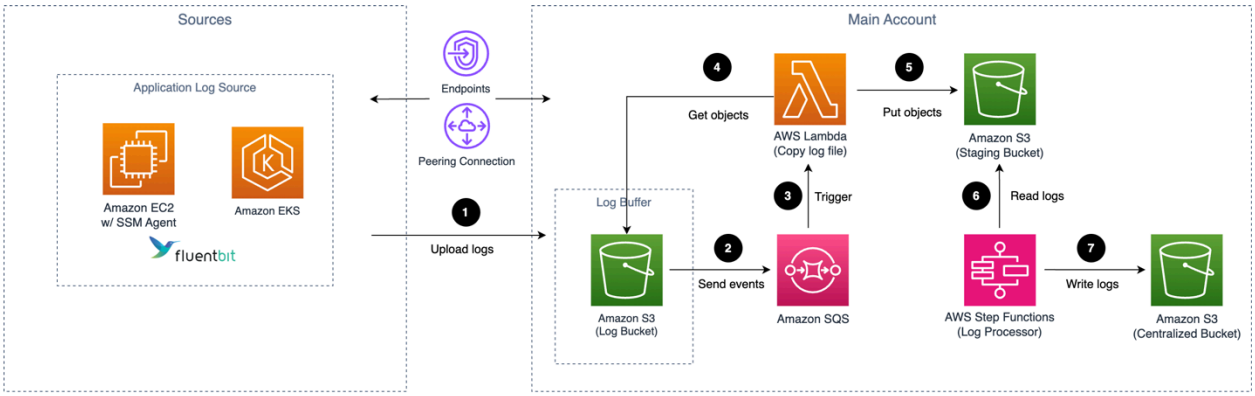
(Option B) The Log Buffer sends events to Amazon SQS.

3. (Option A) Amazon EventBridge triggers the Log Processor Lambda function to execute.

   (Option B) Amazon SQS triggers the OpenSearch Ingestion Service to execute.
4. The AWS Lambda function or OpenSearch Ingestion Service reads and processes the log records.
5. The AWS Lambda function or OpenSearch Ingestion Service ingests the logs into the OpenSearch domain.
6. Logs that fail to be processed are exported to an Amazon S3 bucket (Backup Bucket).

Scenario 2: Using Light Engine

Application log pipeline architecture for EC2/EKS.



The log pipeline runs the following workflow:

1. Fluent Bit works as the underlying log agent to collect logs from application servers and send them to a Log Bucket.
2. An event notification is sent to Amazon SQS using S3 Event Notifications when a new log file is created.
3. Amazon SQS initiates AWS Lambda to execute.
4. AWS Lambda load the log file from the Log Bucket.
5. AWS Lambda put the log file to the Staging Bucket.
6. The Log Processor, AWS Step Functions, processes raw log files stored in the staging bucket in batches.
7. The Log Processor converts raw log files to Apache Parquet format and automatically partitions all incoming data based on criteria including time and Region.

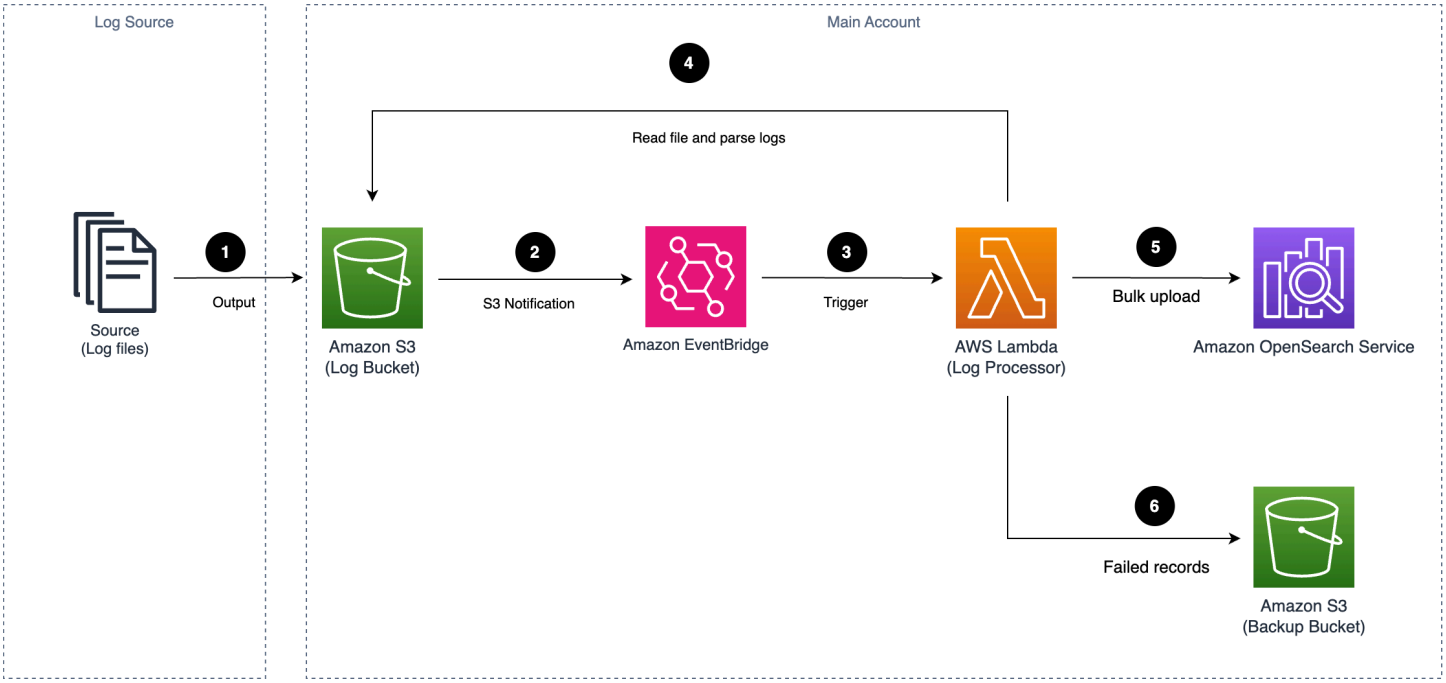
Logs from Amazon S3

Centralized Logging with OpenSearch supports collecting logs from Amazon S3 buckets. The workflow supports three scenarios:

Scenario 1: Using OpenSearch Engine (Ongoing)

In this scenario, the solutions continuously read and parse logs whenever you upload a log file to the specified Amazon S3 location.

Application log pipeline architecture for Amazon S3.



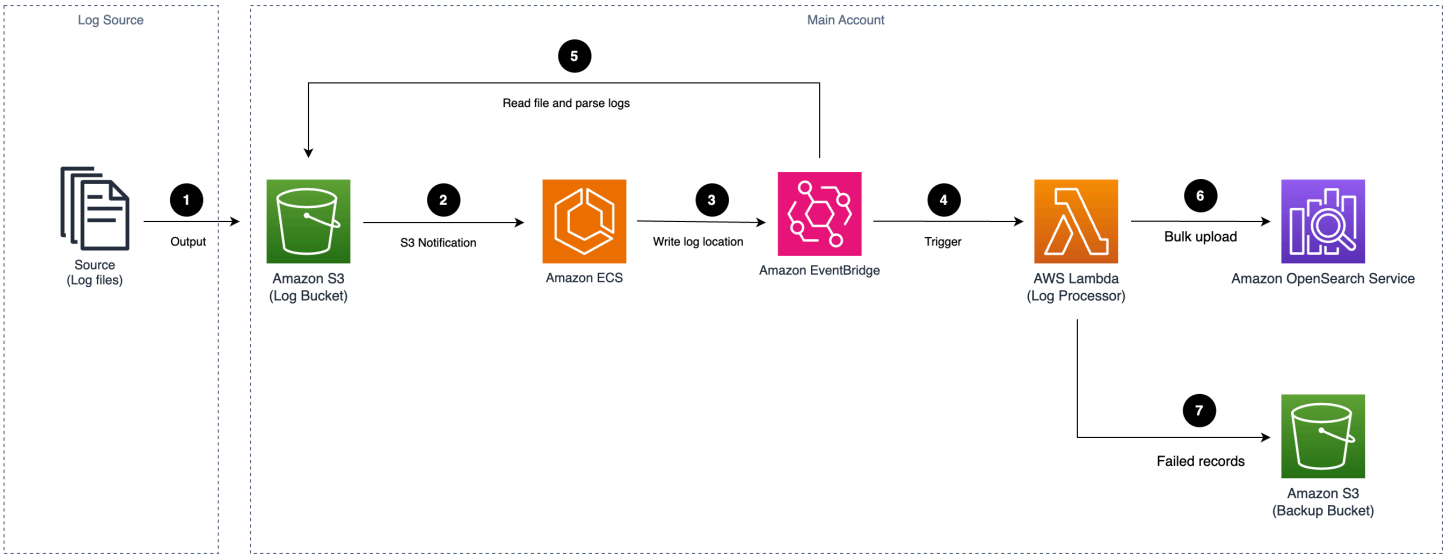
The log pipeline runs the following workflow:

1. User uploads logs to an Amazon S3 bucket (Log Bucket).
2. An event notification is sent to Amazon EventBridge when a new log file is created.
3. Amazon EventBridge initiates AWS Lambda (Log Processor) to execute.
4. The Log Processor reads and processes log files.
5. The Log Processor ingests the processed logs into the OpenSearch domain.
6. Logs that fail to be processed are exported to an Amazon S3 bucket (Backup Bucket).

Scenario 2: Using OpenSearch Engine (One-time)

In this scenario, the solution scans existing log files stored in the specified Amazon S3 location and ingests them into the log analytics engine in a single operation.

Application log pipeline architecture for Amazon S3.

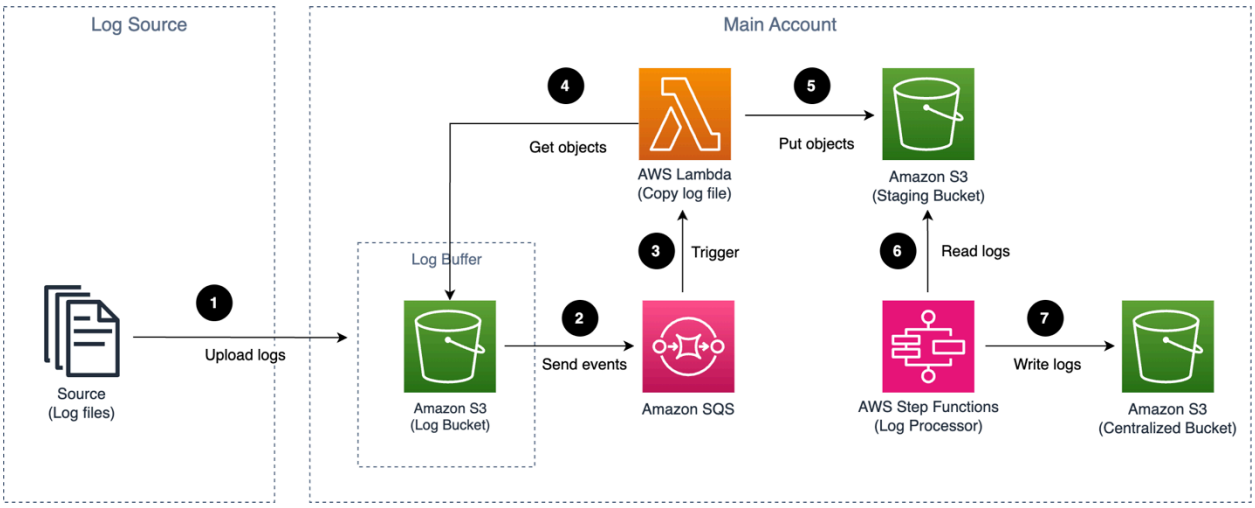


The log pipeline runs the following workflow:

1. User uploads logs to an Amazon S3 bucket (Log Bucket).
2. Amazon ECS Task iterates log files in the Log Bucket.
3. Amazon ECS Task sends the log location to an Amazon EventBridge.
4. Amazon EventBridge initiates AWS Lambda to execute.
5. The Log Processor reads and parses log files.
6. The Log Processor ingests the processed logs into the OpenSearch domain.
7. Logs that fail to be processed are exported to an Amazon S3 bucket (Backup Bucket).

Scenario 3: Using Light Engine (Ongoing)

Application log pipeline architecture for Amazon S3.



The log pipeline runs the following workflow:

1. Logs are uploaded to an Amazon S3 bucket (Log bucket).
2. An event notification is sent to Amazon SQS using S3 Event Notifications when a new log file is created.
3. Amazon SQS initiates AWS Lambda.
4. AWS Lambda copies objects from the Log bucket.
5. AWS Lambda output the copied objects to the Staging bucket.
6. AWS Step Functions periodically trigger Log Processor to process raw log files stored in the staging bucket in batches.
7. The Log Processor converts them into Apache Parquet format and automatically partitions all incoming data based on criteria including time and Region.

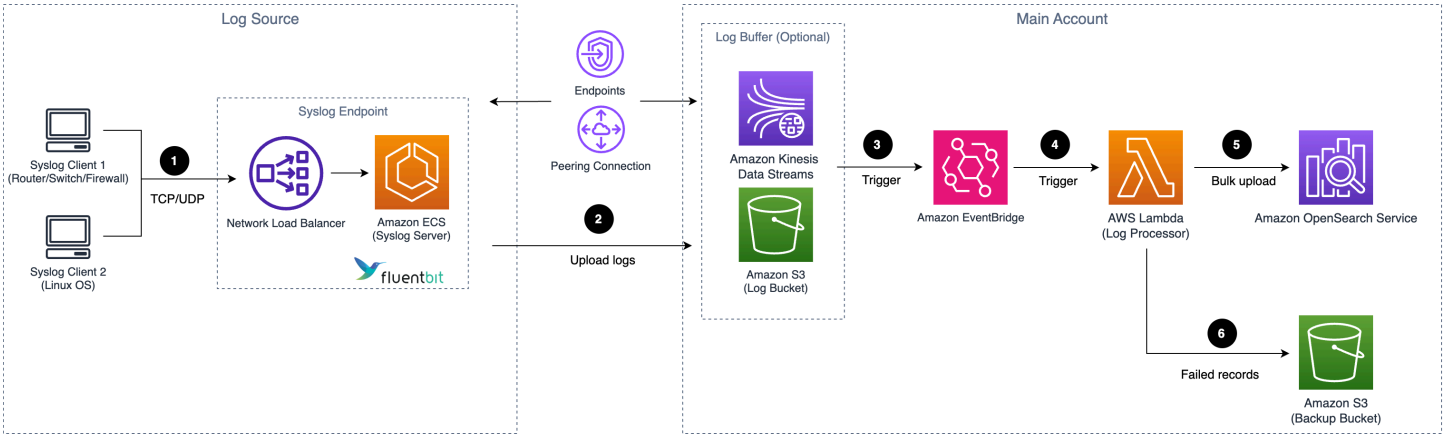
Logs from Syslog Client

Important

1. Make sure your Syslog generator/sender's subnet is connected to Centralized Logging with OpenSearch's **two** private subnets. You may need to use VPC Peering Connection or Transit Gateway to connect these VPCs.
2. The Network Load Balancer together with the Amazon ECS containers in the architecture diagram will be provisioned only when you create a Syslog ingestion and be automated deleted when there is no Syslog ingestion.

Scenario 1: Using OpenSearch Engine

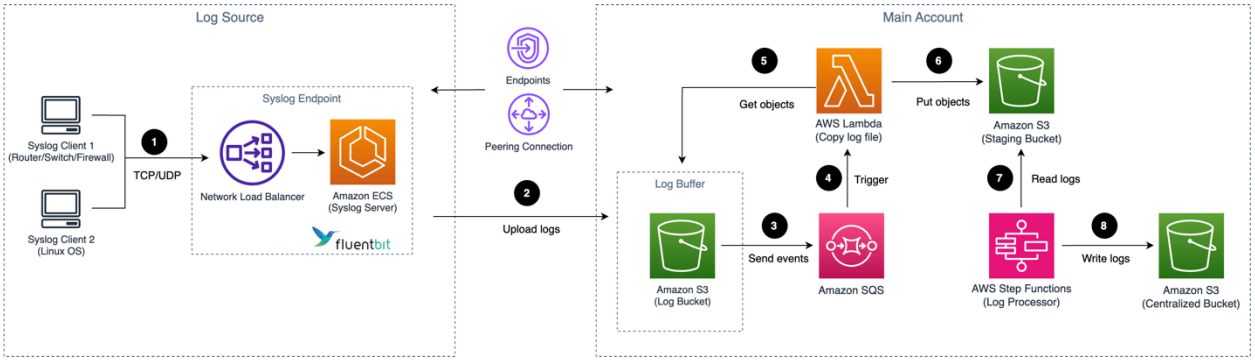
Application log pipeline architecture for Syslog.



1. Syslog client (like Rsyslog) sends logs to a Network Load Balancer in Centralized Logging with OpenSearch's private subnets, and the Network Load Balancer routes to the Amazon ECS containers running Syslog servers.
2. Fluent Bit works as the underlying log agent in the Amazon ECS service to parse logs, and send them to an optional Log Buffer, or ingest into OpenSearch domain directly.
3. The Log Buffer sends messages to Amazon EventBridge.
4. Amazon EventBridge triggers the Log Processor Lambda function to run.
5. The Log Processor Lambda function reads and processes the log records and ingests the logs into the OpenSearch domain.
6. Logs that fail to be processed are exported to an Amazon S3 bucket (Backup Bucket).

Scenario 2: Using Light Engine

Application log pipeline architecture for Syslog.



1. Syslog client (like Rsyslog) send logs to a Network Load Balancer in Centralized Logging with OpenSearch's private subnets, and the Network Load Balancer routes to the Amazon ECS containers running Syslog servers.
2. Fluent Bit works as the underlying log agent in the Amazon ECS Service to parse logs, and send them to an optional Log Buffer, or ingest into OpenSearch domain directly.
3. An event notification is sent to Amazon SQS using S3 Event Notifications when a new log file is created.
4. Amazon SQS initiates AWS Lambda.
5. AWS Lambda copies objects from the Log bucket.
6. AWS Lambda output the copied objects to the Staging bucket
7. AWS Step Functions periodically trigger Log processor to process raw log files stored in the staging bucket in batches.
8. The Log Processor converts them into Apache Parquet format and automatically partitions all incoming data based on criteria including time and Region.

Light Engine

The Light Engine is a complementary log analytics engine to the default OpenSearch Engine. It is designed for analyzing structured and infrequent logs, offering up to a 90% cost reduction. The Light Engine uses several AWS services, including Amazon Athena, AWS Glue, AWS Lambda, and AWS Step Functions, to achieve its scale and cost-effective performance.

OpenSearch Engine vs. Light Engine

We suggest using two engines to tackle real-world challenges. For mission-critical, high-performance, or essential workloads, opt for OpenSearch. Conversely, use Light Engine to save costs when handling secondary, less critical workloads that do not require the same level of performance.

The following table provides guidance for you to choose the appropriate analytics engine. For more assistance, consult an AWS Solutions Architect.

	OpenSearch Engine	Light Engine
Ingest latency	Seconds ~ Minutes (depends on buffer layer type)	5 minutes preceding

	OpenSearch Engine	Light Engine
Query latency	Milliseconds ~ Seconds	Seconds ~ Minutes
Text search	Full-text	Fuzzy (the "Like" syntax of SQL)
Query language	DSL (Domain Specific Language)	SQL
Dashboards	OpenSearch Dashboards	Grafana
Alarm	Provided by OpenSearch Dashboards	Provided by Grafana
Access control	Field level	Table level
Sharing	Export data	Parquet files in Amazon S3
Operational effort	High	Low
Schema	Semistructured	Structured
Cost	High	Low
Pricing model	Fixed; OpenSearch node types and counts, storage	On demand

Key components

Log Bucket

A Log Bucket is an Amazon S3 bucket where you (or the log agent) deliver the raw logs to. The Log Bucket must reside in the same AWS account and Region as the solution you are implementing.

Staging Bucket

The Staging Bucket is an Amazon S3 storage location that holds a small portion of the raw log files, which are copied from the Log Bucket. The solution then processes these raw log files in the Staging Bucket in batches.

Centralized Bucket

The Centralized Bucket is an Amazon S3 storage location where Light Engine stores the partitioned, and compressed (in Apache Parquet format) log files. The schema and format are optimized for query by Light Engine.

Archive Bucket

The Archive Bucket is an Amazon S3 storage location where Light Engine moves the data from the Centralized Bucket, and then use Amazon S3 Lifecycle policy to delete the logs.

Log Processor

The Log Processor is triggered at a regular interval (default is 5 minutes) by Amazon EventBridge. It processes logs to accomplish the following tasks:

- Process raw log files stored on Amazon S3 in batches, and transform them into Apache Parquet format.
- Automatically partition all incoming data by time, Region, and other relevant attributes.
- Calculate metrics of log data and save to the Centralized bucket.
- Trigger Amazon SNS notifications when task execution fails.
- Each Pipeline/Ingestion corresponds to an Amazon EventBridge rule that periodically triggers the log processor, for instance, every 5 minutes.

Log Merger

The Log Merger periodically (default is once every day at 1:00AM UTC) merges small parquet format files into large parquet format files, and further improve the query performance. It accomplishes the following tasks:

- Merge small files into files of a specified size, reduce the number of files, and reduce data storage.
- Optimize the partition granularity and update the AWS AWS Glue Data Catalog to reduce the number of partitions.
- Trigger Amazon SNS notifications when task execution fails.
- Each pipeline corresponds to an Amazon EventBridge rule to periodically trigger log merger.

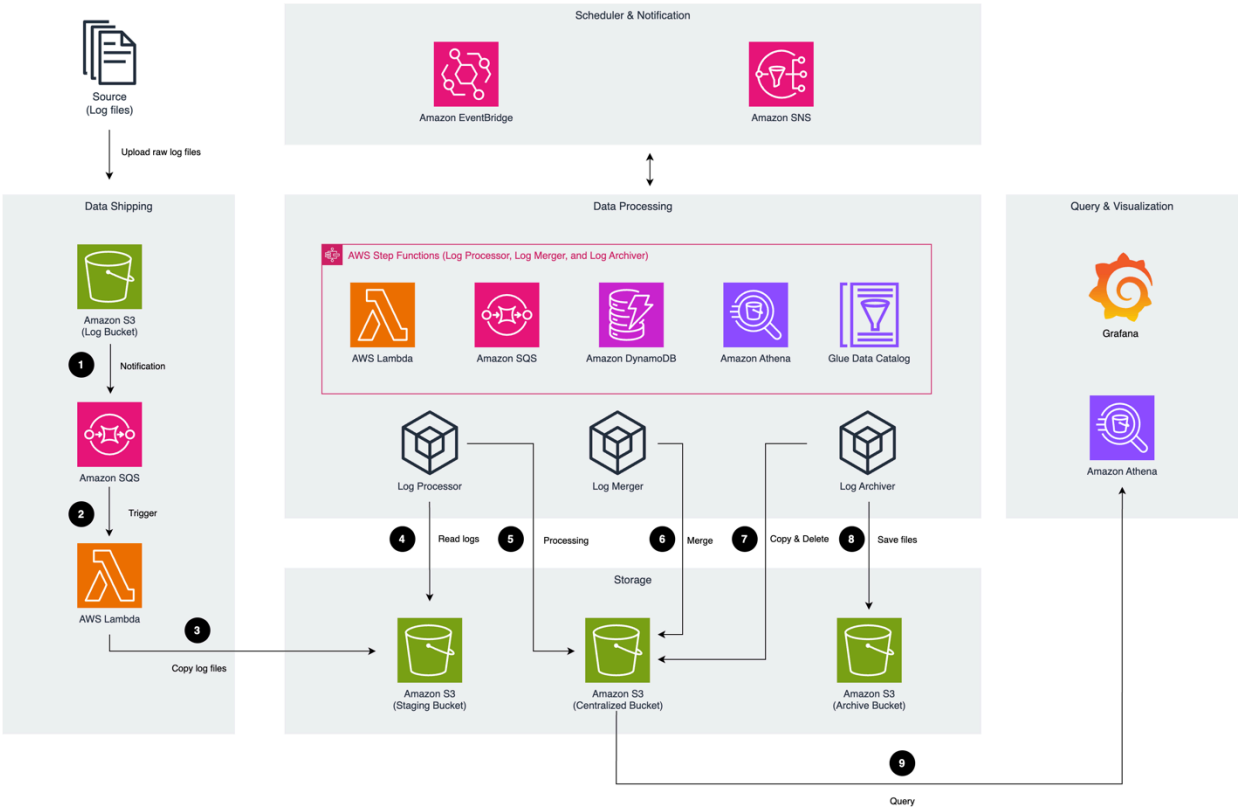
Log Archiver

The Log Archiver manages the lifecycle of data stored in Amazon S3, and cleanup table partitions.

- Move the expired data in Centralized to archived until the lifecycle rule deletes the files.
- Update AWS AWS Glue Data Catalog and delete expired table partitions
- Trigger Amazon SNS notifications when task execution fails.
- Each pipeline corresponds to an Amazon EventBridge rule to periodically trigger log archive, for instance, every day at 1:00AM.

Architecture Diagram

Light Engine architecture.



The Light Engine runs the following workflow:

1. Logs are uploaded to an Amazon S3 bucket (Log Bucket). An event notification is sent to Amazon SQS using S3 Event Notifications when a new log file is created.
2. Amazon SQS triggers AWS Lambda to execute.
3. AWS Lambda copy log files from Log Bucket to Staging Bucket.
4. The Log Processor, scheduled by Amazon EventBridge, read logs from the Staging Bucket.

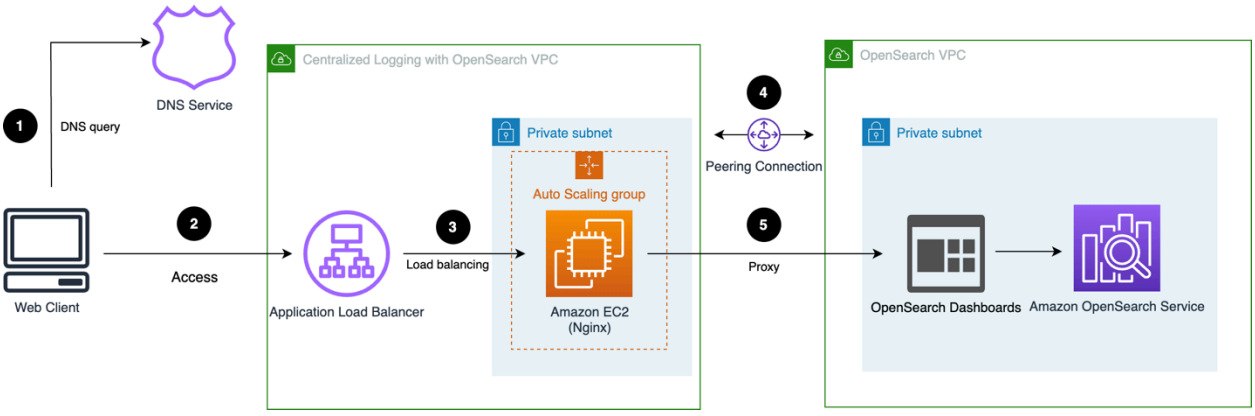
5. The Log Processor parses, calculates metrics (for AWS Service logs), compresses using zstd method, partitions, and saves the processed data in Parquet format into the Centralized Bucket.
6. Log Merger, scheduled by Amazon EventBridge, reads logs from the Centralized Bucket. It then merges small log files into larger ones, and saves them back to the Centralized Bucket. This process helps to optimize the storage, and improve the performance when querying.
7. Log Archiver, scheduled by Amazon EventBridge, periodically copies the log files to the Archive Bucket, and delete log files.
8. Log Archiver saves the copied log files to the Archived Bucket, and then uses the S3 Lifecycle to delete the logs according to the configurations.
9. Users query and visualize logs in Grafana, and Grafana uses Athena to query processed logs in Centralized Bucket.

   When any errors occur during data processing, notifications will be sent to the Simple Notification Service (Amazon SNS). You can configure Amazon SNS to deliver these notifications via SMS, email, or instant messaging, to be promptly informed of any issues that arise.

Access proxy

Centralized Logging with OpenSearch creates an Auto Scaling group together with an Application Load Balancer.

Centralized Logging with OpenSearch creates an Auto Scaling group together with an Application Load Balancer.



The workflow is as follows:

1. Users access the custom domain for the proxy, and the domain must be resolved via DNS service (for example, using Route 53 on AWS).

2. The DNS service routes the traffic to internet-facing Application Load Balancer.
3. The Application Load Balancer distributes traffic to a backend NGINX server running on Amazon EC2 within an Auto Scaling group.
4. (optional) VPC peering is required if the VPC for the proxy is not the same as the OpenSearch Service.
5. The NGINX server redirects the requests to OpenSearch Dashboards.

Plan your deployment

This section describes the cost, security, and other considerations before deploying the solution.

Supported AWS Regions

This solution uses services that may not be currently available in all AWS Regions. Launch this solution in an AWS Region where required services are available. For the most current availability by Region, refer to the AWS Regional Services List.

Centralized Logging with OpenSearch provides two types of authentications, Amazon Cognito User Pool and OpenID Connect (OIDC) Provider. Choose to launch the solution with OpenID Connect if one of the following cases occurs:

- Amazon Cognito User Pool is not available in your AWS Region.
- You already have an OpenID Connect Provider and want to authenticate against it.

Supported Regions for deployment

Region Name	Launch with Amazon Cognito User Pool	Launch with OpenID Connect
US East (N. Virginia)	✓	✓
US East (Ohio)	✓	✓
US West (N. California)	✓	✓
US West (Oregon)	✓	✓
Canada (Central)	✓	✓
Canada (Calgary)	X	✓
Africa (Cape Town)	X	✓
Asia Pacific (Hong Kong)	X	✓
Asia Pacific (Mumbai)	✓	✓

Region Name	Launch with Amazon Cognito User Pool	Launch with OpenID Connect
Asia Pacific (Osaka)	X	✓
Asia Pacific (Seoul)	✓	✓
Asia Pacific (Singapore)	✓	✓
Asia Pacific (Sydney)	✓	✓
Asia Pacific (Tokyo)	✓	✓
Asia Pacific (Hyderabad)	X	✓
Asia Pacific (Jakarta)	✓	✓
Asia Pacific (Melbourne)	X	✓
Israel (Tel Aviv)	X	✓
Middle East (Bahrain)	✓	✓
Middle East (UAE)	X	✓
Europe (Frankfurt)	✓	✓
Europe (Ireland)	✓	✓
Europe (London)	✓	✓
Europe (Milan)	X	✓
Europe (Paris)	✓	✓
Europe (Stockholm)	✓	✓
Europe (Spain)	X	✓
Europe (Zurich)	X	✓
South America (Sao Paulo)	✓	✓

Region Name	Launch with Amazon Cognito User Pool	Launch with OpenID Connect
China (Beijing) Region Operated by Sinnet	X	✓
China (Ningxia) Regions operated by NWCD	X	✓

Important

You can have only one active Centralized Logging with OpenSearch solution stack in one Region. If your deployment failed, make sure you have deleted the failed stack before retrying the deployment.

Cost

Important

The following cost estimations are examples and may vary depending on your environment.

You will be responsible for the cost of the AWS services used when running the solution. The main factors affecting the solution cost include:

- Log analytics engine selection
- Type of logs to be ingested
- Volume of logs to be ingested/processed
- Size of the log message
- Location of logs
- Additional features

As of this revision, the following examples demonstrate the cost estimation of 10/100/1000 GB daily log ingestion for running this solution with default settings in the US East (N. Virginia) Region. The cost formulation depends on the type of log analytics engine. The solution console can manage both OpenSearch Engine and Light Engine, and the solution console cost remains the same.

Using OpenSearch Engine

This section describes the cost when choosing OpenSearch Engine. It is composed of Amazon OpenSearch Service Cost, Processing Cost, and Additional Features Cost.

Amazon OpenSearch Service Cost

- **OD**: On Demand
- **AURI_1**: All Upfront Reserved Instance 1 Year
- **Tiering**: The days stored in each tier. For example, 7H + 23W + 60C indicates that the log is stored in hot tier for 7 days, warm tier for 23 days, and cold tier for 60 days.
- **Replica**: The number of shard replicas.

Daily log Volum (GB)	Reten (days)	Tierin	Replic	OD Month (USD)	AURI_ Month (USD)	Dedic Maste	Data Node	Amaz EBS (GB)	UltraV Nodes	UltraV / Cold Amaz S3 Storag (GB)	OD cost per GB (USD)	AURI_1 cost per GB ($)
10	30	30H	0	216.28	158.54	N/A	c6g.lar [2]	380	N/A	0	0.720	0.52847
10	30	30H	1	289.35	223.94	N/A	m6g.la [2]	760	N/A	0	0.964	0.74647
100	30	7H + 23W	0	989.49	825.97	m6g.la [3]	m6g.la [2]	886	mediu	0	0.329	0.27532

Daily log Volum (GB)	Reten (days)	Tierin	Replic	OD Month (USD)	AURI_ Month (USD)	Dedic Maste	Data Node	Amaz EBS (GB)	UltraV Nodes	UltraV / Cold Amaz S3 Storag (GB)	OD cost per GB (USD)	AURI_1 cost per GB ($)
100	30	7H + 23W	1	1295.8	1066.9	m6g.la [3]	m6g.la [4]	1772	mediu	0	0.431	0.35564
100	90	7H + 23W + 60C	0	1133.4	969.97	m6g.la [3]	m6g.la [2]	886	mediu	8300	0.125	0.10777
100	90	7H + 23W + 60C	1	1439.8	1210.9	m6g.la [3]	m6g.la [4]	1772	mediu	8300	0.159	0.13455
100	180	7H + 23W + 150C	0	1349.4	1185.9	m6g.la [3]	m6g.la [2]	886	mediu	17300	0.074	0.06589
100	180	7H + 23W + 150C	1	1655.8	1426.9	m6g.la [3]	m6g.la [4]	1772	mediu	17300	0.091	0.07927
1000	30	7H + 23W	0	6101.1	5489.4	m6g.la [3]	r6g.xla e[6]	8856	mediu]	23000	0.203	0.18298
1000	30	7H + 23W	1	8759.4	7635.8	m6g.la [3]	r6g.2x ge[6]	17712	mediu]	23000	0.291	0.25453

Daily log Volum (GB)	Reten (days)	Tierin	Replic	OD Month (USD)	AURI_ Month (USD)	Dedic Maste	Data Node	Amaz EBS (GB)	UltraV Nodes	UltraV / Cold Amaz S3 Storag (GB)	OD cost per GB (USD)	AURI_1 cost per GB ($)
1000	90	7H + 23W + 60C	0	8027.3	7245.4	m6g.la [3]	r6g.xla e[6]	8856	mediu]	83000	0.089	0.0805
1000	90	7H + 23W + 60C	1	10199	9075.8	m6g.la [3]	r6g.2x ge[6]	17712	mediu]	83000	0.113	0.10084
1000	180	7H + 23W + 150C	0	9701.1	9089.4	m6g.la [3]	r6g.xla e[6]	8856	mediu]	17300	0.053	0.0505
1000	180	7H + 23W + 150C	1	12644	11420	m6g.la [3]	r6g.2x ge[6]	17712	mediu]	17300	0.070	0.06345

Processing Cost

Log ingestion through Amazon S3

This section is applicable to:

- AWS service logs, including Amazon S3 Access Logs, CloudFront standard logs, CloudTrail logs (Amazon S3), Application Load Balancer access logs, AWS WAF logs, VPC Flow Logs (Amazon S3), AWS Config logs, Amazon RDS/Aurora logs, and AWS Lambda Logs.
- Application Logs that use Amazon S3 as a data buffer.

Assumptions:

- The logs stored in Amazon S3 are in gzip format.
- A 4MB compressed log file in Amazon S3 is roughly 100 MB in raw log size.
- A Lambda with 1 GB memory takes about 26 seconds to process a 4 MB compressed log file, namely 260 milliseconds (ms) per MB raw logs.
- The maximum compressed log file size is 5 MB.
- Ingesting logs from Amazon S3 will incur Amazon SQS and Amazon S3 request fees, which are very low, or usually within the AWS Free Tier.

You have N GB raw log per day, and the daily cost estimation is as follows:

When you use AWS Lambda as a log processor:

- Lambda Cost = 260 ms per MB x 1024 MB x N GB/day x $0.0000000167 per ms
- Amazon S3 Storage Cost = $0.023 per GB x NGB/day x 4% (compression)

When you use OpenSearch Ingestion as a log processor:

- OSI Pipeline Cost = $0.24 per OCU per hour
- The maximum amount of Amazon S3 data 1 OCU can handle is around 20MB/s

The total estimated monthly cost for ingesting logs is:

Total Monthly Cost (Lambda as processor) = (Lambda Cost + Amazon S3 Storage Cost) x 30 days

Daily Log Volume	Daily Lambda Cost (USD)	Daily Amazon S3 Storage Cost (USD)	Monthly Cost (USD)
10	0.044	0.009	1.610
100	0.445	0.092	16.099
1000	4.446	0.920	160.986
5000	22.23	4.600	804.900

Total Monthly Cost (OpenSearch Ingestion as processor) = (OpenSearch Ingestion Cost + Amazon S3 Storage Cost) x 30 days

Daily Log Volume	Daily Lambda Cost (USD)	Daily Amazon S3 Storage Cost (USD)	Monthly Cost (USD)
10	5.760	0.001	173.1
100	5.760	0.009	175.5
1000	11.52	0.920	373.2
5000	34.56	4.600	1174.8

For Amazon RDS/Aurora logs and AWS Lambda Logs that deliver to CloudWatch Logs, apart from the Amazon S3 and Lambda costs listed previously, there is the additional cost of using Firehose (KDF) to subscribe to the CloudWatch Logs Stream and put them into an Amazon S3 bucket, and KDF is charging for a 5KB increment (less than 5KB per record is billed as 5KB).

Assuming Log size is 0.2 KB per record, then the daily KDF cost is estimated as follows:

- Firehose Cost = $0.029 per GB x N GB/day x (5KB/0.2 KB)

For example, for 1GB logs per day, the extra monthly cost of KDF is $21.75.

> **Important**
>
> If you want to save costs charged by Firehose, make sure you activate logs only when needed. For example, you can choose not to activate Amazon RDS general logs unless required.

Log ingestion through Amazon Kinesis Data Streams

This section is applicable to:

- AWS Services Logs including CloudFront real-time logs, CloudTrail logs (delivers to CloudWatch Logs), and VPC Flow Logs (delivers to CloudWatch Logs).

- Application Logs that use Amazon Kinesis Data Streams as the log buffer.

> **Important**
>
> The cost estimation does not include the logging cost of service. For example, CloudFront real-time logs are charged based on the number of log lines generated ($0.01 for every 1,000,000 log lines). There are also log delivery to CloudWatch charges for CloudTrail and VPC Flow Logs that enabled CloudWatch Logging. Check the service pricing for more details.

The cost estimation is based on the following assumptions and facts:

- The average log message size is 1 KB.
- The daily log volume is L GB.
- The Lambda processor memory is 1024 MB.
- Every Lambda invocation processes 1 MB logs.
- One Lambda invocation processes one shard of Kinesis, and Lambda can scale up to more concurrent invocations to process multiple shards.
- The Lambda runtime to process log less than 5 MB is 500ms.
- 30% additional shards are provided to handle traffic jitter.
- One Kinesis shard intake log size is = 1 MB /second x 3600 seconds per hour x 24 hours x 0.7 = 60.48 GB/day.
- The desired Kinesis Shard number S is = Round_up_to_next_integer(Daily log volume L / 60.48).

Based on the preceding assumptions, here is the daily cost estimation formula:

- Kinesis Shard Hour Cost = $0.015 / shard hour x 24 hours per day x S shards
- Kinesis PUT Payload Unit Cost = $0.014 per million units x 1 million per GB x L GB per day
- Lambda Cost = $0.0000000167 per 1ms x 500 ms per invocation x 1,000 invocations per GB x L GB per day

Total Monthly Cost = (Kinesis Shard Hour Cost + Kinesis PUT Payload Unit Cost + Lambda Cost) x 30 days

Daily Log Volume (GB)	Shards	Daily Kinesis Shard Hour Cost (USD)	Daily Kinesis PUT Payload Unit Cost (USD)	Daily Lambda Cost (USD)	Monthly Cost (USD)
10	1	0.36	0.14	0.0835	17.505
100	2	0.72	1.4	0.835	88.65
1000	17	6.12	14	8.35	854.1

Additional Features Cost

> **Note**
>
> You will not be charged if you do not use the additional features in the Centralized Logging with OpenSearch console.

Access Proxy

If you deploy the Access Proxy through Centralized Logging with OpenSearch, additional charges will apply. The total cost varies depending on the instance type and number of instances. As of this revision, the following are two examples for the cost estimation in the US East (N. Virginia) Region.

Example 1: Instance Type - t3.nano, Instance Number - 2

- EC2 cost = t3.nano 1Y RI All Upfront price $26.28 x 2 / 12 months = $4.38/month
- Amazon EBS Cost = Amazon EBS $0.1 GB/month x 8 GB x 2 = $1.6/month (The Amazon EBS attached to the EC2 instance is 8 GB)
- Elastic Load Balancing Cost = $0.0225 per APPLICATION LOAD BALANCER-hour x 720 hours/ month = $16.2/month

Total Monthly Cost = $4.38 EC2 Cost + $1.6 Amazon EBS Cost + $16.2 Elastic Load Balancing Cost = **$22.18**

Example 2: Instance Type - t3.large, Instance Number - 2

- EC2 Cost = t3.large 1Y RI All Upfront $426.612 x 2 / 12 months = $71.1/month
- Amazon EBS Cost = $0.1 GB/month x 8 GB x 2 = $1.6/month (The Amazon EBS attached to the EC2 instance is 8 GB)
- Elastic Load Balancing Cost = $0.0225 per APPLICATION LOAD BALANCER-hour x 720 hours/month = $16.2/month

Total Monthly Cost = $71.1 EC2 Cost + $1.6 Amazon EBS Cost + $16.2 Elastic Load Balancing Cost = **$88.9**

Amazon OpenSearch Service Alarms

If you deploy the alarms through Centralized Logging with OpenSearch, the Amazon CloudWatch Pricing will apply.

Pipeline Alarms

Log Type	Alarm Count	Number of Standard Resolution Alarm Metrics	Monthly Cost per Ingestion per Pipeline
AWS Service logs	4	0.1 USD	0.4 USD
Application logs	5	0.1 USD	0.5 USD

Pipeline Monitoring

Log processor

Assumptions:

- Deployment in the US East (N. Virginia) Region (us-east-1)
- A processor Lambda will be triggered every 60 seconds. The monthly metric put request number is 60 (requests) x 24 (hours) x 30 (days) = 43,200
- PutMetricData: 43,200 requests x 0.00001 USD = 0.432 USD
- There are 4 metrics for Service Logs (total logs, failed logs, loaded logs, excluded logs) and 3 metrics (total logs, failed logs, loaded logs) for Application logs

- Amazon CloudWatch Logs API = PutMetricData x Number of Metrics
- Amazon CloudWatch Logs Metric = Number of Metrics x 0.3

Log Type	Monthly Metric Put Request Number	Number of Metrics	Amazon CloudWatch Logs API	Amazon CloudWatch Logs Metric	Monthly Cost Per Source/Per Pipeline
AWS Service logs	43,200	4	1.728 USD	1.2 USD	2.928 USD
Application logs	43,200	3	1.296 USD	0.9 USD	2.196 USD

Fluent Bit

Assumptions:

- Deployment in the US East (N. Virginia) Region (us-east-1)
- There are 7 metrics: `FluentBitOutputProcRecords`, `FluentBitOutputProcBytes`, `FluentBitOutputDroppedRecords`, `FluentBitOutputErrors`, `FluentBitOutputRetriedRecords`, `FluentBitOutputRetriesFailed`, `FluentBitOutputRetries`. For more information, refer to the Monitoring section.
- Number of Metrics requested: an interval of 60 seconds to put logs from Fluent Bit to Amazon CloudWatch (60 requests in an hour). Monthly put requests are 60 (requests) x 24 (hours) x 30 (days) = 43,200
- PutMetricData: 43,200 requests x 0.00001 USD = 0.432 USD
- CloudWatch Logs API = PutMetricData x Number of Metrics x Number of Instances
- CloudWatch Logs Metric = Number of Metrics x 0.3

Number of EC2 Instances / EKS Nodes	Amazon CloudWatch Logs API	Amazon CloudWatch Logs Log Storage and Ingested (Calculated by AWS Pricing Calculator)	Amazon CloudWatch Logs Metric	Monthly Cost Per Source/Per Pipeline
1	3.024 USD	0.04 USD	2.1 USD	5.164 USD
10	30.24 USD	0.35 USD	2.1 USD	32.69 USD
100	302.4 USD	3.53 USD	2.1 USD	308.03 USD

Using Light Engine

The pricing model of Light Engine is on demand. The cost varies on the raw log volume and the query log volume processed per day. The following provides monthly cost estimation in three scenarios:

- 10 GB raw log, and query 50 GB per day
- 100 GB raw log, and query 500 GB per day
- 1TB raw log, and query 1TB per day

Amazon Service	Monthly Cost (USD) Raw log: 10GB daily Query: 50GB daily	Monthly Cost (USD) Raw log: 100GB daily Query: 300GB daily	Monthly Cost (USD) Raw log: 1TB daily Query: 1TB daily
Amazon S3	$1.49	$19.98	$148.99
AWS Lambda	$0.37	$0.73	$1.10
Amazon SQS	$0.00	$0.00	$0.00
Amazon DynamoDB	$3.79	$3.79	$3.79

Amazon Service	Monthly Cost (USD) Raw log: 10GB daily Query: 50GB daily	Monthly Cost (USD) Raw log: 100GB daily Query: 300GB daily	Monthly Cost (USD) Raw log: 1TB daily Query: 1TB daily
AWS Step Functions	$8.07	$16.14	$26.90
Amazon SNS	$0.18	$0.18	$0.18
Amazon Athena	$7.25	$43.51	$148.54
Amazon EC2	$29.20	$29.20	$29.20
Total	**$50.35**	**$113.53**	**$358.70**

Solution Console Cost

> **Note**
>
> AWS Step Functions, Amazon CloudWatch, AWS Systems Manager, and Amazon EventBridge are all within AWS Free Tier.

A web console is created automatically when you deploy the solution. Assume the visits to the console are 3,000 times in a month (30 days), it will incur the following cost:

Service	Monthly Cost (USD)
Amazon CloudFront (1GB Data Transfer Out to internet and 1GB Data Transfer Out to Origin)	0.25
Amazon S3	0.027
Amazon Cognito	0.05
AWS AppSync	0.01
Amazon DynamoDB	1.00
AWS Lambda	0.132

Service	Monthly Cost (USD)
Total	**1.469**

View cost in Cost Explorer

You can view the spending of the entire solution, or at the pipeline level. All pipelines provisioned through the web console will, by default, have the `CLOSolutionCostAnalysis` tag associated with them. When creating a log pipeline, you have the option to input additional tags, and those tags will be propagated to all resources associated with that pipeline. You can then use the tag to view the cost of a specific pipeline or a group of pipelines with the same tag.

To enable this function, you must activate user-defined cost allocation tags first, and then view the cost in AWS Cost Explorer.

Step 1: Activate user-defined cost allocation tags

For tags to appear on your billing reports, you must activate them. Your user-defined cost allocation tags represent the tag key, which you activate in the Billing and Cost Management console. Once you activate or deactivate the tag key, it will affect all tag values that share the same tag key. A tag key can have multiple tag values. For more information, see the AWS Billing and Cost Management and Cost Management API Reference.

1. Sign in to the AWS Management Console and open the AWS Billing and Cost Management and Cost Management console.
2. In the navigation pane, choose **Cost allocation tags**.
3. Select the tag key **CLOSolutionCostAnalysis** to activate.
4. Choose **Activate**.

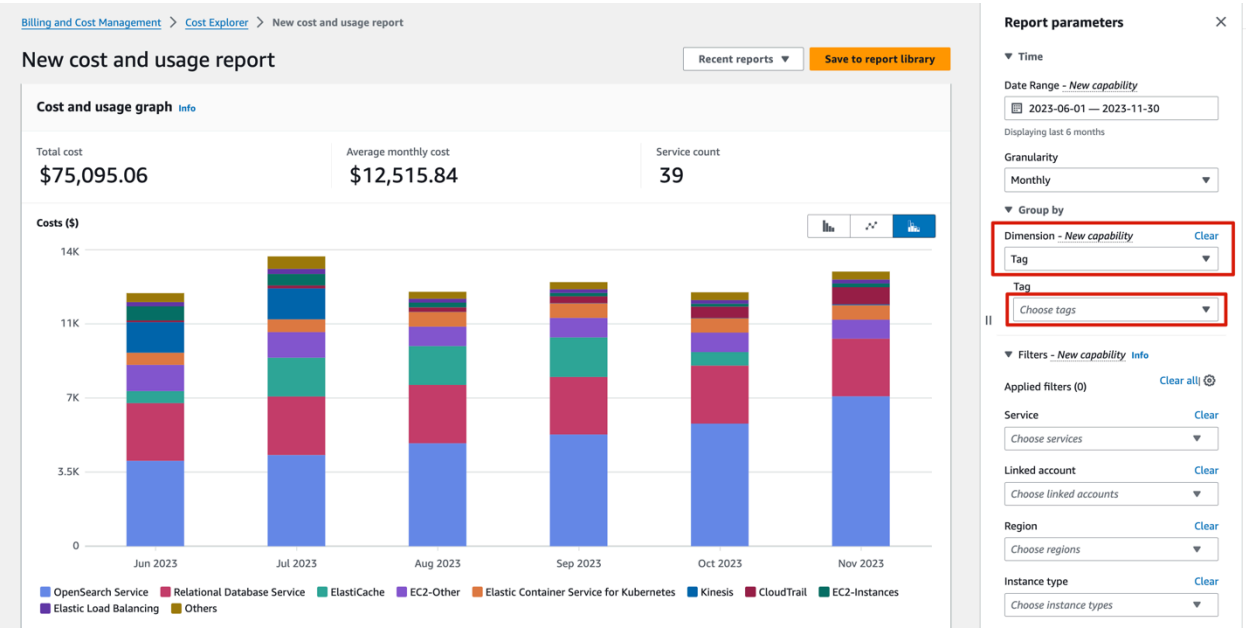
> **Note**
>
> After you create and apply user-defined tags to your resources, it can take up to 24 hours for the tag keys to appear on your cost allocation tags page for activation.

For an example of how tag keys appear in your billing report with cost allocation tags, see Viewing a cost allocation report.

Step 2: View Cost Explorer dashboard

1. Sign in to the AWS Management Console and open the AWS Billing and Cost Management and Cost Management console.
2. In the navigation pane, choose **Cost Explorer**.
3. Choose **Tag** as the displayed Dimension and select the specific tag **CLOSolutionCostAnalysis** to filter.
4. If your activated tag is absent in the dropdown list, this is because the activation process is still in progress. It can take up to 24 hours for tag keys to activate. Try later.

New cost and usage report example bar graph.



Security

When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS. This shared responsibility model reduces your operational burden because AWS operates, manages, and controls the components including the host operating system, the virtualization layer, and the physical security of the facilities in which the services operate. For more information about AWS security, see AWS Cloud Security.

IAM Roles

AWS Identity and Access Management (IAM) roles allow customers to assign granular access policies and permissions to services and users on the AWS Cloud. This solution creates IAM roles that grant the solution's AWS Lambda functions, AWS AppSync and Amazon Cognito access to create regional resources.

Security Groups

The security groups created in this solution are designed to control and isolate network traffic between the solution components. We recommend that you review the security groups and further restrict access as needed once the deployment is up and running.

Amazon CloudFront

This solution deploys a web console hosted in an Amazon Simple Storage Service (Amazon S3) bucket. To help reduce latency and improve security, this solution includes an Amazon CloudFront distribution with an origin access identity, which is a CloudFront user that provides public access to the solution's website bucket contents. For more information, refer to Restricting Access to Amazon S3 Content by Using an Origin Access Identity in the Amazon CloudFront Developer Guide.

Amazon EC2

This solution creates a NGINX based proxy, which will allow you to access the OpenSearch provisioned within the VPC environment. The NGINX is hosted using EC2 instances. We recommend you to use AWS Systems Manager Patch Manager to patch the instances periodically. Patch Manager is a capability of AWS Systems Manager that automates the process of patching managed nodes with updates. You can choose to show only a report of missing patches (a Scan operation), or to automatically install all patches that are missing (a Scan and install operation).

Automated deployment

Before you launch the solution, review the architecture, supported Regions, 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.

Prerequisites

Review all the considerations and make sure you have the following in the target Region you want to deploy the solution:

- At least one vacancy to create new VPCs, if you choose to launch with a new VPC.
- At least two vacant Elastic IP addresses, if you choose to launch with a new VPC.
- At least five vacant S3 buckets.

> **Important**
>
> The solution provisions an Amazon CloudFront distribution to serve its web console, with TLS 1.0 and 1.1 enabled by default. We recommend associating a custom domain and upgrading the TLS certificate to version 1.2 or higher after deployment.

Deployment in AWS Regions

Centralized Logging with OpenSearch provides two ways to authenticate and log into the Centralized Logging with OpenSearch console. For some AWS Regions where Amazon Cognito User Pool is not available (for example, Hong Kong), you must launch the solution with OpenID Connect provider.

- Launch with Amazon Cognito User Pool
- Launch with OpenID Connect

For more information about supported Regions, see Supported AWS Regions.

Deployment in AWS China Regions

AWS China Regions do not have a Amazon Cognito User Pool. Launch the solution with OpenID Connect.

- Launch with OpenID Connect

Launch with Amazon Cognito User Pool

Time to deploy: Approximately 15 minutes

Deployment Overview

Use the following steps to deploy this solution on AWS.

- Step 1. Launch the stack
- Step 2. Launch the web console

Step 1. Launch the stack

This AWS CloudFormation template automatically deploys the Centralized Logging with OpenSearch solution on AWS.

1. Sign in to the AWS Management Console and select the button to launch the AWS CloudFormation template.

	Launch in AWS Management Console
Launch with a new VPC	
Launch with an existing VPC	

2. The template is launched in the default Region after you log in to the console. To launch the Centralized Logging with OpenSearch 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 is shown 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 AWS STS quotas in the *AWS Identity and Access Management User Guide*.
5. Under **Parameters**, review the parameters for the template and modify them as necessary.
   - If you are launching the solution in a new VPC, this solution uses the following parameters:

Parameter	Default	Description
Admin User Email	`<Requires input>`	Specify the email of the Administrator. This email address will receive a temporary password to access the Centralized Logging with OpenSearch web console. You can create more users directly in the provisioned Amazon Cognito User Pool after launching the solution.

   - If you are launching the solution in an existing VPC, this solution uses the following parameters:

Parameter	Default	Description
Admin User Email	`<Requires input>`	Specify the email of the Administrator. This email address will receive a temporary password to access the Centralized Logging with OpenSearch web console. You can create more users directly in the provisioned Amazon Cognito User Pool after launching the solution.

Parameter	Default	Description
VPC ID	*<Requires input>*	Specify the existing VPC ID in which you are launching the Centralized Logging with OpenSearch solution.
Public Subnet IDs	*<Requires input>*	Specify the two public subnets in the selected VPC. The subnets must have routes pointing to an Internet Gateway.
Private Subnet IDs	*<Requires input>*	Specify the two private subnets in the selected VPC. The subnets must have routes pointing to a NAT Gateway.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Add new tag** and type in the following key and value:
   - Key: `CLOSolutionCostAnalysis`
   - Value: `CLOSolutionCostAnalysis`

   You can activate the `CLOSolutionCostAnalysis` tag after all resources have been successfully deployed. Choose **Next**.
8. On the **Review and create** page, review and confirm the settings. Select the box acknowledging that the template creates IAM resources.
9. Choose **Submit** 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 15 minutes.

Step 2. Launch the web console

After the stack is successfully created, this solution generates a CloudFront domain name that gives you access to the Centralized Logging with OpenSearch web console. Meanwhile, a generated temporary password (excluding the last digit) will be sent to your email address.

1. Sign in to the AWS CloudFormation console.
2. On the **Stacks** page, select the solution's stack.
3. Choose the **Outputs** tab and record the domain name.
4. Open the **WebConsoleUrl** using a web browser, and navigate to a sign-in page.
5. Enter the **Email** and the temporary password.
6. a. Set a new account password.
7. b. (Optional) Verify your email address for account recovery.
8. After the verification is complete, the system opens the Centralized Logging with OpenSearch web console.

Once you have logged into the Centralized Logging with OpenSearch console, you can import an Amazon OpenSearch Service domain and build log analytics pipelines.

Launch with OpenID Connect (OIDC)

Time to deploy: Approximately 30 minutes

Prerequisites

> **Important**
>
> The Centralized Logging with OpenSearch console is served via the CloudFront distribution, which is considered as an internet information service. If you are deploying the solution in **AWS China Regions**, the domain must have a valid ICP Recordal.

- A domain. You will use this domain to access the Centralized Logging with OpenSearch console (Required for AWS China Regions, optional for AWS Regions).

- An SSL certificate in AWS IAM. The SSL must be associated with the given domain. Follow the instructions in Upload SSL Certificate to IAM. Note that this is required for AWS China Regions, but is not recommended for all AWS Regions.
- ACM certificate in the US East (N. Virginia) Region (us-east-1). Note that this is not required for AWS China Regions, and is optional for AWS Regions.

Deployment Overview

Use the following steps to deploy this solution on AWS.

- Step 1. Create OIDC client
- Step 2. Launch the stack
- Step 3. Setup DNS Resolver
- Step 4. Launch the web console

Step 1. Create OIDC client

You can use different kinds of OpenID Connector (OIDC) providers. This section introduces Option 1 to Option 5.

- (Option 1) Using Amazon Cognito from another Region as an OIDC provider.
- (Option 2) Authing, which is an example of a third-party authentication provider in China.
- (Option 3) Keycloak, which is a solution maintained by AWS and can serve as an authentication identity provider.
- (Option 4) ADFS, which is a service offered by Microsoft.
- (Option 5) Other third-party authentication platforms such as Auth0.

Complete the following steps to create an OIDC client, and obtain the `client_id` and `issuer`.

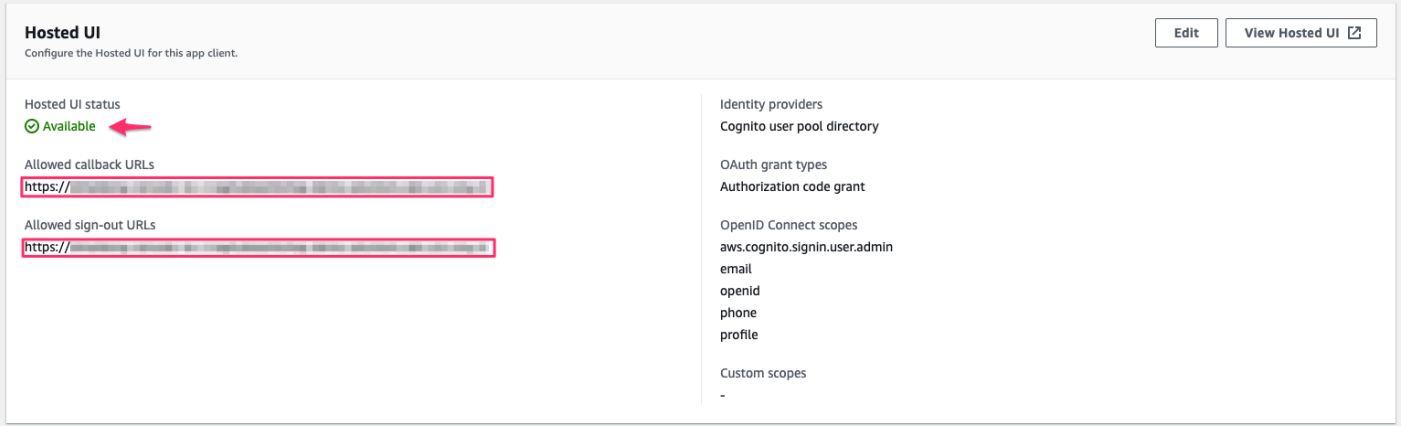
(Option 1) Using Amazon Cognito User Pool from another Region

You can use the Amazon Cognito User Pool in a supported AWS Standard Region as the OIDC provider.

1. Go to the Amazon Cognito console in an AWS Standard Region.

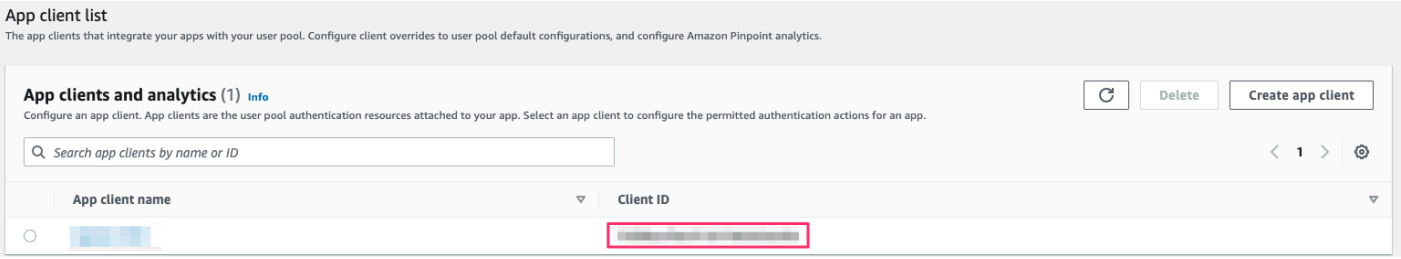
2. Set up the hosted UI with the Amazon Cognito console based on this guide.
3. Choose **Public client** when selecting the **App type**.
4. Enter the **Callback URL** and **Sign out URL** using your domain name for Centralized Logging with the OpenSearch console. If your hosted UI is set up, you should be able to see something like the following.

Hosted UI. Configure the hosted UI for this app client.

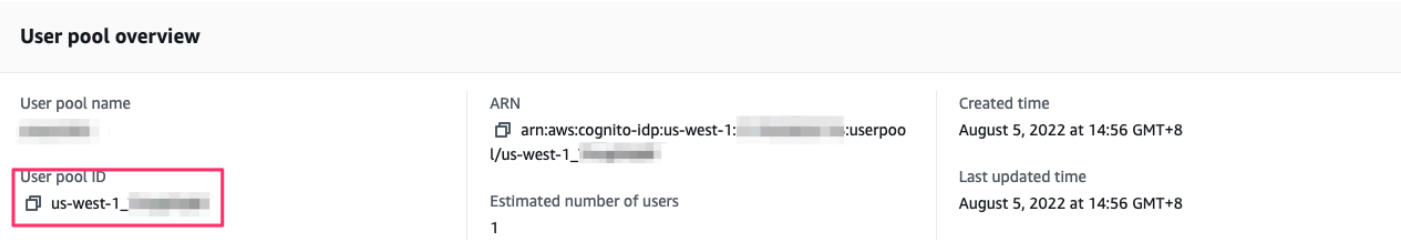


5. Save the App client ID, User pool ID and the AWS Region to a file, which will be used later.

App client list. Appl clients and analytics.



User pool overview screen.

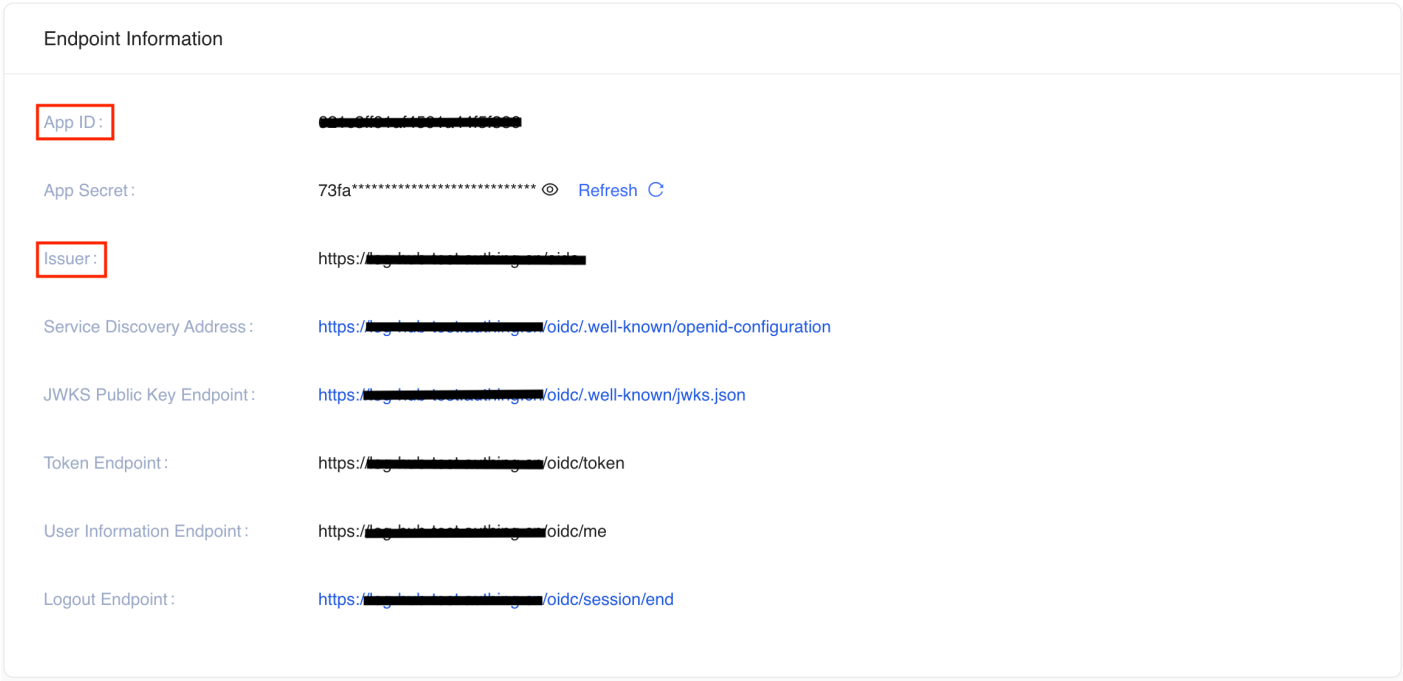


In Step 2. Launch the stack, the OidcClientID is the App client ID, and OidcProvider is `https://cognito-idp.${REGION}.amazonaws.com/${USER_POOL_ID}`.

(Option 2) Authing.cn OIDC client

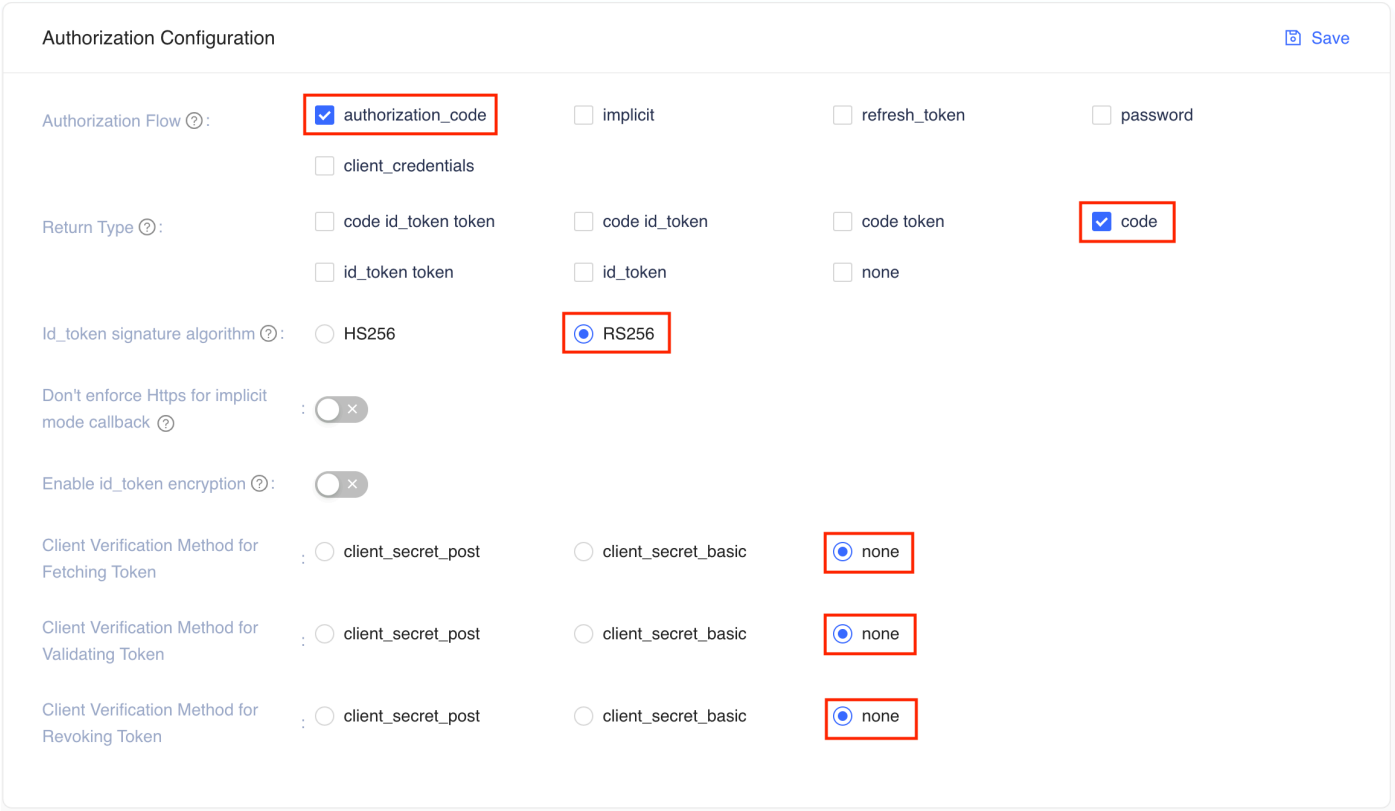
1. Go to the Authing console.
2. Create a user pool if you don't have one.
3. Select the user pool.
4. On the left navigation bar, select **Self-built App** under **Applications**.
5. Choose the **Create** button.
6. Enter the **Application Name** and **Subdomain**.
7. Save the App ID (that is, client_id) and Issuer to a text file from Endpoint Information, which will be used later.

   Endpoint information screen.



8. Update the `Login Callback URL` and `Logout Callback URL` to your IPC recorded domain name.
9. Set the **Authorization Configuration**.

   Authorization configuration screen.

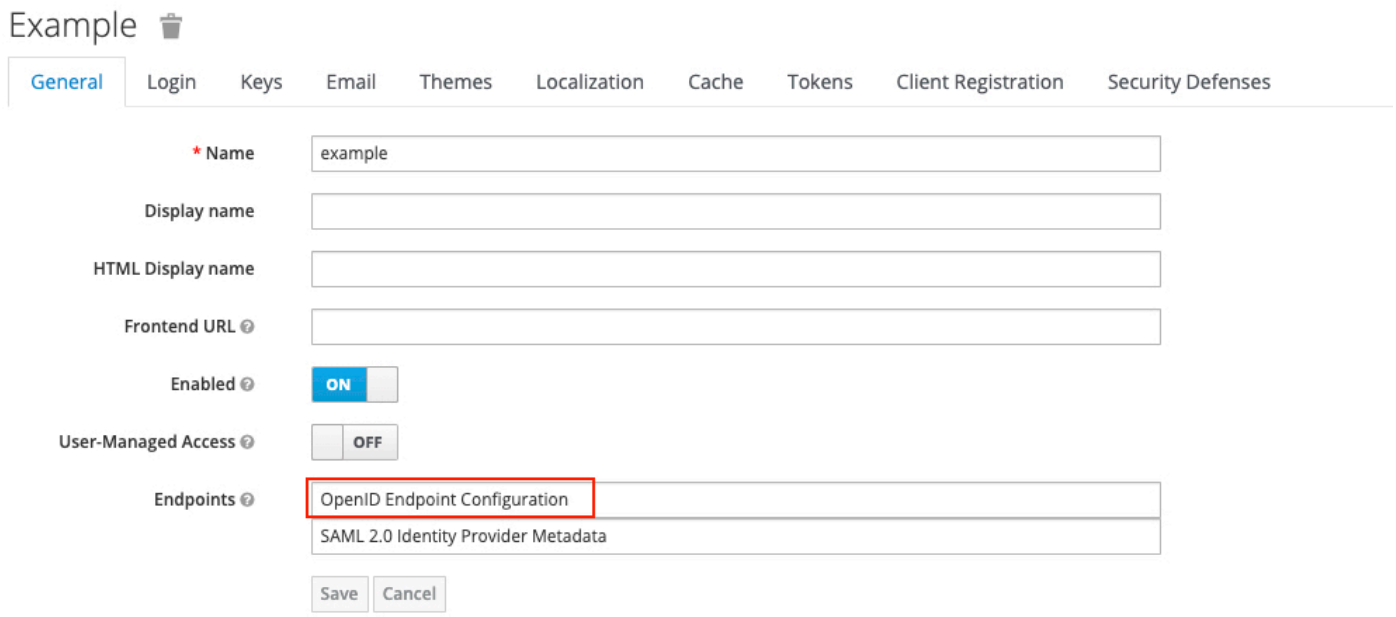


You have successfully created an authing self-built application.

(Option 3) Keycloak OIDC client

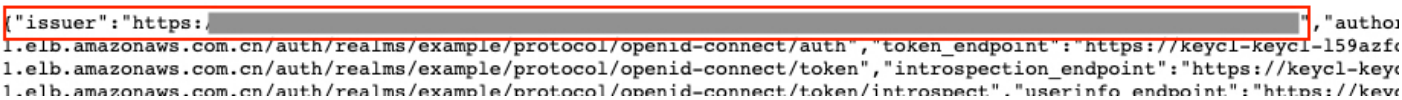
1. Deploy the Keycloak solution by following this guide.
2. Sign in to the Keycloak console.
3. On the left navigation bar, select **Add realm**. Skip this step if you already have a realm.
4. Go to the realm setting page. Choose **Endpoints**, and then **OpenID Endpoint Configuration** from the list.

   Example screen with General tab and fields for data input.



5. In the JSON file that opens up in your browser, record the **issuer** value, which will be used later.

   Example "issuer" value.



6. Go back to the Keycloak console and select **Clients** on the left navigation bar, and choose **Create**.
7. Enter a Client ID, which must contain 24 letters (case-insensitive) or numbers. Record the **Client ID**, which will be used later.
8. Change client settings. Enter `link:https://<Centralized Logging with OpenSearch Console domain>` in **Valid Redirect URIs**, and enter * and \+ in *Web Origins*.
9. In the **Advanced Settings**, set the **Access Token Lifespan** to at least 5 minutes.
10. Select **Users** on the left navigation bar.
11. Choose **Add user** and enter **Username**.
12. After the user is created, select **Credentials**, and enter **Password**.

The issuer value is `https://<KEYCLOAK_DOMAIN_NAME>/auth/realms/<REALM_NAME>`.

(Option 4) ADFS OpenID Connect Client

1. Make sure your ADFS is installed. For information about how to install ADFS, refer to this guide.
2. Make sure you can log in to the ADFS Sign On page. The URL should be `https://adfs.domain.com/adfs/ls/idpinitiatedSignOn.aspx`, and you must replace **adfs.domain.com** with your real ADFS domain.
3. Log on your Domain Controller, and open Active Directory Users and Computers.
4. Create a **Security Group** for Centralized Logging with OpenSearch Users, and add your planned Centralized Logging with OpenSearch users to this Security Group.
5. Log on to the ADFS server, and open **ADFS Management**.
6. Right click **Application Groups**, choose **Application Group**, and enter the name for the Application Group. Select **Web browser accessing a web application** option under **Client-Server Applications**, and choose **Next**.
7. Record the **Client Identifier** (client_id) under **Redirect URI**, enter your Centralized Logging with OpenSearch domain (for example, xx.domain.com), and choose **Add**, and then choose **Next**.
8. In the **Choose Access Control Policy** window, select **Permit specific group**, choose **parameters** under Policy part, add the created Security Group in Step 4, then choose **Next**. You can configure other access control policy based on your requirements.
9. Under the **Summary** window, choose **Next**, and choose **Close**.
10. Open the Windows PowerShell on ADFS Server, and run the following commands to configure ADFS to allow CORS for your planned URL.

```
Set-AdfsResponseHeaders -EnableCORS $true
Set-AdfsResponseHeaders -CORSTrustedOrigins https://<your-centralized-logging-with-opensearch-domain>
```

11. Under Windows PowerShell on ADFS server, run the following command to get the Issuer (issuer) of ADFS, which is similar to `https://adfs.domain.com/adfs`.

```
Get-ADFSProperties | Select IdTokenIssuer
```

Example IdTokenIssuer.

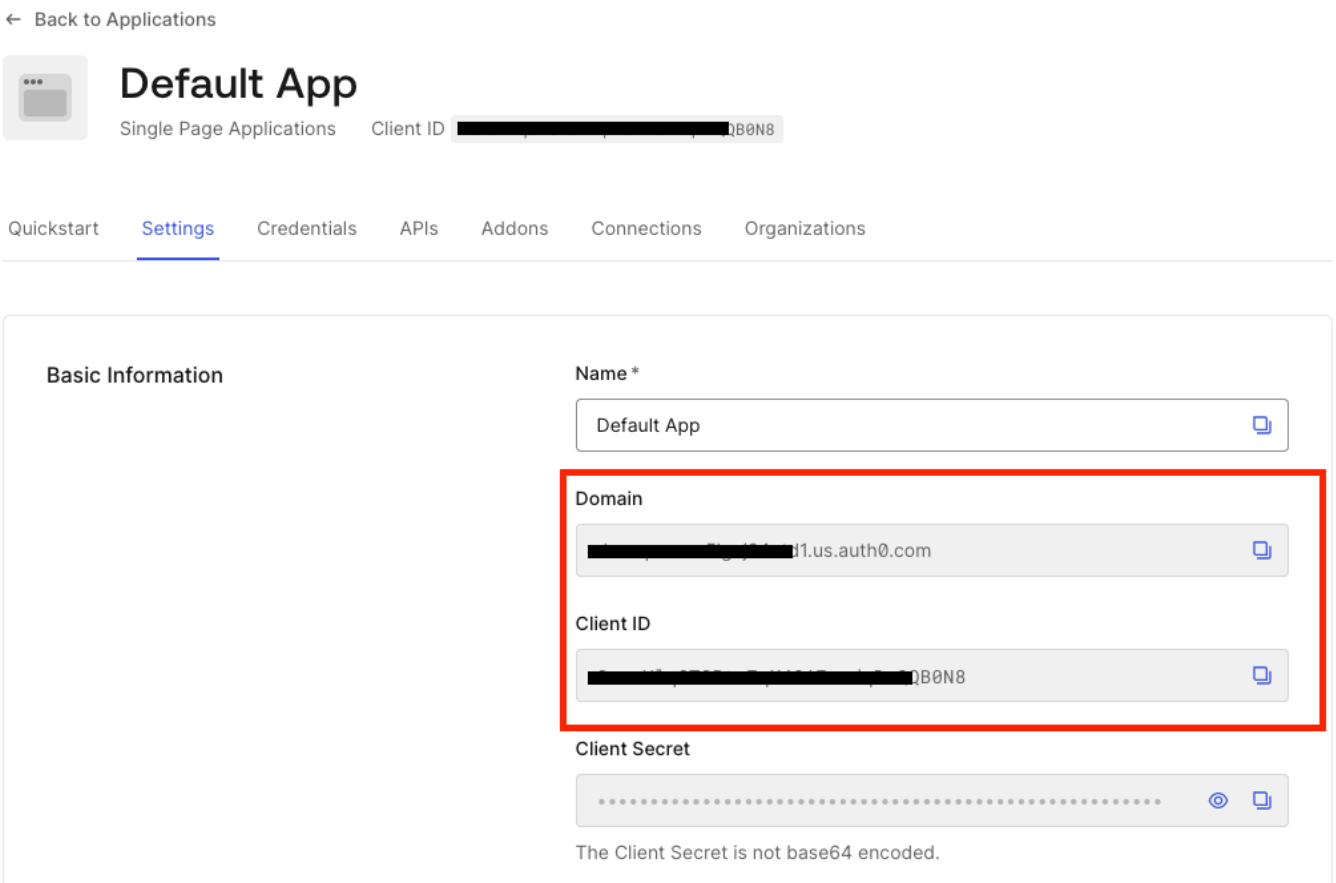
```
PS C:\Users\Administrator.AWS> Get-ADFSProperties | Select IdTokenIssuer

IdTokenIssuer
-------------
https://sts.aws.azeroth.zone/adfs
```

(Option 5) Other third-party authentication platforms such as Auth0

1. On Auth0, go to the Applications page
2. Click **+ Create Application**
3. Choose **Single Page Web Applications**
4. Go to Settings
5. Save `Domain` - this is your `OidcProvider` value
6. Save `Client ID` - this is your `OidcClientId` value

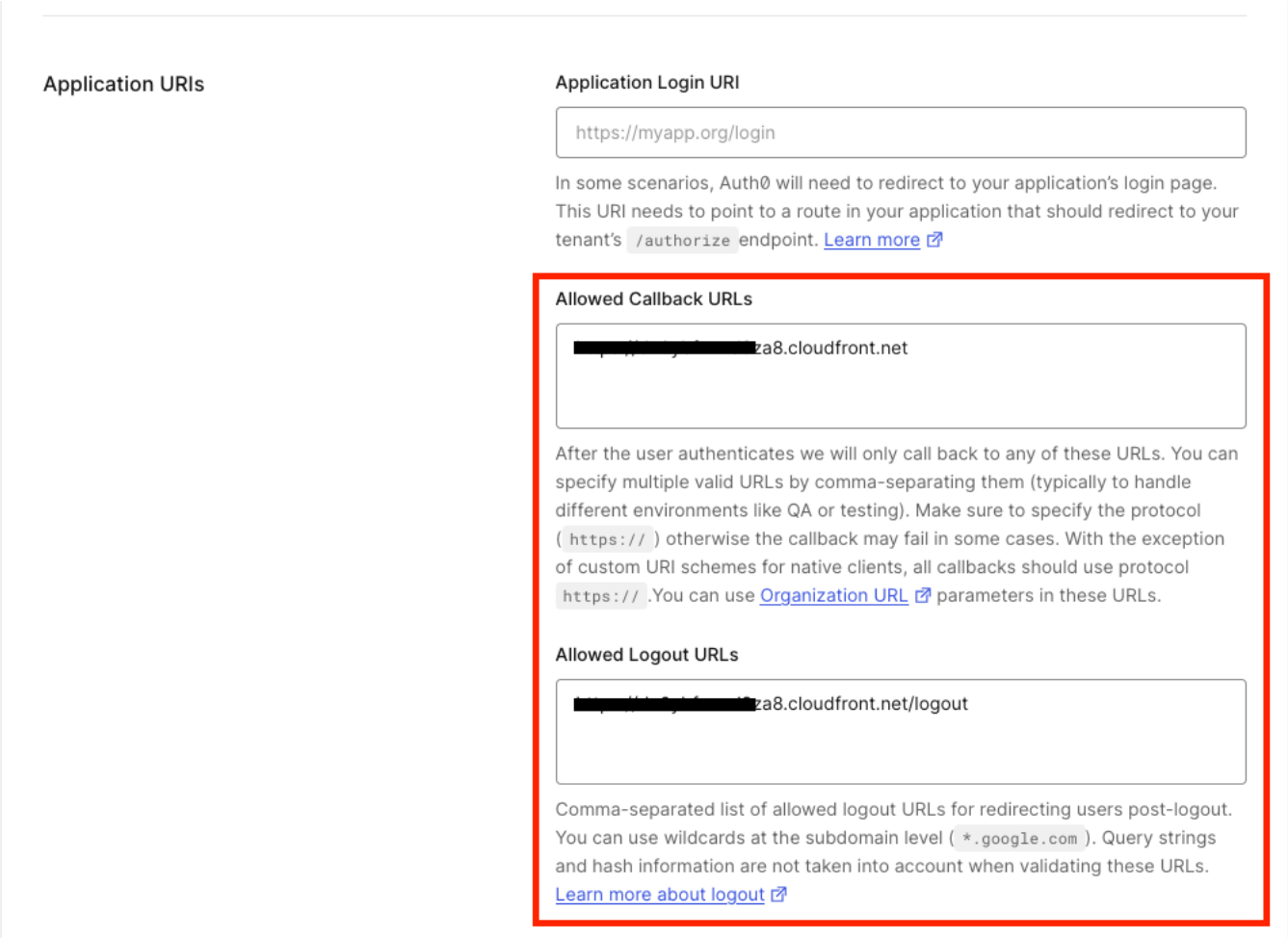
Example Auth0 application Domain & Client ID



7. Scroll down to **Application URIs**

8. Update `Allowed Callback URLs` and `Allowed Logout URLs`

Example Auth0 application redirect and logout URL's



9. Click **Save Changes**

Step 2. Launch the stack

> **Important**
>
> You can only have one active Centralized Logging with OpenSearch solution stack in one Region of an AWS account. If your deployment failed (for example, not meeting the requirements in prerequisites), make sure you have deleted the failed stack before retrying the deployment.

1. Sign in to the AWS Management Console and use the following buttons to launch the AWS CloudFormation template.

	Launch in AWS Management Console
Launch with a new VPC in AWS Regions	
Launch with an existing VPC in AWS Regions	
Launch with a new VPC in AWS China Regions	
Launch with an existing VPC in AWS China Regions	

2. The template is launched in the default Region after you log in to the console. To launch the Centralized Logging with OpenSearch 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 AWS STS quotas in the *AWS Identity and Access Management User Guide*.
5. Under **Parameters**, review the parameters for the template and modify them as necessary.
   - If you are launching the solution in a new VPC, this solution uses the following parameters:

Parameter	Default	Description
OidcClientId	`<Requires input>`	OpenID Connector client Id.

Parameter	Default	Description
OidcProvider	*<Requires input>*	OpenID Connector provider issuer. The issuer must begin with https://
Domain	*Optional input*	Custom domain for Centralized Logging with OpenSearch console. Do NOT add the HTTP(S) prefix.
IamCertificateID	*Optional input*	The ID of the SSL certificate in IAM. The ID is composed of 21 characters of capital letters and digits. Use the list-server-certificates command to retrieve the ID.
AcmCertificateArn	*Optional input*	ARN for ACM certificates requested (or imported) the certificate in the US East (N. Virginia) Region (us-east-1).

- If you are launching the solution in an existing VPC, this solution uses the following parameters:

Parameter	Default	Description
OidcClientId	*<Requires input>*	OpenID Connector client Id.
OidcProvider	*<Requires input>*	OpenID Connector provider issuer. The issuer must begin with https://
Domain	*Optional input*	Custom domain for Centralized Logging with OpenSearch console. Do NOT add the HTTP(S) prefix.

Parameter	Default	Description
IamCertificateID	*Optional input*	The ID of the SSL certificate in IAM. The ID is composed of 21 characters of capital letters and digits. Use the list-server-certificates command to retrieve the ID.
AcmCertificateArn	*Optional input*	ARN for ACM certificates requested (or imported) the certificate in the US East (N. Virginia) Region (us-east-1).
VPC ID	`<Requires input>`	Specify the existing VPC ID in which you are launching the solution.
Public Subnet IDs	`<Requires input>`	Specify the two public subnets in the selected VPC. The subnets must have routes pointing to an Internet Gateway.
Private Subnet IDs	`<Requires input>`	Specify the two private subnets in the selected VPC. The subnets must have routes pointing to an NAT Gateway.

IMPORTANT: * If you are deploying the solution in AWS China Regions, you must enter **Domain** and **IamCertificateID**. * If you are deploying the solution in AWS Regions:

+

- When a custom domain name is required, you must enter **Domain** and **AcmCertificateArn**.
- If no custom domain name is required, leave it blank for **Domain**, **IamCertificateID**, and **AcmCertificateArn**.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Add new tag** and type in the following key and value:

   a. Key: `CLOSolutionCostAnalysis`

   b. Value: `CLOSolutionCostAnalysis`

   You can activate the `CLOSolutionCostAnalysis` tag after all resources have been successfully deployed. Choose **Next**.
8. On the **Review and create** page, review and confirm the settings. Check the box acknowledging that the template creates IAM resources.
9. Choose **Submit** 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 15 minutes.

Step 3. Setup DNS Resolver

This solution provisions a CloudFront distribution that gives you access to the Centralized Logging with OpenSearch console.

1. Sign in to the AWS CloudFormation console.
2. Select the solution's stack.
3. Choose the **Outputs** tab.
4. Obtain the **WebConsoleUrl** as the endpoint.
5. Create a CNAME record in the DNS resolver, which points to the endpoint address.

Step 4. Launch the web console

> **Important**
>
> Your login credentials are managed by the OIDC provider. Before signing in to the Centralized Logging with OpenSearch console, make sure you have created at least one user in the OIDC provider's user pool.

1. Use the previous assigned CNAME to open the **OIDC Customer Domain URL** using a web browser.
2. Choose Sign in to Centralized Logging with OpenSearch, and navigate to OIDC provider.
3. Enter sign-in credentials. You may be requested to change your default password for first-time login, which depends on your OIDC provider's policy.
4. After the verification is complete, the system opens the Centralized Logging with OpenSearch web console.

Once you have logged into the Centralized Logging with OpenSearch console, you can import an Amazon OpenSearch Service domain and build log analytics pipelines.

Getting started

After deploying the solution, refer to this section to quickly learn how to use Centralized Logging with OpenSearch for log ingestion (AWS CloudTrail logs as an example), and log visualization.

You can also choose to start with Domain management, then build AWS Service Log Analytics Pipelines and Application Log Analytics Pipelines.

Steps

- Step 1: Import an Amazon OpenSearch Service domain. Import an existing Amazon OpenSearch Service domain into the solution.
- Step 2: Create Access Proxy. Create a public access proxy, which allows you to access the templated dashboard from anywhere.
- Step 3: Ingest CloudTrail Logs. Ingest CloudTrail logs into the specified Amazon OpenSearch Service domain.
- Step 4: Access built-in dashboard. View the dashboard of CloudTrail logs.

Step 1: Import an Amazon OpenSearch Service domain

To use the Centralized Logging with OpenSearch solution for the first time, you must import Amazon OpenSearch Service domains first.

Centralized Logging with OpenSearch supports Amazon OpenSearch Service domain with fine-grained access control enabled within a VPC only.

> **Important**
>
> Currently, Centralized Logging with OpenSearch supports Amazon OpenSearch Service with OpenSearch 1.3 or later.

Prerequisites

At least one Amazon OpenSearch Service domain within VPC. If you don't have an Amazon OpenSearch Service domain yet, you can create an Amazon OpenSearch Service domain within VPC. See Launching your Amazon OpenSearch Service domains within a VPC.

Steps

Use the following procedure to import an Amazon OpenSearch Service domain through the Centralized Logging with OpenSearch console.

1. Sign in to the Centralized Logging with OpenSearch console (see instructions for accessing the console using Amazon Cognito user pool or OIDC).
2. In the navigation pane, under **Domains**, choose **Import OpenSearch Domain**.
3. On the **Step 1. Select domain** page, choose a domain from the dropdown list.
4. Choose **Next**.
5. On the **Step 2. Configure network** page, under **Network creation**, choose **Automatic**. If your Centralized Logging with OpenSearch and OpenSearch domains reside in two different VPCs, the *Automatic* mode will create a VPC Peering Connection between them, and update route tables. See details in Set up VPC Peering.
6. On the Step 3. Create tags page, choose Import.

Step 2: Create Access Proxy

> **Note**
>
> Access proxy is optional and it incurs additional cost. If you can connect to Amazon OpenSearch Service's VPC (such as through a VPN connection), you don't need to activate an access proxy. You must use it only if you want to connect to the Amazon OpenSearch Service dashboard from the public internet.

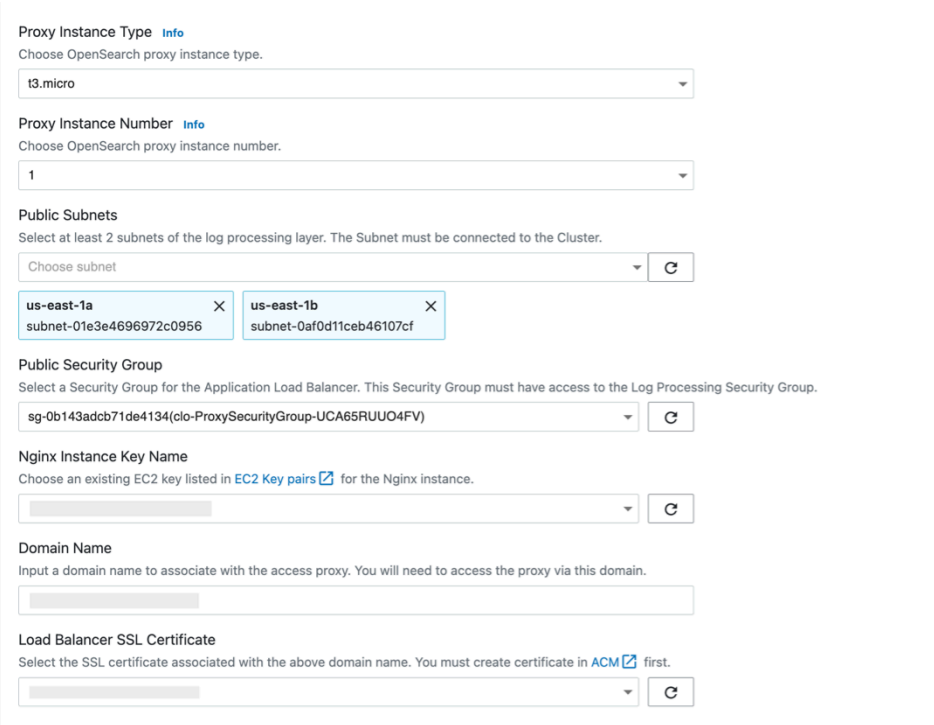
You can create a NGINX proxy and create a DNS record pointing to the proxy, so that you can access the Amazon OpenSearch Service dashboard securely from a public network. For more information, refer to Access Proxy.

Create a NGINX proxy

1. Sign in to the Centralized Logging with OpenSearch console (see instructions for accessing the console using Amazon Cognito user pool or OIDC).
2. In the navigation pane, under **Domains**, choose **OpenSearch domains**.
3. Select the domain from the table.

4. Under General configuration, choose **Enable** at the Access Proxy label.
5. On the **Create access proxy** page, under **Public access proxy**, select at least 2 subnets that contain `CLVpc/DefaultVPC/publicSubnetX` for the **Public Subnets**.
6. For **Public Security Group**, choose the Security Group that contains `ProxySecurityGroup`.
7. Choose the NGINX Instance Key Name.
8. Enter the **Domain Name**.
9. Choose the associated **Load Balancer SSL Certificate that** applies to the domain name.

NGINX Instance key name. Specify the EC2 key name of the NGINX proxy.



10. Choose **Create**.

After provisioning the proxy infrastructure, you must create an associated DNS record in your DNS resolver. The following introduces how to find the Application Load Balancer domain, and then create a CNAME record pointing to this domain.

Create a DNS record

1. Sign in to the Centralized Logging with OpenSearch console (see instructions for accessing the console using Amazon Cognito user pool or OIDC).
2. In the navigation pane, under **Domains**, choose **OpenSearch domains**.

3. Select the domain from the table.
4. Choose the **Access Proxy** tab. Find **Load Balancer Domain**, which is the Application Load Balancer domain.
5. Go to the DNS resolver, and create a CNAME record pointing to this domain. If your domain is managed by Amazon Route 53, refer to Creating records by using the Amazon Route 53 console.

Step 3: Ingest AWS CloudTrail Logs

You can build a log analytics pipeline to ingest AWS CloudTrail logs.

> **Important**
>
> Make sure your CloudTrail and Centralized Logging with OpenSearch are in the same AWS Region.

1. Sign in to the Centralized Logging with OpenSearch Console (see instructions for accessing the console using Amazon Cognito user pool or OIDC).
2. In the navigation pane, select **AWS Service Log Analytics Pipelines**.
3. Choose **Create a log ingestion**.
4. In the **AWS Services** section, choose `AWS CloudTrail`.
5. Choose **Next**.
6. Under **Specify settings**, for **Trail**, select one from the dropdown list.
7. Choose **Next**.
8. In the **Specify OpenSearch domain** section, select the imported domain for the **Amazon OpenSearch Service domain**.
9. Choose `Yes` for **Sample dashboard**.
10. Keep default values and choose **Next**.
11. Choose **Create**.

Step 4: Access the dashboard

After the DNS record takes effect, you can access the built-in dashboard from anywhere via proxy.

1. Enter the domain of the proxy in your browser. Alternatively, click the **Link** button under **Access Proxy** in the **General Configuration** section of the domain.
2. Enter your credentials to log in to the Amazon OpenSearch Service Dashboard.
3. Click the username icon of the Amazon OpenSearch Service dashboard from the top right corner.
4. Choose `Switch Tenants`.
5. On the **Select your tenant** page, choose **Global**, and click **Confirm**.
6. On the left navigation panel, choose **Dashboards**.
7. Choose the dashboard created automatically and start to explore your data.

Domain management

This chapter describes how to manage Amazon OpenSearch Service domains through the Centralized Logging with OpenSearch console. An Amazon OpenSearch Service domain is synonymous with an Amazon OpenSearch Service cluster.

In this chapter, you will learn:

- Import and remove an Amazon OpenSearch Service Domain
- Create an access proxy
- Create recommended alarms

You can read the Getting Started chapter first and walk through the basic steps for using the Centralized Logging with OpenSearch solution.

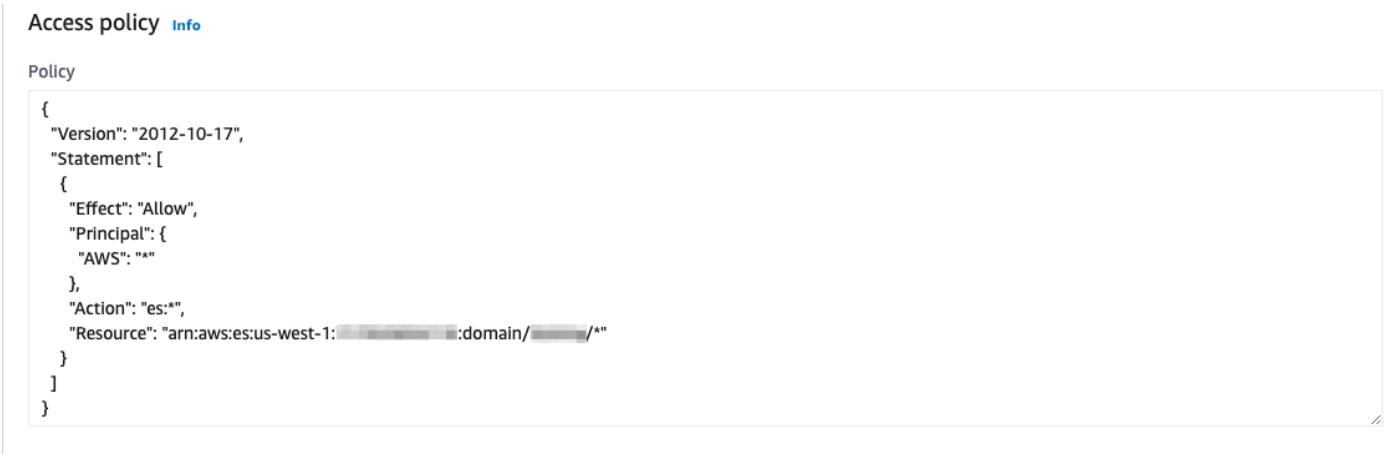
Domain Operations

Once logged into the Centralized Logging with OpenSearch console, you can import an Amazon OpenSearch Service domain.

Prerequisites

1. Centralized Logging with OpenSearch supports Amazon OpenSearch Service, and engine version OpenSearch 1.3 or later.
2. Centralized Logging with OpenSearch supports OpenSearch clusters within VPC. If you don't have an Amazon OpenSearch Service domain yet, you can create an Amazon OpenSearch Service domain within VPC. See Launching your Amazon OpenSearch Service domains within a VPC.
3. Centralized Logging with OpenSearch supports OpenSearch clusters with fine-grained access control only. In the security configuration, the Access policy should look like the following image:

   Sample access policy.



Import an Amazon OpenSearch Service Domain

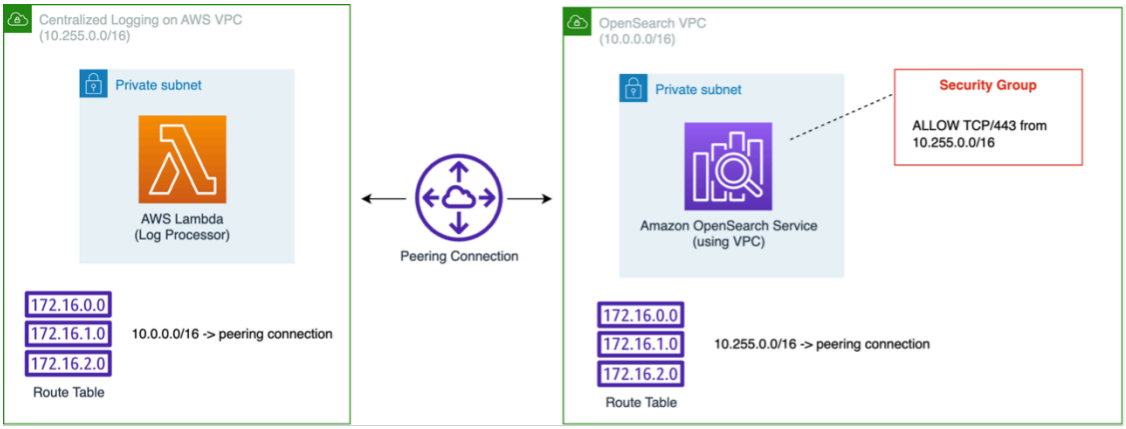
1. Sign in to the Centralized Logging with OpenSearch console.
2. In the left navigation panel, under **Domains**, choose **Import OpenSearch Domain**.
3. On the **Select domain** page, choose a domain from the dropdown list. The dropdown list will display only domains in the same Region as the solution.
4. Choose **Next**.
5. On the **Configure network** page, under **Network creation**, choose **Manual** and choose **Next**; or choose **Automatic**, and go to step 9.
6. Under **VPC**, choose a VPC from the list. By default, the solution creates a standalone VPC, and you can choose the one named `LogHubVpc/DefaultVPC`. You can also choose the same VPC as your Amazon OpenSearch Service domains.
7. Under **Log Processing Subnet Group**, select at least 2 subnets from the dropdown list. By default, the solution creates two private subnets. You can choose subnets named `LogHubVpc/DefaultVPC/privateSubnet1` and `LogHubVpc/DefaultVPC/privateSubnet2`.
8. Under **Log Processing Security Group**, select one from the dropdown list. By default, the solution creates one Security Group named `ProcessSecurityGroup`.
9. On the **Create tags** page, add tags if needed.
10. Choose **Import**.

Set up VPC Peering

By default, the solution creates a standalone VPC. You must create VPC Peering to allow the log processing layer to have access to your Amazon OpenSearch Service domains.

> **Note**
>
> Automatic mode will create VPC peering and configure route table automatically. You do not need to set up VPC peering again.

VPC peering connecting the solution and an OpenSearch VPC.



Follow this section to create VPC peering, update your security group, and update route tables.

Create VPC Peering Connection

1. Sign in to the Centralized Logging with OpenSearch console.
2. In the left navigation panel, under **Domains**, select **OpenSearch Domains**.
3. Find the domain that you imported and select the domain name.
4. Choose the **Network** tab.
5. Copy the VPC ID in both sections **OpenSearch domain network** and **Log processing network**. You will create a Peering Connection between these two VPCs.
6. Navigate to VPC Console Peering Connections.
7. Select the Create peering connection button.
8. On the **Create peering connection** page, enter a name.

9. For the Select a local VPC to peer with, VPC ID (Requester), select the VPC ID of the Log processing network.
10. For the Select another VPC to peer with, VPC ID (Accepter), select the VPC ID of the OpenSearch domain network.
11. Choose **Create peering connection**, and navigate to the peering connection detail page.
12. Choose the **Actions** button and choose **Accept request**.

Update Route Tables

1. Go to the Centralized Logging with OpenSearch console.
2. In the **OpenSearch domain network** section, choose the subnet under **Availability Zone and Subnets** to open the subnet console in a new tab.
3. Select the subnet, and choose the **Route table** tab.
4. Select the associated route table of the subnet to open the route table configuration page.
5. Select the **Routes** tab, and choose **Edit routes**.
6. Add a route 10.255.0.0/16 (the CIDR of Centralized Logging with OpenSearch, if you created the solution with existing VPC, change this value) pointing to the Peering Connection you created.
7. Go back to the Centralized Logging with OpenSearch console.
8. Choose the VPC ID under the **OpenSearch domain network** section.
9. Select the VPC ID on the VPC Console and find its **IPv4 CIDR**.
10. On the Centralized Logging with OpenSearch console, in the **Log processing network** section, choose the subnets under **Availability Zone and Subnets** to open the subnets in new tabs.
11. Repeat step 3, 4, 5, 6 to add an opposite route. Namely, configure the IPv4 CIDR of the OpenSearch VPC to point to the Peering Connection. You must repeat the steps for each subnet of the Log processing network.

Update Security Group of OpenSearch Domain

1. On the Centralized Logging with OpenSearch console, under the **OpenSearch domain network** section, select the Security Group ID in **Security Groups** to open the Security Group in a new tab.
2. On the console, select **Edit inbound rules**.

3. Add the rule `ALLOW TCP/443 from 10.255.0.0/16` (the CIDR of Centralized Logging with OpenSearch, if you created Centralized Logging with OpenSearch with existing VPC, change this value).
4. Choose **Save rules**.

> **Note**
>
> If you prefer to use Transit Gateway rather than VPC peering for connectivity between OpenSearch domain VPC and the solution VPC, select manual network create option during domain import. After creation, you'll need to configure your route tables to direct traffic through the Transit Gateway instead of VPC peering connection.

Remove an Amazon OpenSearch Service domain

If needed, you can remove the Amazon OpenSearch Service domains.

> **Important**
>
> Removing the domain from Centralized Logging with OpenSearch will NOT delete the Amazon OpenSearch Service domain in your AWS account. It will NOT impact any existing log analytics pipelines.

1. Sign in to the Centralized Logging with OpenSearch console.
2. In the navigation pane, under **Domains**, choose **OpenSearch Domains**.
3. Select the domain from the table.
4. Choose **Remove**.
5. In the confirmation dialog box, choose **Remove**.

Access proxy

By default, an Amazon OpenSearch Service domain within VPC cannot be accessed from the internet. Centralized Logging with OpenSearch creates a highly available NGINX cluster that allows you to access the OpenSearch Dashboards from the internet. Alternatively, you can choose to

access the Amazon OpenSearch Service domains using SSH Tunnel. Refer to the Access proxy architecture for more implementation details.

This section covers the following:

1. Create a proxy
2. Create an associated DNS record
3. Access Amazon OpenSearch Service via proxy
4. Delete a proxy

Create a proxy

You can create the NGINX-based proxy using the Centralized Logging with OpenSearch console or by deploying a standalone CloudFormation stack.

Prerequisites

- Make sure an Amazon OpenSearch Service **domain** within VPC is available.
- The domain associated **SSL certificate** is created or uploaded in AWS Certificate Manager (ACM).
- Make sure you have the EC2 private key (.pem) file.

(Option 1) Using **the Centralized Logging with OpenSearch console**

1. Log in to the Centralized Logging with OpenSearch console.
2. In the navigation pane, under **Domains**, choose **OpenSearch domains**.
3. Select the domain from the table.
4. Under **General configuration**, choose **Enable** at the **Access Proxy** label.

> **Note**
>
> Once the access proxy is enabled, a link to the access proxy will be available.

5. On the **Create access proxy** page, choose the **Proxy Instance Type** and **Proxy Instance Number**.
6. Under **Public access proxy**, select at least 2 subnets for **Public Subnets**. You can choose 2 public subnets named `CLVPC/DefaultVPC/publicSubnetX`, which are created by Centralized Logging with OpenSearch by default.

7. Choose a Security Group of the Application Load Balancer in **Public Security Group**. You can choose a security group named `ProxySecurityGroup`, which is created by Centralized Logging with OpenSearch default.
8. Choose the NGINX Instance Key Name.
9. Enter the **Domain Name**.
10. Choose **Load Balancer SSL Certificate** associated with the domain name.
11. Choose **Create**.

(Option 2) Using the CloudFormation stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - NGINX access proxy* solution in the AWS Cloud.

1. Log in to the AWS Management Console and select the button to launch the AWS CloudFormation template.

You can also download the template as a starting point for your own implementation. . To launch the stack in a different AWS Region, use the Region selector in the console navigation bar. . On the **Create stack** page, verify that the correct template URL shows in the **Amazon S3 URL** text box and choose **Next**. . On the **Specify stack details** page, assign a name to your stack. . Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.

+

Parameter	Default	Description
VPCId	*<Requires input>*	The VPC to deploy the NGINX proxy resources, for example, `vpc-bef13dc7` .
PublicSubnetIds	*<Requires input>*	The public subnets where Elastic Load Balancing is deployed. You must select

Parameter	Default	Description
		at least two public subnets, for example, `subnet-12345abc` , `subnet-54321cba` .
ELBSecurityGroupId	*<Requires input>*	The security group being associated with the Elastic Load Balancing, for example, `sg-123456` .
ELBDomain	*<Requires input>*	The custom domain name of the Elastic Load Balancing , for example, `dashboard.example.com` .
ELBDomainCertificateArn	*<Requires input>*	The SSL certificate ARN associated with the ELBDomain. The certificate must be created from ACM.
PrivateSubnetIds	*<Requires input>*	The private subnets where NGINX instances are deployed. You must select at least two private subnets, for example, `subnet-12345abc` , `subnet-54321cba` .
NginxSecurityGroupId	*<Requires input>*	The security group associated with the NGINX instances . The security group must allow access from Elastic Load Balancing security group.
KeyName	*<Requires input>*	The PEM key name of the NGINX instances.

Parameter	Default	Description
EngineType	`OpenSearch`	The engine type of the OpenSearch. Select OpenSearch.
Endpoint	*`<Requires input>`*	The OpenSearch endpoint, for example, `vpc-your_opensearch_domain_name-xcvgw6uu2o6zafsiefxubwuohe.us-east-1.es.amazonaws.com .`
CognitoEndpoint	*Optional input*	The Amazon Cognito User Pool endpoint URL of the OpenSearch domain, for example, mydomain.auth.us-east-1.amazoncognito.com. Leave empty if your OpenSearch domain is not authenticated through Amazon Cognito User Pool.

1. Choose **Next**.
2. On the **Configure stack options** page, choose **Next**.
3. On the **Review and create** page, review and confirm the settings. Check the box acknowledging that the template creates IAM resources.
4. Choose **Submit** 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 15 minutes.

Recommended Proxy Configuration

The following table provides a list of recommended proxy configuration examples for different number of concurrent users. You can create a proxy according to your own use cases.

Number of Concurrent Users	Proxy Instance Type	Number of Proxy Instances
4	t3.nano	1
6	t3.micro	1
8	t3.nano	2
10	t3.small	1
12	t3.micro	2
20	t3.small	2
25	t3.large	1
50+	t3.large	2

Create an associated DNS record

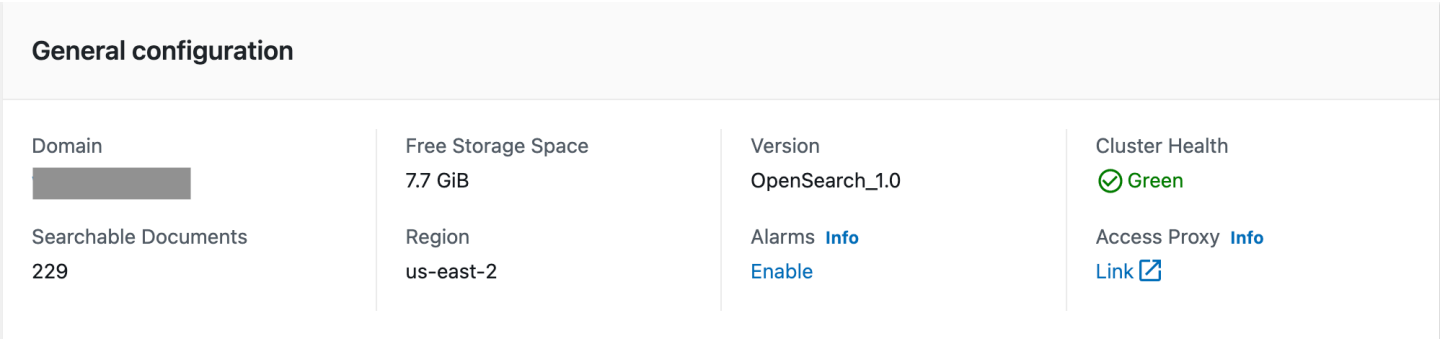
After provisioning the proxy infrastructure, you must create an associated DNS record in your DNS resolver. The following introduces how to find the Application Load Balancer domain, and then create a CNAME record pointing to this domain.

1. Log in to the Centralized Logging with OpenSearch console.
2. In the navigation pane, under **Domains**, choose **OpenSearch domains**.
3. Select the domain from the table.
4. Choose the **Access Proxy** tab. You can see the **Load Balancer Domain**, which is the Application Load Balancer domain.
5. Go to the DNS resolver, create a CNAME record pointing to this domain. If your domain is managed by Amazon Route 53, refer to Creating records by using the Amazon Route 53 console.

Access Amazon OpenSearch Service via proxy

After the DNS record takes effect, you can access the Amazon OpenSearch Service built-in dashboard from anywhere via proxy. You can enter the domain of the proxy in your browser, or choose the **Link** button under **Access Proxy** in the **General Configuration** section.

Example General configuration screen.



Delete a Proxy

1. Log in to the Centralized Logging with OpenSearch console.
2. In the navigation pane, under **Domains**, choose **OpenSearch domains**.
3. Select the domain from the table.
4. Choose the **Access Proxy** tab.
5. Choose the **Delete**.
6. On the confirmation prompt, choose **Delete**.

Domain alarms

Amazon OpenSearch Service provides a set of recommended CloudWatch alarms to monitor the health of Amazon OpenSearch Service domains. Centralized Logging with OpenSearch helps you to create the alarms automatically, and send a notification to your email (or SMS) via Amazon SNS.

Create alarms

(Option 1) Using the Centralized Logging with OpenSearch console

1. Log in to the Centralized Logging with OpenSearch console.
2. In the navigation pane, under **Domains**, choose **OpenSearch domains**.

3. Select the domain from the table.
4. Under **General configuration**, choose **Enable** at the **Alarms label**.
5. Enter the Email.
6. Choose the alarms that you want to create and adjust the settings if necessary.
7. Choose **Create**.

(Option 2) Using the CloudFormation stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - Alarms* solution in the AWS Cloud.

1. Log in to the AWS Management Console and select the button to launch the AWS CloudFormation template.

You can also download the template as a starting point for your own implementation. . To launch the stack in a different AWS Region, use the Region selector in the console navigation bar. . On the **Create stack** page, verify that the correct template URL shows in the **Amazon S3 URL** text box and choose **Next**. . On the **Specify stack details** page, assign a name to your stack. . Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.

+

Parameter	Default	Description
Endpoint	*<Requires input>*	The endpoint of the OpenSearch domain, for example, `vpc-your_opensearch_domain_name-xcvgw6uu2o6zafsiefxubwuohe.us-east-1.es.amazonaws.com` .

Parameter	Default	Description
DomainName	*<Requires input>*	The name of the OpenSearch domain.
Email	*<Requires input>*	The notification email address. Alarms will be sent to this email address via Amazon SNS.
ClusterStatusRed	Yes	Whether to enable alarm when at least one primary shard and its replicas are not allocated to a node.
ClusterStatusYellow	Yes	Whether to enable alarm when at least one replica shard is not allocated to a node.
FreeStorageSpace	10	Whether to enable alarm when a node in your cluster is down to the free storage space you entered in GiB. We recommend setting it to 25% of the storage space for each node. 0 means that the alarm is disabled.
ClusterIndexWritesBlocked	1	Index writes blocked error occurs for >= x times in 5 minutes, 1 consecutive time. Input 0 to disable this alarm.
UnreachableNodeNumber	3	Nodes minimum is < x for 1 day, 1 consecutive time. 0 means that the alarm is disabled.

Parameter	Default	Description
AutomatedSnapshotFailure	Yes	Whether to enable alarm when an automated snapshot failed. AutomatedSnapshotFailure maximum is >= 1 for 1 minute, 1 consecutive time.
CPUUtilization	Yes	Whether to enable alarm when sustained high usage of CPU occurred. CPUUtilization or WarmCPUUtilization maximum is >= 80% for 15 minutes, 3 consecutive times.
JVMMemoryPressure	Yes	Whether to enable alarm when JVM RAM usage peak occurred. JVMMemoryPressure or WarmJVMMemoryPressure maximum is >= 80% for 5 minutes, 3 consecutive times.
MasterCPUUtilization	Yes	Whether to enable alarm when sustained high usage of CPU occurred in master nodes. MasterCPUUtilization maximum is >= 50% for 15 minutes, 3 consecutive times.
MasterJVMMemoryPressure	Yes	Whether to enable alarm when JVM RAM usage peak occurred in master nodes. MasterJVMMemoryPressure maximum is >= 80% for 15 minutes, 1 consecutive time.

Parameter	Default	Description
KMSKeyError	Yes	Whether to enable alarm when the AWS KMS encryption key is disabled. KMSKeyError is >= 1 for 1 minute, 1 consecutive time.
KMSKeyInaccessible	Yes	Whether to enable alarm when the AWS KMS encryption key has been deleted or has revoked its grants to OpenSearch Service. KMSKeyInaccessible is >= 1 for 1 minute, 1 consecutive time.

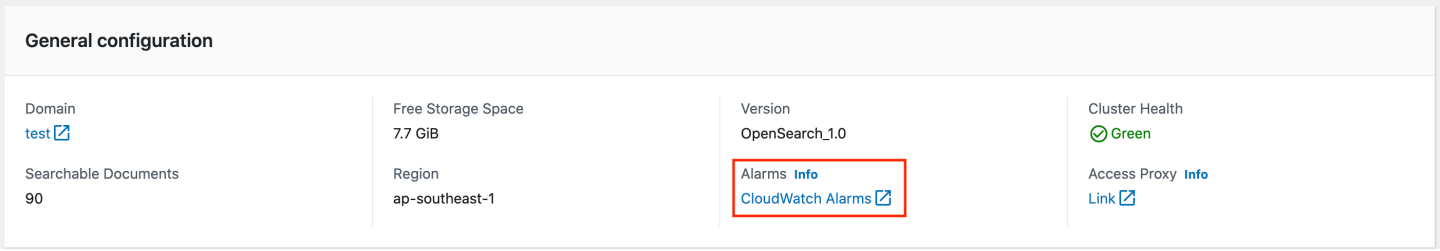
1. Choose **Next**.
2. On the **Configure stack options** page, choose **Next**.
3. On the **Review and create** page, review and confirm the settings. Check the box acknowledging that the template creates AWS Identity and Access Management (IAM) resources.
4. Choose **Submit** to deploy the stack.

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 5 minutes.

Once you have created the alarms, a confirmation email will be sent to your email address. You must choose the **Confirm** link in the email.

Go to the CloudWatch Alarms page by choosing the **General configuration > Alarms > CloudWatch Alarms** link on the Centralized Logging with OpenSearch console, and the link location is shown as follows:

General configuration screen.



Make sure that all the alarms are in **OK** status because you might have missed the notification if the alarms have changed its status before subscription.

> **Note**
>
> The alarm will not send an Amazon SNS notification to your email address if triggered before subscription. We recommend you check the alarms status after enabling the OpenSearch alarms. If you see any alarm, which is in **In Alarm** status, you should fix that issue first.

Delete alarms

1. Log in to the Centralized Logging with OpenSearch console.
2. In the navigation pane, under **Domains**, choose **OpenSearch domains**.
3. Select the domain from the table.
4. Choose the **Alarms** tab.
5. Choose the **Delete**.
6. On the confirmation prompt, choose **Delete**.

AWS service logs

Centralized Logging with OpenSearch supports ingesting AWS service logs into Amazon OpenSearch Service through log analytics pipelines, which you can build using the **Centralized Logging with OpenSearch web console** or via a **standalone CloudFormation template**.

Centralized Logging with OpenSearch reads the data source, parse, cleanup/enrich, and ingest logs into Amazon OpenSearch Service domains for analysis. Moreover, the solution provides templated dashboards to facilitate log visualization.

Amazon OpenSearch Service is suitable for real-time log analytics and frequent queries and has full-text search capability.

As of release 2.1.0, the solution starts to support log ingestion into Light Engine, which is suitable for non-real-time log analytics and infrequent queries and has SQL-like search capability. You will see an option to choose the desired log analytics engine when creating the log analytics pipeline

> **Important**
>
> Supported AWS services must be in the same Region as Centralized Logging with OpenSearch. To ingest logs from different AWS Regions, we recommend using S3 Cross-Region Replication.

Supported AWS services

Most of the supported AWS services output logs to Amazon CloudWatch Logs, Amazon S3, Amazon Kinesis Data Streams, or Amazon Kinesis DataFirehose. The log outputs must be in the same AWS Region as the Centralized Logging with OpenSearch solution.

The following table lists the supported AWS services and the supported log analytics engines.

AWS Service	Log Type	OpenSearch Engine Support	Light Engine Support
AWS CloudTrail	N/A	Yes	Yes

AWS Service	Log Type	OpenSearch Engine Support	Light Engine Support
Amazon S3	Access logs	Yes	No
Amazon RDS/Aurora	MySQL Logs	Yes	Yes
Amazon CloudFront	Standard access logs	Yes	Yes
Application Load Balancer	Access logs	Yes	Yes
AWS WAF	Web ACL logs	Yes	Yes
AWS Lambda	N/A	Yes	No
Amazon VPC	Flow logs	Yes	Yes
AWS Config	N/A	Yes	No

The solution supports detects the log location of the resource automatically, reads the logs, and then ingests them into the log analytics engines. The solution also provides dashboard templates for all supported AWS service. It automatically ingests logs into the log analytics engine. You can go to the OpenSearch Dashboards or Grafana to view the dashboards after the pipeline being provisioned.

In this chapter, you will learn how to create log ingestion and dashboards for the following AWS services:

- AWS CloudTrail
- Amazon S3
- Amazon RDS/Aurora
- Amazon CloudFront
- AWS Lambda
- Application Load Balancer
- AWS WAF
- Amazon VPC

- AWS Config

Cross-Region log ingestion

When you deploy Centralized Logging with OpenSearch in one Region, the solution allows you to ingest service logs from another Region.

> **Note**
>
> For Amazon RDS/Aurora and AWS Lambda service logs, this feature is not supported.

The Region where the service resides is referred to as "Source Region", while the Region where the Centralized Logging with OpenSearch console is deployed as "Logging Region".

For AWS CloudTrail, you can create a new trail that sends logs into a S3 bucket in the Logging Region, and you can find the CloudTrail in the list. To learn how to create a new trail, refer to Creating a trail.

For other services with logs located in S3 buckets, you can manually transfer logs (for example, using the S3 Cross-Region Replication feature) to the Logging Region S3 bucket.

You can complete the following steps to implement cross-Region log ingestion:

1. Set the service log location in another Region to be the Logging Region (such as AWS WAF), or automatically copy logs from the Source Region to the Logging Region using Cross-Region Replication (CRR).
2. In the solution console, choose **AWS Service Log** in the left navigation pane, and choose **Create a pipeline**.
3. In the **Select an AWS Service** area, choose a service in the list, and choose **Next**.
4. In **Creation Method**, choose **Manual**, then enter the resource name and Amazon S3 log location parameter, and choose **Next**.
5. Change log analytics engines and log lifecycle settings, and choose **Next**.
6. Add tags if you need, and choose **Next** to create the pipeline.

Then you can use the OpenSearch dashboard or Grafana to discover logs and view dashboards.

AWS CloudTrail logs

AWS CloudTrail monitors and records account activity across your AWS infrastructure. It outputs all the data to the specified S3 bucket or a CloudWatch Log Group.

You can create a log analytics pipeline either by using the Centralized Logging with OpenSearch console or by deploying a standalone CloudFormation stack.

> **Important**
>
> - The CloudTrail logging bucket must be in the same Region as the Centralized Logging with OpenSearch solution.
> - The Amazon OpenSearch Service index is rotated on a daily basis by default, and you can adjust the index in the Additional Settings.

Create log ingestion (OpenSearch Engine)

Using the Centralized Logging with OpenSearch console

1. Sign in to the Centralized Logging with OpenSearch console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose **Create a log ingestion**.
4. In the AWS Services section, choose AWS CloudTrail.
5. Choose **Next**.
6. Under Specify settings, choose Automatic or Manual.
   - For **Automatic** mode, choose a CloudTrail in the dropdown list.
   - For **Manual** mode, enter the CloudTrail name.
   - (Optional) If you are ingesting CloudTrail logs from another account, select a linked account from the **Account** dropdown list first.
7. Under **Log Source**, Select **Amazon S3** or **CloudWatch** as the log source.
8. Choose **Next**.
9. In **the Specify OpenSearch domain** section, select an imported domain for the Amazon OpenSearch Service domain.

10. Choose **Yes** for **Sample dashboard** if you want to ingest an associated built-in Amazon OpenSearch Service dashboard.
11. You can change the **Index Prefix** of the target Amazon OpenSearch Service index if needed. The default prefix is your trail name.
12. In the **Log Lifecycle** section, enter the number of days to manage the Amazon OpenSearch Service index lifecycle. Centralized Logging with OpenSearch will create the associated Index State Management (ISM) policy automatically for this pipeline.
13. In the **Select log processor** section, choose the log processor.
    a. When selecting Lambda as a log processor, you can configure the Lambda concurrency if needed.
    b. (Optional) OSI as log processor is now supported in these Regions. When OSI is selected, type in the minimum and maximum number of OCU. See more information here.
14. Choose **Next**.
15. Add tags if needed.
16. Choose **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - CloudTrail Log Ingestion* solution in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Regions		Template
AWS China Regions		Template

1. Log in to the AWS Management Console and select the preceding button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.

2. To launch the stack 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.

Parameter	Default	Description
Log Bucket Name	*<Requires input>*	The S3 bucket name that stores the logs.
Log Bucket Prefix	*<Requires input>*	The S3 bucket path prefix that stores the logs.
Log Source Account ID	<Optional>	The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at **Step 1** will be used.
Log Source Region	*Optional input*	The AWS Region of the S3 bucket. By default, the Region you selected at **Step 2** will be used.
Log Source Account Assume Role	*Optional input*	The IAM Role ARN used for cross-account log ingestion . Required for cross-account log ingestion (add a member account first).

Parameter	Default	Description
KMS-CMK ARN	*Optional input*	The KMS-CMK ARN for encryption. Leave it blank to create a new AWS KMS key.
Enable OpenSearch Ingestion as processor	*Optional input*	Ingestion table ARN. Leave empty if you do not use OSI as Processor.
Amazon S3 Backup Bucket	*<Requires input>*	The Amazon S3 backup bucket name to store the failed ingestion logs.
Engine Type	`OpenSearch`	The engine type of the OpenSearch.
OpenSearch Domain Name	*<Requires input>*	The domain name of the Amazon OpenSearch Service cluster.
OpenSearch Endpoint	*<Requires input>*	The OpenSearch endpoint URL. For example, `vpc-your_opensearch_domain_name-xcvgw6uu2o6zafsiefxubwuohe.us-east-1.es.amazonaws.com`
Index Prefix	*<Requires input>*	The common prefix of OpenSearch index for the log. The index name will be `<Index Prefix>-<Log Type>-<Other Suffix>` .
Create Sample Dashboard	`Yes`	Whether to create a sample OpenSearch dashboard.

Parameter	Default	Description
VPC ID	*<Requires input>*	Select a VPC that has access to the OpenSearch domain. The log processing Lambda will reside in the selected VPC.
Subnet IDs	*<Requires input>*	Select at least two subnets that have access to the OpenSearch domain. The log processing Lambda will reside in the subnets. Make sure that the subnets have access to the Amazon S3 service.
Security Group ID	*<Requires input>*	Select a Security Group that will be associated with the log processing Lambda. Make sure that the Security Group has access to the OpenSearch domain.
Number Of Shards	5	Number of shards to distribute the index evenly across all data nodes. Keep the size of each shard between 10-50 GB.
Number of Replicas	1	Number of replicas for OpenSearch Index. Each replica is a full copy of an index. If the OpenSearch option is set to Domain with standby, you need to configure it to 2.

Parameter	Default	Description
Age to Warm Storage	*Optional input*	The age required to move the index into warm storage (for example, 7d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when warm storage is enabled in OpenSearch.
Age to Cold Storage	*Optional input*	The age required to move the index into cold storage (for example, 30d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when cold storage is enabled in OpenSearch.
Age to Retain	*Optional input*	The age to retain the index (for example, 180d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). If the value is "", the index will not be deleted.
Rollover Index Size	*Optional input*	The minimum size of the shard storage required to roll over the index (for example, 30GB).

Parameter	Default	Description
Index Suffix	`yyyy-MM-dd`	The common suffix format of OpenSearch index for the log (Example: yyyy-MM-dd, yyyy-MM-dd-HH). The index name will be `<Index Prefix>-<Log Type>-<Index Suffix>-000001` .
Compression type	`best_compression`	The compression type to use to compress stored data. Available values are best_compression and default.
Refresh Interval	`1s`	How often the index should refresh, which publishes its most recent changes and makes them available for searching. Can be set to -1 to disable refreshing. The default is 1s.
EnableS3Notification	`True`	An option to enable or disable notifications for Amazon S3 buckets. The default option is recommended for most cases.
LogProcessorRoleName	*Optional input*	Specify a role name for the log processor. The name should NOT duplicate an existing role name. If no name is specified, a random name is generated.

Parameter	Default	Description
QueueName	*Optional input*	Specify a queue name for an Amazon SQS queue. The name should NOT duplicate an existing queue name. If no name is given, a random name will be generated.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review and create** page, review and confirm the settings. Check the box acknowledging that the template creates IAM resources.
9. Choose **Submit** 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.

View dashboard

The dashboard includes the following visualizations.

Visualization Name	Source Field	Description
Global Control	awsRegion	Provides users with the ability to drill down data by Region.
Event History	log event	Presents a bar chart that displays the distribution of events over time.
Event by Account ID	userIdentity.accountId	Breaks down events based on the AWS account ID, enabling you to analyze activity patterns across different accounts within your organization.

Visualization Name	Source Field	Description
Top Event Names	eventName	Shows the most frequentl y occurring event names, helping you identify common activities or potential anomalies.
Top Event Sources	eventSource	Highlights the top sources generating events, providing insights into the services or resources that are most active or experiencing the highest event volume.
Event Category	eventCategory	Categorizes events into different types or classific ations, facilitating analysis and understanding of event distribution across categories.
Top Users	* userIdentity.sessionContext .sessionIssuer.userName * userIdentity.sessionContext .sessionIssuer.arn * userIdent ity.accountId * userIdent ity.sessionContext.sessionI ssuer.type	Identifies the users or IAM roles associated with the highest number of events, aiding in user activity monitoring and access management.
Top Source IPs	sourceIPAddress	Lists the source IP addresses associated with events, enabling you to identify and investigate potentially suspicious or unauthorized activities.

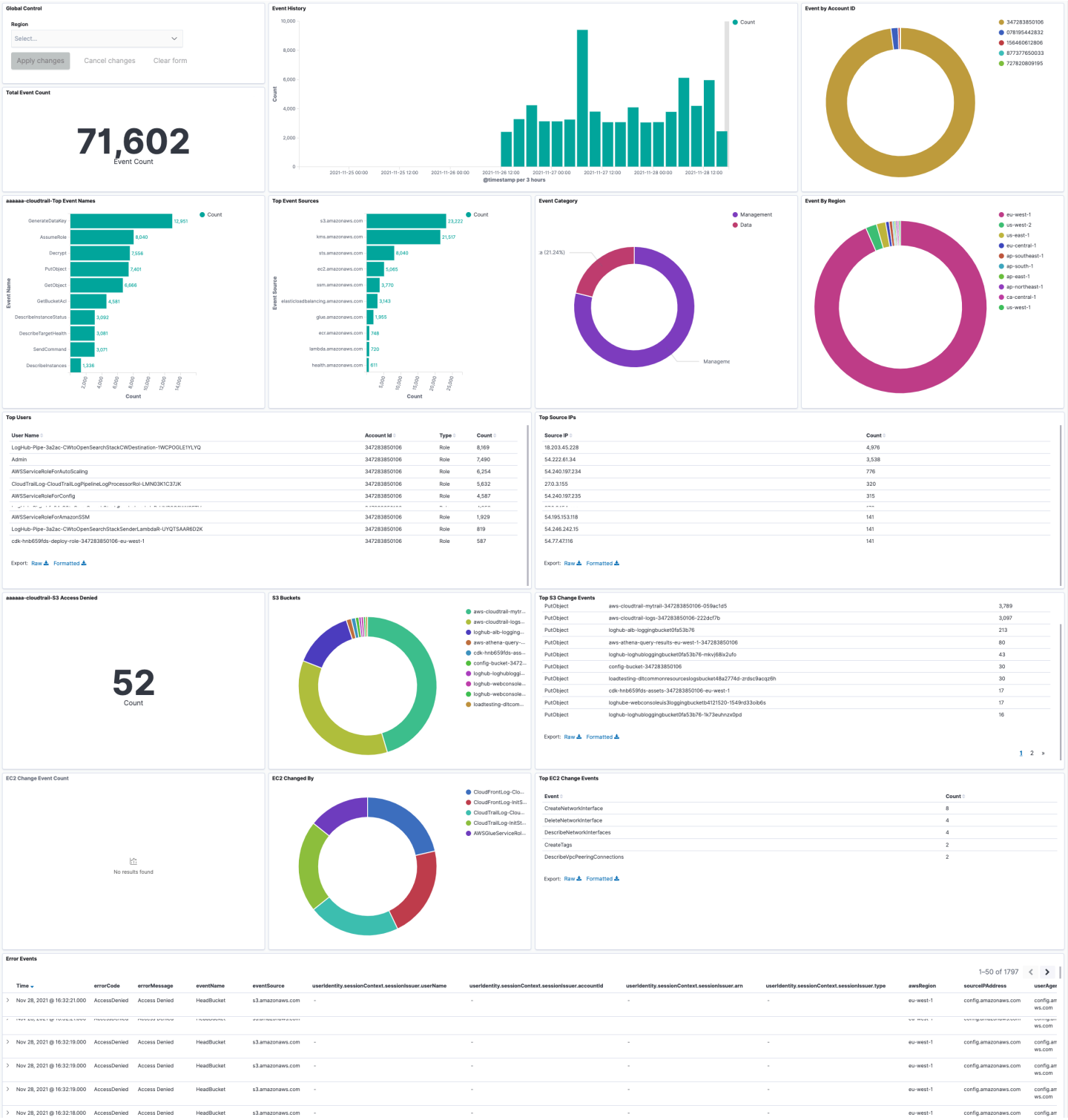
Visualization Name	Source Field	Description
Amazon S3 Access Denied	* eventSource: s3* * errorCode : AccessDenied	Displays events where access to Amazon S3 resources was denied, helping you identify and troubleshoot permissio n issues or potential security breaches.
S3 Buckets	requestParameters. bucketName	Provides a summary of S3 bucket activity, including create, delete, and modify operations, allowing you to monitor changes and access patterns.
Top Amazon S3 Change Events	* eventName * requestPa rameters.bucketName	Presents the most common types of changes made to Amazon S3 resources, such as object uploads, deletions , or modifications, aiding in change tracking and auditing.
EC2 Change Event Count	* eventSource: ec2* * eventName: (RunInstances or TerminateInstances or RunInstances or StopInsta nces)	Shows the total count of EC2-related change events, giving an overview of the volume and frequency of changes made to EC2 instances and resources.
EC2 Changed By	userIdentity.sessionContext .sessionIssuer.userName	Identifies the users or IAM roles responsible for changes to EC2 resources, assisting in accountability and tracking of modifications.

Visualization Name	Source Field	Description
Top EC2 Change Events	eventName	Highlights the most common types of changes made to EC2 instances or related resources , allowing you to focus on the most significant or frequent changes.
Error Events	* awsRegion * errorCode * errorMessage * eventName * eventSource * sourceIPAddress * userAgent * userIdentity.accountId * userIdentity.sessionContext.sessionIssuer.accountId * userIdentity.sessionContext.sessionIssuer.arn * userIdentity.sessionContext.sessionIssuer.type * userIdentity.sessionContext.sessionIssuer.userName	Displays events that resulted in errors or failures, helping you identify and troubleshoot issues related to API calls or resource operations.

You can access the built-in dashboard in Amazon OpenSearch Service to view log data. For more information, see the Access Dashboard.

You can choose the following image to view the high-resolution sample dashboard.

CloudTrail logs sample dashboard.


Global Control
Region
Select...
Apply changes
Cancel changes
Clear form
Total Event Count
71,602
Event Count
Event History
Count
@timestamp per 3 hours
Event by Account ID
347283850106
078195442832
156460612806
877377650033
727820809195
aaaaaa-cloudtrail-Top Event Names
Event Name
Count
GenerateDataKey 12,951
AssumeRole 8,040
Decrypt 7,556
PutObject 7,401
GetObject 6,666
GetBucketAcl 4,581
DescribeInstanceStatus 3,092
DescribeTargetHealth 3,081
SendCommand 3,071
DescribeInstances 1,336
Top Event Sources
Event Source
Count
s3.amazonaws.com 23,222
kms.amazonaws.com 21,517
sts.amazonaws.com 8,040
ec2.amazonaws.com 5,065
ssm.amazonaws.com 3,770
elasticloadbalancing.amazonaws.com 3,143
glue.amazonaws.com 1,955
ecr.amazonaws.com 748
lambda.amazonaws.com 720
health.amazonaws.com 611
Event Category
Management
Data
Event By Region
eu-west-1
us-west-2
us-east-1
eu-central-1
ap-southeast-1
ap-south-1
ap-east-1
ap-northeast-1
ca-central-1
us-west-1
Top Users
User Name
Account Id
Type
Count
LogHub-Pipe-3a2ac-CWtoOpenSearchStackCWDestination-1WCPOGLE1YLYQ 347283850106 Role 8,169
Admin 347283850106 Role 7,490
AWSServiceRoleForAutoScaling 347283850106 Role 6,254
CloudTrailLog-CloudTrailLogPipelineLogProcessorRol-LMN03K1C37JK 347283850106 Role 5,632
AWSServiceRoleForConfig 347283850106 Role 4,587
AWSServiceRoleForAmazonSSM 347283850106 Role 1,929
LogHub-Pipe-3a2ac-CWtoOpenSearchStackSenderLambdaR-UYQTSAAR6D2K 347283850106 Role 819
cdk-hnb659fds-deploy-role-347283850106-eu-west-1 347283850106 Role 587
Export: Raw Formatted
Top Source IPs
Source IP
Count
18.203.45.228 4,976
54.222.61.34 3,538
54.240.197.234 776
270.3.155 320
54.240.197.235 315
54.195.153.118 141
54.246.242.15 141
54.77.47.116 141
Export: Raw Formatted
aaaaaa-cloudtrail-S3 Access Denied
52
Count
S3 Buckets
Top S3 Change Events
PutObject aws-cloudtrail-mytrail-347283850106-059ac1d5 3,789
PutObject aws-cloudtrail-logs-347283850106-222dcf7b 3,097
PutObject loghub-alb-loggingbucket0fa53b76 213
PutObject aws-athena-query-results-eu-west-1-347283850106 80
PutObject loghub-loghubloggingbucket0fa53b76-mkvj68ix2ufo 43
PutObject config-bucket-347283850106 30
PutObject loadtesting-dltcommonresourceslogsbucket48a2774d-zrdsc9acqz6h 30
PutObject cdk-hnb659fds-assets-347283850106-eu-west-1 17
PutObject loghube-webconsoleuis3loggingbucketb41211520-1549rd33oib6s 17
PutObject loghub-loghubloggingbucket0fa53b76-1k73euhnzx0pd 16
Export: Raw Formatted
1 2 »
EC2 Change Event Count
No results found
EC2 Changed By
Top EC2 Change Events
Event
Count
CreateNetworkInterface 8
DeleteNetworkInterface 4
DescribeNetworkInterfaces 4
CreateTags 2
DescribeVpcPeeringConnections 2
Export: Raw Formatted
Error Events
1–50 of 1797
Time
errorCode
errorMessage
eventName
eventSource
userIdentity.sessionContext.sessionIssuer.userName
userIdentity.sessionContext.sessionIssuer.accountId
userIdentity.sessionContext.sessionIssuer.arn
userIdentity.sessionContext.sessionIssuer.type
awsRegion
sourceIPAddress
userAgent
Nov 28, 2021 @ 16:32:21.000 AccessDenied Access Denied HeadBucket s3.amazonaws.com - - - - eu-west-1 config.amazonaws.com config.amazonaws.com
Nov 28, 2021 @ 16:32:19.000 AccessDenied Access Denied HeadBucket s3.amazonaws.com - - - - eu-west-1 config.amazonaws.com config.amazonaws.com
Nov 28, 2021 @ 16:32:19.000 AccessDenied Access Denied HeadBucket s3.amazonaws.com - - - - eu-west-1 config.amazonaws.com config.amazonaws.com
Nov 28, 2021 @ 16:32:19.000 AccessDenied Access Denied HeadBucket s3.amazonaws.com - - - - eu-west-1 config.amazonaws.com config.amazonaws.com
Nov 28, 2021 @ 16:32:18.000 AccessDenied Access Denied HeadBucket s3.amazonaws.com - - - - eu-west-1 config.amazonaws.com config.amazonaws.com

Create log ingestion (Light Engine)

Using the Centralized Logging with OpenSearch console

1. Sign in to the Centralized Logging with OpenSearch console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose **Create a log ingestion**.
4. In the **AWS Services** section, choose AWS CloudTrail.
5. Choose **Next**.
6. Under Specify settings, choose Automatic or Manual.
   - For **Automatic** mode, choose a CloudTrail in the dropdown list.
   - For **Manual** mode, enter the CloudTrail name.
   - (Optional) If you are ingesting CloudTrail logs from another account, select a linked account from the **Account** dropdown list first.
7. Choose **Next**.
8. In the **Specify Light Engine Configuration** section, if you want to ingest associated templated Grafana dashboards, select **Yes** for the sample dashboard.
9. You can choose an existing Grafana, or if you must import a new one, you can go to Grafana for configuration.
10. Select an S3 bucket to store partitioned logs and define a name for the log table. We have provided a predefined table name, but you can modify it according to your business needs.
11. If needed, change the log processing frequency, which is set to **5** minutes by default, with a minimum processing frequency of **1** minute.
12. In the **Log Lifecycle** section, enter the log merge time and log archive time. We have provided default values, but you can adjust them based on your business requirements.
13. Select **Next**.
14. If desired, add tags.
15. Select **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - CloudTrail Log Ingestion* solution in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Regions		Template
AWS China Regions		Template

1. Log in to the AWS Management Console and select the preceding button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.
2. To launch the stack 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.

   a. Parameters for **Pipeline settings**

Parameter	Default	Description
Pipeline Id	`<Requires input>`	The unique identifier for the pipeline is essential if you must create multiple Application Load Balancer pipelines and write different Application Load Balancer logs into separate tables. To ensure uniqueness, you can generate a unique pipeline

Parameter	Default	Description
		identifier using uuidgenerator.
Staging Bucket Prefix	`AWSLogs/CloudTrail Logs`	The storage directory for logs in the temporary storage area should ensure the uniqueness and non-overlapping of the Prefix for different pipelines.

b. Parameters for **Destination settings**

Parameter	Default	Description
Centralized Bucket Name	*`<Requires input>`*	Centralized S3 bucket name. For example, centralized-logging-bucket.
Centralized Bucket Prefix	`datalake`	Centralized bucket prefix. By default, the data base location is s3://{Centralized Bucket Name}/{Centralized Bucket Prefix}/amazon_cl_centralized.
Centralized Table Name	`CloudTrail`	Table name for writing data to the centralized database. You can modify it if needed.

c. Parameters for **Scheduler settings**

Parameter	Default	Description
LogProcessor Schedule Expression	`rate(5 minutes)`	Task scheduling expression for performing log processing, with a default value of executing the LogProcessor every 5 minutes. Configuration for reference.
LogMerger Schedule Expression	`cron(0 1 * ? )`	Task scheduling expression for performing log merging, with a default value of executing the LogMerger at 1 AM every day. Configuration for reference.
LogArchive Schedule Expression	`cron(0 2 * ? )`	Task scheduling expression for performing log archiving, with a default value of executing the LogArchive at 2 AM every day. Configuration for reference.
Age to Merge	7	Small file retention days, with a default value of 7, indicating that logs older than 7 days will be merged into small files. It can be adjusted as needed.

Parameter	Default	Description
Age to Archive	30	Log retention days, with a default value of 30, indicating that data older than 30 days will be archived and deleted. It can be adjusted as needed.

d. Parameters for **Notification settings**

Parameter	Default	Description
Notification Service	SNS	Notification method for alerts. If your main stack is using China, you can only choose the SNS method. If your main stack is using Global, you can choose either the SNS or SES method.

Parameter	Default	Description
Recipients	*<Requires input>*	Alert notification: If the Notification Service is SNS, enter the SNS Topic ARN here, ensuring that you have the necessary permissions. If the Notification Service is SES, enter the email addresses separated by commas here, ensuring that the email addresses are already Verified Identitie s in SES. The adminEmai l provided during the creation of the main stack will receive a verification email by default.

e. Parameters for **Dashboard settings**

Parameter	Default	Description
Import Dashboards	FALSE	Whether to import the Dashboard into Grafana, with a default value of false. If set to true, you must provide the Grafana URL and Grafana Service Account Token.
Grafana URL	*<Requires input>*	Grafana access URL for example: https:// alb-72277319.us-wes t-2.elb.amazonaws.com.

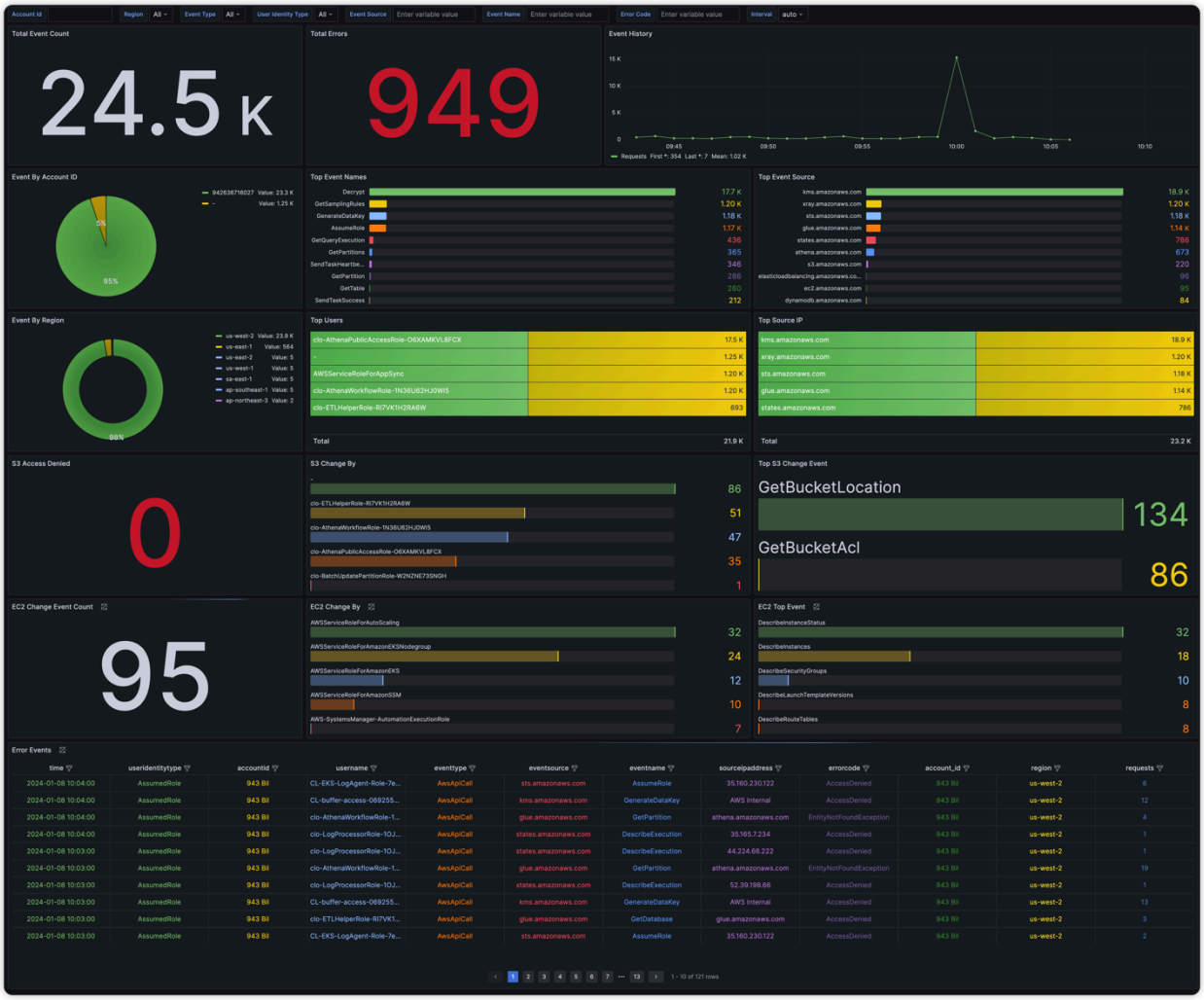
Parameter	Default	Description
Grafana Service Account Token	*<Requires input>*	Grafana Service Account Token: Service Account Token created in Grafana.

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

View dashboard

CloudTrail log sample dashboard



Amazon S3 logs

Amazon S3 server access logging provides detailed records for the requests made to the bucket. S3 Access Logs can be enabled and saved in another S3 bucket.

You can create a log ingestion into Amazon OpenSearch Service either by using the Centralized Logging with OpenSearch console or by deploying a standalone CloudFormation stack.

Important

- The S3 Bucket Region must be the same as the Centralized Logging with OpenSearch solution Region.

- The Amazon OpenSearch Service index is rotated on a daily basis by default, and you can adjust the index in the Additional Settings.

Create log ingestion (OpenSearch Engine)

Using the Centralized Logging with OpenSearch Console

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose the Create a log ingestion button.
4. In the **AWS Services** section, choose **Amazon S3**.
5. Choose **Next**.
6. Under **Specify settings**, choose **Automatic** or **Manual** for **Amazon S3 Access Log enabling**. The automatic mode will enable the Amazon S3 Access Log and save the logs to a centralized S3 bucket if logging is not enabled yet.
   - For **Automatic mode**, choose the S3 bucket from the dropdown list.
   - For Manual mode, enter the Bucket Name and Amazon S3 Access Log location.
   - (Optional) If you are ingesting Amazon S3 logs from another account, select a linked account from the **Account** dropdown list first.
7. Choose **Next**.
8. In the Specify OpenSearch domain section, select an imported domain for the Amazon OpenSearch Service domain.
9. Choose **Yes** for **Sample dashboard** if you want to ingest an associated built-in Amazon OpenSearch Service dashboard.
10. You can change the **Index Prefix** of the target Amazon OpenSearch Service index if needed. The default prefix is your bucket name.
11. In the **Log Lifecycle** section, enter the number of days to manage the Amazon OpenSearch Service index lifecycle. The Centralized Logging with OpenSearch will create the associated Index State Management (ISM) policy automatically for this pipeline.
12. Choose **Next**.
13. Add tags if needed.
14. Choose **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - Amazon S3 Access Log Ingestion* solution in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Regions		Template
AWS China Regions		Template

1. Log in to the AWS Management Console and select the preceding button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.
2. To launch the stack 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.

Parameter	Default	Description
Log Bucket Name	`<Requires input>`	The S3 bucket name which stores the logs.
Log Bucket Prefix	`<Requires input>`	The S3 bucket path prefix which stores the logs.
Log Source Account ID	*Optional input*	The AWS Account ID of the S3 bucket. Required for

Parameter	Default	Description
		cross-account log ingestion (add a member account first). By default, the Account ID you logged in at **Step 1** will be used.
Log Source Region	*Optional input*	The AWS Region of the S3 bucket. By default, the Region you selected at **Step 2** will be used.
Log Source Account Assume Role	*Optional input*	The IAM Role ARN used for cross-account log ingestion . Required for cross-account log ingestion (add a member account first).
KMS-CMK ARN	*Optional input*	The KMS-CMK ARN for encryption. Leave it blank to create a new AWS KMS key.
Enable OpenSearch Ingestion as processor	*Optional input*	Ingestion table ARN. Leave empty if you do not use OSI as Processor.
Amazon S3 Backup Bucket	*<Requires input>*	The Amazon S3 backup bucket name to store the failed ingestion logs.
Engine Type	OpenSearch	The engine type of the OpenSearch.
OpenSearch Domain Name	*<Requires input>*	The domain name of the Amazon OpenSearch Service cluster.

Parameter	Default	Description
OpenSearch Endpoint	*<Requires input>*	The OpenSearch endpoint URL. For example, vpc-your_opensearch_domain_name-xcvgw6uu2o6zafsiefxubwuohe.us-east-1.es.amazonaws.com
Index Prefix	*<Requires input>*	The common prefix of OpenSearch index for the log. The index name will be <Index Prefix>-<Log Type>-<Other Suffix>.
Create Sample Dashboard	Yes	Whether to create a sample OpenSearch dashboard.
VPC ID	*<Requires input>*	Select a VPC that has access to the OpenSearch domain. The log processing Lambda will reside in the selected VPC.
Subnet IDs	*<Requires input>*	Select at least two subnets that have access to the OpenSearch domain. The log processing Lambda will reside in the subnets. Make sure that the subnets have access to the Amazon S3 service.

Parameter	Default	Description
Security Group ID	*<Requires input>*	Select a Security Group that will be associated with the log processing Lambda. Make sure that the Security Group has access to the OpenSearch domain.
Number Of Shards	5	Number of shards to distribute the index evenly across all data nodes. Keep the size of each shard between 10-50 GB.
Number of Replicas	1	Number of replicas for OpenSearch Index. Each replica is a full copy of an index. If the OpenSearch option is set to Domain with standby, you need to configure it to 2.
Age to Warm Storage	*Optional input*	The age required to move the index into warm storage (for example, 7d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when warm storage is enabled in OpenSearch.

Parameter	Default	Description
Age to Cold Storage	*Optional input*	The age required to move the index into cold storage (for example, 30d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when cold storage is enabled in OpenSearch.
Age to Retain	*Optional input*	The age to retain the index (for example, 180d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). If the value is "", the index will not be deleted.
Rollover Index Size	*Optional input*	The minimum size of the shard storage required to roll over the index (for example, 30GB).
Index Suffix	`yyyy-MM-dd`	The common suffix format of OpenSearch index for the log(Example: yyyy-MM-dd, yyyy-MM-dd-HH). The index name will be <Index Prefix>-<Log Type>-<Index Suffix>-000001.

Parameter	Default	Description
Compression type	`best_compression`	The compression type to use to compress stored data. Available values are best_compression and default.
Refresh Interval	`1s`	How often the index should refresh, which publishes its most recent changes and makes them available for searching. Can be set to -1 to disable refreshing. Default is 1s.
EnableS3Notification	`True`	An option to enable or disable notifications for Amazon S3 buckets. The default option is recommend ed for most cases.
LogProcessorRoleName	*Optional input*	Specify a role name for the log processor. The name should NOT duplicate an existing role name. If no name is specified, a random name is generated.
QueueName	*Optional input*	Specify a queue name for an Amazon SQS queue. The name should NOT duplicate an existing queue name. If no name is given, a random name will be generated.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.

8. On the **Review and create** page, review and confirm the settings. Check the box acknowledging that the template creates AWS Identity and Access Management (IAM) resources.
9. Choose **Submit** 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.

View dashboard

The dashboard includes the following visualizations.

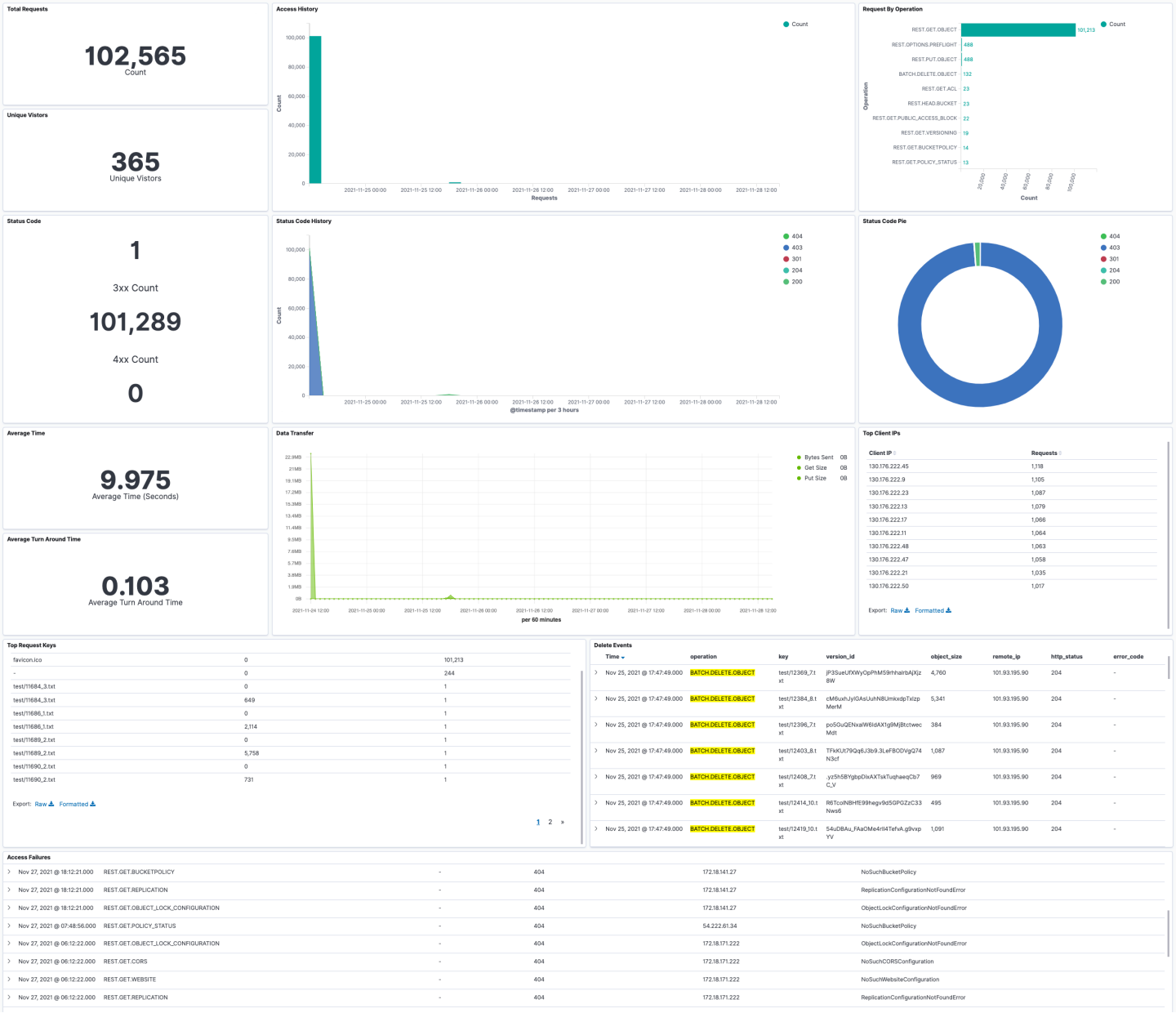
Visualization Name	Source Field	Description
Total Requests	* log event	A visualization showing the total number of requests made to the Amazon S3 bucket, including all types of operations (for example, GET, PUT, DELETE).
Unique Visitors	* log event	This visualization displays the count of unique visitors accessing the Amazon S3 bucket, identified by their IP addresses.
Access History	* log event	Provides a chronological log of all access events made to the Amazon S3 bucket, including details about the operations and their outcomes.
Request By Operation	* operation	This visualization categorizes and shows the distribution of requests based on different

Visualization Name	Source Field	Description
		operations (for example, GET, PUT, DELETE).
Status Code	* http_status	Displays the count of requests made to the Amazon S3 bucket, grouped by HTTP status codes returned by the server (for example, 200, 404, 403).
Status Code History	* http_status	Shows the historical trend of HTTP status codes returned by the Amazon S3 server over a specific period of time.
Status Code Pie	* http_status	Represents the distribution of requests based on different HTTP status codes using a pie chart.
Average Time	* total_time	This visualization calculate s and presents the average time taken for various operations in the Amazon S3 bucket (for example, average time for GET, PUT requests).
Average Turn Around Time	* turn_around_time	Shows the average turnaroun d time for different operation s, which is the time between receiving a request and sending the response back to the client.

Visualization Name	Source Field	Description
Data Transfer	* bytes_sent * object_size * operation	Provides insights into data transfer activities, including the total bytes transferred, object sizes, and different operations involved.
Top Client IPs	* remote_ip	Displays the top client IP addresses with the highest number of requests made to the Amazon S3 bucket.
Top Request Keys	* key * object_size	Shows the top requested keys in the Amazon S3 bucket along with the corresponding object sizes.
Delete Events	* operation * key * version_id * object_size * remote_ip * http_status * error_code	Focuses on delete events, including the operation, key, version ID, object size, client IP, HTTP status, and error code associated with the delete requests.
Access Failures	* operation * key * version_id * object_size * remote_ip * http_status * error_code	Highlights access failures, showing the details of the failed requests, including operation, key, version ID, object size, client IP, HTTP status, and error code.

You can access the built-in dashboard in Amazon OpenSearch Service to view log data. For more information, see the Access Dashboard.

Amazon S3 logs sample dashboard.



Amazon RDS/Aurora logs

You can publish database instance logs to Amazon CloudWatch Logs. Then, you can perform real-time analysis of the log data, store the data in highly durable storage, and manage the data with the CloudWatch Logs Agent.

Prerequisites

Make sure your database logs are enabled. Some databases logs are not enabled by default, and you must update your database parameters to enable the logs.

Refer to How do I enable and monitor logs for an Amazon RDS MySQL DB instance? to learn how to output logs to CloudWatch Logs.

The following table lists the requirements for Amazon RDS/Aurora MySQL parameters.

Parameter	Requirement
Audit Log	The database instance must use a custom option group with the MARIADB_AUDIT_PLUGIN option.
General log	The database instance must use a custom parameter group with the parameter setting general_log = 1 to enable the general log.
Slow query log	The database instance must use a custom parameter group with the parameter setting slow_query_log = 1 to enable the slow query log.
Log output	The database instance must use a custom parameter group with the parameter setting log_output = FILE to write logs to the file system and publish them to CloudWatch Logs.

You can create a log ingestion into Amazon OpenSearch Service either by using the Centralized Logging with OpenSearch console or by deploying a standalone CloudFormation stack.

Important

The Amazon RDS and CloudWatch Region must be the same as the Centralized Logging with OpenSearch solution Region.
The Amazon OpenSearch Service index is rotated on a daily basis by default, and you can adjust the index in the Additional Settings.

Create log ingestion (OpenSearch Engine)

Using the Centralized Logging with OpenSearch Console

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose the Create a log ingestion button.
4. In the **AWS Services** section, choose **Amazon RDS**.
5. Choose **Next**.
6. Under **Specify settings**, choose **Automatic** or **Manual** for **RDS log enabling**. The automatic mode will detect your Amazon RDS log configurations and ingest logs from CloudWatch.
   - For **Automatic mode**, choose the Amazon RDS cluster from the dropdown list.
   - For **Manual mode**, enter the **DB identifier**, select the **Database type** and input the CloudWatch log location in **Log type and location**.
   - (Optional) If you are ingesting Amazon RDS/Aurora logs from another account, select a linked account from the **Account** dropdown first.
7. Choose **Next**.
8. In the Specify OpenSearch domain section, select an imported domain for the Amazon OpenSearch Service domain.
9. Choose **Yes** for **Sample dashboard** if you want to ingest an associated templated Amazon OpenSearch Service dashboard.
10. You can change the **Index Prefix** of the target Amazon OpenSearch Service index if needed. The default prefix is the Database identifier.
11. In the **Log Lifecycle** section, input the number of days to manage the Amazon OpenSearch Service index lifecycle. The Centralized Logging with OpenSearch will create the associated Index State Management (ISM) policy automatically for this pipeline.
12. In the **Log processor settings** section, choose **Log processor type**, configure the Lambda concurrency if needed, and then choose **Next**.
13. Add tags if needed.
14. Choose **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - RDS Log Ingestion* solution in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Regions		Template
AWS China Regions		Template

1. Log in to the AWS Management Console and select the button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.
2. To launch the Centralized Logging with OpenSearch 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.

Parameter	Default	Description
Log Bucket Name	`<Requires input>`	The S3 bucket name to export the logs.
Log Bucket Prefix	`<Requires input>`	The S3 bucket path prefix that stores the logs.
Log Source Account ID	*Optional input*	The AWS Account ID of the CloudWatch log group.

Parameter	Default	Description
		Required for cross-account log ingestion (Please add a member account first). By default, the Account ID you logged in at **Step 1** will be used.
Log Source Region	*Optional input*	The AWS Region of the CloudWatch log group. By default, the Region you selected at **Step 2** will be used.
Log Source Account Assume Role	*Optional input*	The IAM Role ARN used for cross-account log ingestion . Required for cross-account log ingestion (add a member account first).
KMS-CMK ARN	*Optional input*	The KMS-CMK ARN for Amazon SQS encryption. Leave it blank to create a new AWS KMS key.
Enable OpenSearch Ingestion as processor	*Optional input*	Ingestion table ARN. Leave empty if you do not use OSI as Processor.
Log Group Names	*<Requires input>*	The names of the CloudWatch log group for the logs.
Amazon S3 Backup Bucket	*<Requires input>*	The Amazon S3 backup bucket name to store the failed ingestion logs.
Engine Type	OpenSearch	The engine type of the OpenSearch.

Parameter	Default	Description
OpenSearch Domain Name	*<Requires input>*	The domain name of the Amazon OpenSearch Service cluster.
OpenSearch Endpoint	*<Requires input>*	The OpenSearch endpoint URL. For example, vpc-your_opensearch_domain_name-xcvgw6uu2o6zafsiefxubwuohe.us-east-1.es.amazonaws.com
Index Prefix	*<Requires input>*	The common prefix of OpenSearch index for the log. The index name will be <Index Prefix>-<Log Type>-<Other Suffix>.
Create Sample Dashboard	Yes	Whether to create a sample OpenSearch dashboard.
VPC ID	*<Requires input>*	Select a VPC that has access to the OpenSearch domain. The log processing Lambda will be resides in the selected VPC.
Subnet IDs	*<Requires input>*	Select at least two subnets that has access to the OpenSearch domain. The log processing Lambda function will reside in the subnets. Please make sure that the subnetshas access to the Amazon S3 service.

Parameter	Default	Description
Security Group ID	*<Requires input>*	Select a Security Group that will be associated to the log processing Lambda. Please make sure that the Security Group has access to the OpenSearch domain.
Number Of Shards	5	Number of shards to distribute the index evenly across all data nodes. Keep the size of each shard between 10-50 GiB.
Number of Replicas	1	Number of replicas for OpenSearch Index. Each replica is a full copy of an index.
Age to Warm Storage	*Optional input*	The age required to move the index into warm storage (for example, 7d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when warm storage is enabled in OpenSearch.

Parameter	Default	Description
Age to Cold Storage	*Optional input*	The age required to move the index into cold storage (for example, 30d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when cold storage is enabled in OpenSearch.
Age to Retain	*Optional input*	The age to retain the index (for example, 180d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). If the value is "", the index will not be deleted.
Rollover Index Size	*Optional input*	The minimum size of the shard storage required to roll over the index (for example, 30GB).
Index Suffix	`yyyy-MM-dd`	The common suffix format of OpenSearch index for the log(Example: yyyy-MM-dd, yyyy-MM-dd-HH). The index name will be <Index Prefix>-<Log Type>-<Index Suffix>-000001.

Parameter	Default	Description
Compression type	`best_compression`	The compression type to use to compress stored data. Available values are best_compression and default.
Refresh Interval	`1s`	How often the index should refresh, which publishes its most recent changes and makes them available for searching. Can be set to -1 to disable refreshing. Default is 1s.
EnableS3Notification	`True`	An option to enable or disable notifications for Amazon S3 buckets. The default option is recommend ed for most cases.
LogProcessorRoleName	*Optional input*	Specify a role name for the log processor. The name should NOT duplicate an existing role name. If no name is specified, a random name is generated.
QueueName	*Optional input*	Specify a queue name for an Amazon SQS queue. The name should NOT duplicate an existing queue name. If no name is given, a random name will be generated.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.

8. On the **Review and create** page, review and confirm the settings. Check the box acknowledging that the template creates AWS Identity and Access Management (IAM) resources.
9. Choose **Submit** 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 15 minutes.

View dashboard

The dashboard includes the following visualizations.

Visualization Name	Source Field	Description
Controller	* db-identifier * sq-table-name	This visualization allows users to filter data based on the db-identifier and sq-table-name fields.
Total Log Events Overview	* db-identifier * log event	This visualization presents an overview of the total log events for the specified database ('db-identifier'). It helps monitor the frequency of various log events.
Slow Query History	* log event	This visualization shows the historical data of slow query log events. It allows you to track the occurrences of slow queries and identify potential performance issues.
Average Slow Query Time History	* Average sq-duration	This visualization depicts the historical trend of the average duration of slow queries ('sq-duration'). It helps in understanding the database' s performance over time and

Visualization Name	Source Field	Description
		identifying trends related to slow query durations.
Total Slow Queries	* log event	This visualization provides the total count of slow queries in the log events. It gives an immediate view of how many slow queries have occurred during a specific time period, which is useful for assessing the database's performance and potential bottlenecks.
Average Slow Query Duration	* Average sq-duration	This visualization shows the average duration of slow queries ('sq-duration') over time. It is valuable for understanding the typical performance of slow queries in the database.
Top Slow Query IP	* sq-ip * sq-duration	This visualization highlight s the IP addresses ('sq-ip') associated with the slowest queries and their respective durations ('sq-duration'). It helps identify sources of slow queries and potential areas for optimization.

Visualization Name	Source Field	Description
Slow Query Scatter Plot	* sq-duration * sq-ip * sq-query	This scatter plot visualization represents the relationship between the duration of slow queries ('sq-duration'), the IP addresses ('sq-ip') from which they originated, and the actual query content ('sq-query'). It helps in understanding query performance patterns and identifying potential issues related to specific queries and their sources.
Slow Query Pie	* sq-query	This pie chart visualization shows the distribution of slow queries based on their content ('sq-query'). It provides an overview of the types of queries causing performance issues, allowing you to focus on optimizing specific query patterns.
Slow Query Table Name Pie	* sq-table-name	This pie chart visualization displays the distribution of slow queries based on the table names ('sq-table-name') they access. It helps identify which tables are affected by slow queries, enabling targeted optimization efforts for specific tables.

Visualization Name	Source Field	Description
Top Slow Query	* sq-query	This visualization presents the slowest individual queries based on their content ('sq-query'). It is helpful in pinpointing specific queries that have the most significa nt impact on performance, allowing developers and administrators to focus on optimizing these critical queries.
Slow Query Logs	* db-identifier * sq-db-name * sq-table-name * sq-query * sq-ip * sq-host-name * sq-rows-examined * sq-rows-sent * sq-id * sq-duration * sq-lock-wait	This visualization provides detailed logs of slow queries, including database ('sq-db-name'), table ('sq-table-name'), query content ('sq-query'), IP address ('sq-ip'), hostname ('sq-host-name'), rows examined ('sq-rows-examined'), rows sent ('sq-rows-sent'), query ID ('sq-id'), query duration ('sq-dura tion'), and lock wait time ('sq-lock-wait'). It is beneficia l for in-depth analysis and troubleshooting of slow query performance.

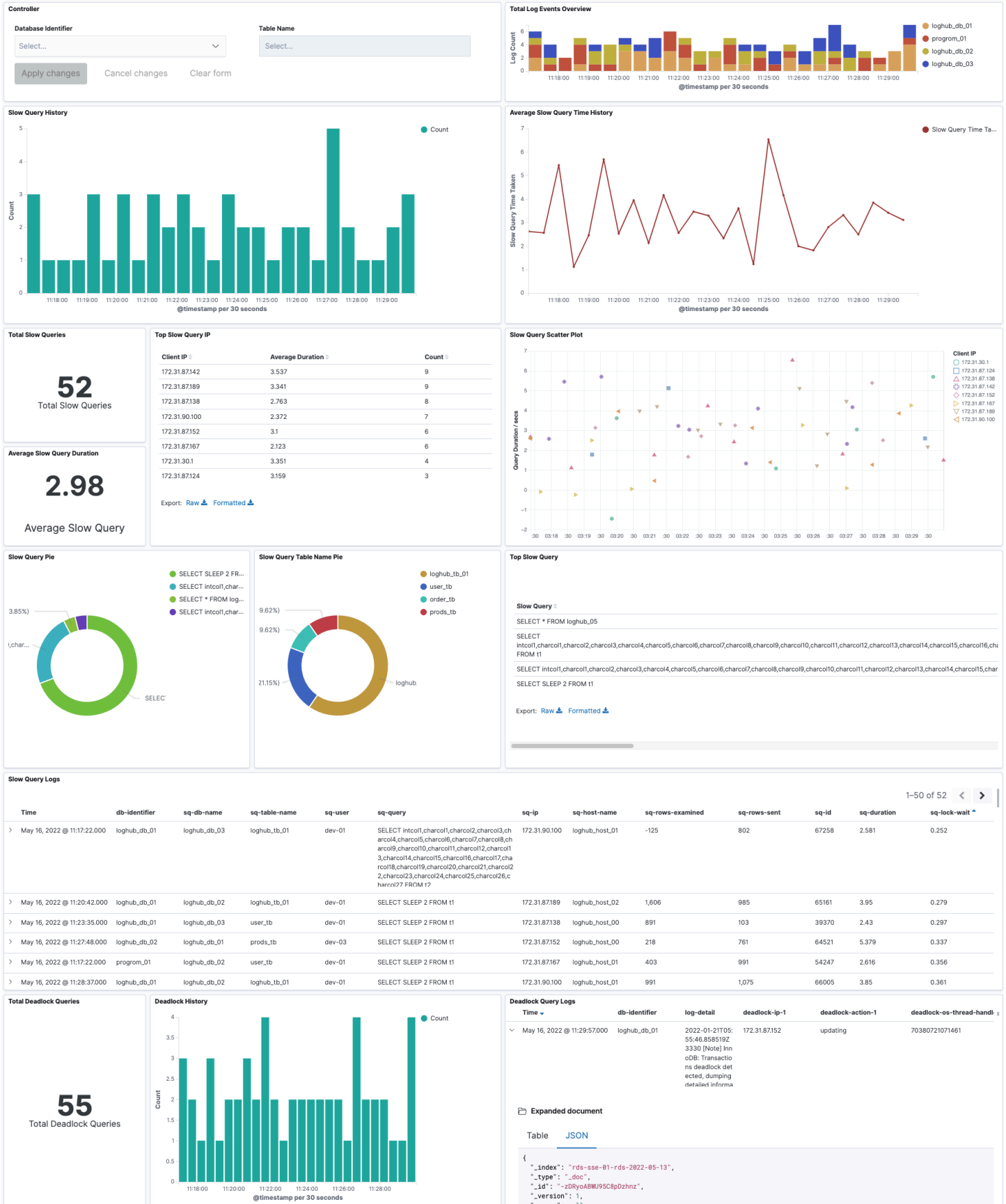
Visualization Name	Source Field	Description
Total Deadlock Queries	* log event	This visualization shows the total number of deadlock occurrences based on the log events. Deadlocks are critical issues that can cause database transactions to fail, and monitoring their frequency is essential for database stability.
Deadlock History	* log event	This visualization displays the historical data of deadlock occurrences based on the log events. Understanding the pattern of deadlocks over time can help identify recurring issues and take preventive measures to reduce their impact on the database.
Deadlock Query Logs	* db-identifier * log-detail * deadlock-ip-1 * deadlock-action-1 * deadlock-os-thread-handle-1 * deadlock-query-1 * deadlock-query-id-1 * deadlock-thread-id-1 * deadlock-user-1 * deadlock-action-2 * deadlock-ip-2 * deadlock-os-thread-handle-2 * deadlock-query-2 * deadlock-query-id-2 * deadlock-thread-id-2 * deadlock-user-2	This visualization provides detailed logs of deadlock occurrences

Visualization Name	Source Field	Description
Total Error Logs	* log event	This visualization presents the total count of error log events. Monitoring error logs helps identify database issues and potential errors that need attention and resolution.
Error History	* log event	This visualization shows the historical data of error log events. Understanding the error patterns over time can aid in identifying recurring issues and taking correctiv e actions to improve the database's overall health and stability.
Error Logs	* db-identifier * err-label * err-code * err-detail * err-sub-system * err-thread	This visualization displays the error logs generated by the Amazon RDS instance. It provides valuable insights into any errors, warnings, or issues encountered within the database system, helping to identify and troublesh oot problems effectively. Monitoring error logs is essential for maintaining the health and reliability of the database.

Visualization Name	Source Field	Description
Audit History	* log event	This visualization presents the audit history of the Amazon RDS instance. It tracks the various log events and activities related to database access, modificat ions, and security-related events. Monitoring the audit logs is crucial for complianc e, detecting unauthorized access, and tracking changes made to the database.
Audit Logs	* db-identifier * audit-ope ration * audit-ip * audit-que ry * audit-retcode * audit-con nection-id * audit-host-name * audit-query-id * audit-user	This visualization provides an overview of the audit logs generated by the Amazon RDS instance. It shows the operations performed on the database, including queries executed, connection details, IP addresses, and associated users. Monitoring audit logs enhances the security and governance of the database, helping to detect suspiciou s activities and track user actions.

You can access the built-in dashboard in Amazon OpenSearch Service to view log data. For more information, see the Access Dashboard.

Amazon RDS/Aurora logs sample dashboard.


Controller
Database Identifier
Select...
Table Name
Select...
Apply changes
Cancel changes
Clear form
Total Log Events Overview
loghub_db_01
progrom_01
loghub_db_02
loghub_db_03
@timestamp per 30 seconds
Slow Query History
Count
@timestamp per 30 seconds
Average Slow Query Time History
Slow Query Time Ta...
Slow Query Time Taken
@timestamp per 30 seconds
Total Slow Queries
52
Total Slow Queries
Top Slow Query IP
Client IP
Average Duration
Count
172.31.87.142 3.537 9
172.31.87.189 3.341 9
172.31.87.138 2.763 8
172.31.90.100 2.372 7
172.31.87.152 3.1 6
172.31.87.167 2.123 6
172.31.30.1 3.351 4
172.31.87.124 3.159 3
Export: Raw Formatted
Slow Query Scatter Plot
Client IP
Query Duration / secs
Average Slow Query Duration
2.98
Average Slow Query
Slow Query Pie
SELECT SLEEP 2 FR...
SELECT intcol1,char...
SELECT * FROM log...
SELECT intcol1,char...
3.85%)
SELEC
Slow Query Table Name Pie
loghub_tb_01
user_tb
order_tb
prods_tb
9.62%)
9.62%)
21.15%)
loghub
Top Slow Query
Slow Query
SELECT * FROM loghub_05
SELECT intcol1,charcol1,charcol2,charcol3,charcol4,charcol5,charcol6,charcol7,charcol8,charcol9,charcol10,charcol11,charcol12,charcol13,charcol14,charcol15,charcol16,ch FROM t1
SELECT intcol1,charcol1,charcol2,charcol3,charcol4,charcol5,charcol6,charcol7,charcol8,charcol9,charcol10,charcol11,charcol12,charcol13,charcol14,charcol15,char
SELECT SLEEP 2 FROM t1
Export: Raw Formatted
Slow Query Logs
1–50 of 52
Time db-identifier sq-db-name sq-table-name sq-user sq-query sq-ip sq-host-name sq-rows-examined sq-rows-sent sq-id sq-duration sq-lock-wait
May 16, 2022 @ 11:17:22.000 loghub_db_01 loghub_db_03 loghub_tb_01 dev-01 SELECT intcol1,charcol1,charcol2,charcol3,charcol4,charcol5,charcol6,charcol7,charcol8,charcol9,charcol10,charcol11,charcol12,charcol13,charcol14,charcol15,charcol16,charcol17,charcol18,charcol19,charcol20,charcol21,charcol22,charcol23,charcol24,charcol25,charcol26,charcol27 FROM t2 172.31.90.100 loghub_host_01 -125 802 67258 2.581 0.252
May 16, 2022 @ 11:20:42.000 loghub_db_01 loghub_db_02 loghub_tb_01 dev-01 SELECT SLEEP 2 FROM t1 172.31.87.189 loghub_host_02 1,606 985 65161 3.95 0.279
May 16, 2022 @ 11:23:35.000 loghub_db_01 loghub_db_03 user_tb dev-01 SELECT SLEEP 2 FROM t1 172.31.87.138 loghub_host_00 891 103 39370 2.43 0.297
May 16, 2022 @ 11:27:48.000 loghub_db_02 loghub_db_01 prods_tb dev-03 SELECT SLEEP 2 FROM t1 172.31.87.152 loghub_host_00 218 761 64521 5.379 0.337
May 16, 2022 @ 11:17:22.000 progrom_01 loghub_db_02 user_tb dev-01 SELECT SLEEP 2 FROM t1 172.31.87.167 loghub_host_01 403 991 54247 2.616 0.356
May 16, 2022 @ 11:28:37.000 loghub_db_01 loghub_db_02 loghub_tb_01 dev-01 SELECT SLEEP 2 FROM t1 172.31.90.100 loghub_host_01 991 1,075 66005 3.85 0.361
Total Deadlock Queries
55
Total Deadlock Queries
Deadlock History
Count
@timestamp per 30 seconds
Deadlock Query Logs
Time db-identifier log-detail deadlock-ip-1 deadlock-action-1 deadlock-os-thread-handl
May 16, 2022 @ 11:29:57.000 loghub_db_01 2022-01-21T05:55:46.858519Z 3330 [Note] InnoDB: Transactions deadlock detected, dumping detailed informa 172.31.87.152 updating 70380721071461
Expanded document
Table JSON
{
"_index": "rds-sse-01-rds-2022-05-13",
"_type": "_doc",
"_id": "-zDRyoABWU95C8pDzhnz",
"_version": 1,
"_score": null,
Total Error Logs
47
Total Error Logs
Error History
Count
Error Logs
1–47 of 47
Time db-identifier err-label err-code err-detail err-sub-system err-thread
May 16, 2022 @ 11:29:51.000 - Warning MY-013172 /rdsdbbin/mysql/bin/mysqld: Shutdown complete (mysqld 8.0.23) Source distribution. Server -98
May 16, 2022 @ 11:29:42.000 - System MY-013172 /rdsdbbin/mysql/bin/mysqld: Shutdown complete (mysqld 8.0.23) Source distribution. Server 101

Create log ingestion (Light Engine)

Using the Centralized Logging with OpenSearch Console

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose the Create a log ingestion button.
4. In the **AWS Services** section, choose **Amazon RDS**.
5. Choose **Light Engine**, choose **Next**.
6. Under **Specify settings**, choose **Automatic** or **Manual** for **RDS log enabling**. The automatic mode will detect your Amazon RDS log configurations and ingest logs from CloudWatch.
   - For **Automatic mode**, choose the Amazon RDS cluster from the dropdown list.
   - For **Manual mode**, enter the **DB identifier**, select the **Database type** and input the CloudWatch log location in **Log type and location**.
   - (Optional) If you are ingesting Amazon RDS/Aurora logs from another account, select a linked account from the **Account** dropdown first.
7. Choose **Next**.
8. In the **Specify Light Engine Configuration** section, if you want to ingest associated templated Grafana dashboards, select **Yes** for the sample dashboard.
9. You can choose an existing Grafana, or if you must import a new one, you can go to Grafana for configuration.
10. Select an S3 bucket to store partitioned logs and define a name for the log table. We have provided a predefined table name, but you can modify it according to your business needs.
11. If needed, change the log processing frequency, which is set to **5** minutes by default, with a minimum processing frequency of **1** minute.
12. In the **Log Lifecycle** section, enter the log merge time and log archive time. We have provided default values, but you can adjust them based on your business requirements.
13. Select **Next**.
14. If desired, add tags.
15. Select **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - RDS Log Ingestion* solution in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Regions		Template
AWS China Regions		Template

1. Log in to the AWS Management Console and select the preceding button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.
2. To launch the stack 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.

   a. Parameters for **Pipeline settings**

Parameter	Default	Description
Pipeline Id	`<Requires input>`	The unique identifier for the pipeline is essential if you must create multiple Application Load Balancer pipelines and write different Application Load Balancer

Parameter	Default	Description
		logs into separate tables. For uniqueness, you can generate a unique pipeline identifier using uuidgenerator.
Staging Bucket Prefix	`AWSLogs/RDS`	The storage directory for logs in the temporary storage area should ensure the uniqueness and non-overlapping of the Prefix for different pipelines.

b. Parameters for **Destination settings**

Parameter	Default	Description
Centralized Bucket Name	*<Requires input>*	Centralized S3 bucket name. For example, centralized-logging-bucket.
Centralized Bucket Prefix	`datalake`	Centralized bucket prefix. By default, the data base location is s3://{Centralized Bucket Name}/{Centralized Bucket Prefix}/amazon_cl_centralized.
Centralized Table Name	`rds`	Table name for writing data to the centralized database. You can modify it if needed.

c. Parameters for **Scheduler settings**

Parameter	Default	Description
LogProcessor Schedule Expression	`rate(5 minutes)`	Task scheduling expression for performing log processing, with a default value of executing the LogProcessor every 5 minutes. Configuration for reference.
LogMerger Schedule Expression	`cron(0 1 * ? )`	Task scheduling expression for performing log merging, with a default value of executing the LogMerger at 1 AM every day. Configuration for reference.
LogArchive Schedule Expression	`cron(0 2 * ? )`	Task scheduling expression for performing log archiving, with a default value of executing the LogArchive at 2 AM every day. Configuration for reference.
Age to Merge	7	Small file retention days, with a default value of 7, indicating that logs older than 7 days will be merged into small files. It can be adjusted as needed.

Parameter	Default	Description
Age to Archive	30	Log retention days, with a default value of 30, indicating that data older than 30 days will be archived and deleted. It can be adjusted as needed.

d. Parameters for **Notification settings**

Parameter	Default	Description
Notification Service	SNS	Notification method for alerts. If your main stack is using China, you can only choose the SNS method. If your main stack is using Global, you can choose either the SNS or SES method.

Parameter	Default	Description
Recipients	*<Requires input>*	Alert notification: If the Notification Service is SNS, enter the SNS Topic ARN here, ensuring that you have the necessary permissions. If the Notification Service is SES, enter the email addresses separated by commas here, ensuring that the email addresses are already Verified Identitie s in SES. The adminEmai l provided during the creation of the main stack will receive a verification email by default.

e. Parameters for **Dashboard settings**

Parameter	Default	Description
Import Dashboards	FALSE	Whether to import the Dashboard into Grafana, with a default value of false. If set to true, you must provide the Grafana URL and Grafana Service Account Token.
Grafana URL	*<Requires input>*	Grafana access URL，for example: https://a lb-72277319.us-wes t-2.elb.amazonaws.com.

Parameter	Default	Description
Grafana Service Account Token	*<Requires input>*	Grafana Service Account Token Service Account Token created in Grafana.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review and create** page, review and confirm the settings. Check the box acknowledging that the template creates IAM resources.
9. Choose **Submit** 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.

View dashboard

Amazon RDS/Aurora logs sample dashboard.


4 Mil
2
3
2 Mil
2 Mil
8
3 s
4
24 K

Amazon CloudFront logs

CloudFront standard logs provide detailed records about every request made to a distribution.

You can create a log ingestion into Amazon OpenSearch Service either by using the Centralized Logging with OpenSearch console or by deploying a standalone CloudFormation stack.

> **Important**
>
> The CloudFront logging bucket must be the same Region as the Centralized Logging with OpenSearch solution.
> The Amazon OpenSearch Service index is rotated on a daily basis by default, and you can adjust the index in the Additional Settings.

Create log ingestion (OpenSearch Engine)

Using the Centralized Logging with OpenSearch Console

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose the Create a log ingestion button.
4. In the AWS Services section, choose Amazon CloudFront.
5. Choose **Next**.
6. Under **Specify settings**, choose **Automatic** or **Manual** for **CloudFront logs enabling**. The automatic mode will detect the CloudFront log location automatically.
   - For **Automatic mode**, choose the CloudFront distribution and Log Type from the dropdown list.
   - For Standard Log, the solution will automatically detect the log location if logging is enabled.
   - For Real-time log, the solution will prompt you for confirmation to create or replace the CloudFront real-time log configuration.
   - For **Manual mode**, enter the **CloudFront Distribution ID** and **CloudFront Standard Log location**. (Note that CloudFront real-time log is not supported in Manual mode)
   - (Optional) If you are ingesting CloudFront logs from another account, select a linked account from the **Account** dropdown list first.
7. Choose **Next**.

8. In the Specify OpenSearch domain section, select an imported domain for the Amazon OpenSearch Service domain.
9. Choose **Yes** for **Sample dashboard** if you want to ingest an associated templated Amazon OpenSearch Service dashboard.
10. You can change the **Index Prefix** of the target Amazon OpenSearch Service index if needed. The default prefix is the CloudFront distribution ID.
11. In the **Log Lifecycle** section, input the number of days to manage the Amazon OpenSearch Service index lifecycle. The Centralized Logging with OpenSearch will create the associated Index State Management (ISM) policy automatically for this pipeline.
12. In the **Log processor settings** section, choose **Log processor type**, configure the Lambda concurrency if needed, and then choose **Next**.
13. Add tags if needed.
14. Choose **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - CloudFront Standard Log Ingestion* template in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Regions		Template
AWS China Regions		Template

1. Log in to the AWS Management Console and select the preceding button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.
2. To launch the stack 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.

Parameter	Default	Description
Log Bucket Name	`<Requires input>`	The S3 bucket name that stores the logs.
Log Bucket Prefix	`<Requires input>`	The S3 bucket path prefix that stores the logs.
Log Source Account ID	*Optional input*	The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at **Step 1** will be used.
Log Source Region	*Optional input*	The AWS Region of the S3 bucket. By default, the Region you selected at **Step 2** will be used.
Log Source Account Assume Role	*Optional input*	The IAM Role ARN used for cross-account log ingestion . Required for cross-account log ingestion (add a member account first).
KMS-CMK ARN	*Optional input*	The KMS-CMK ARN for encryption. Leave it blank to create a new AWS KMS key.

Parameter	Default	Description
Enable OpenSearch Ingestion as processor	*Optional input*	Ingestion table ARN. Leave empty if you do not use OSI as Processor.
Amazon S3 Backup Bucket	*<Requires input>*	The Amazon S3 backup bucket name to store the failed ingestion logs.
Engine Type	OpenSearch	The engine type of the OpenSearch.
OpenSearch Domain Name	*<Requires input>*	The domain name of the Amazon OpenSearch Service cluster.
OpenSearch Endpoint	*<Requires input>*	The OpenSearch endpoint URL. For example, vpc-your_opensearch_domain_name-xcvgw6uu2o6zafsiefxubwuohe.us-east-1.es.amazonaws.com
Index Prefix	*<Requires input>*	The common prefix of OpenSearch index for the log. The index name will be <Index Prefix>-<Log Type>-<Other Suffix>.
Create Sample Dashboard	Yes	Whether to create a sample OpenSearch dashboard.
VPC ID	*<Requires input>*	Select a VPC that has access to the OpenSearch domain. The log processing Lambda will reside in the selected VPC.

Parameter	Default	Description
Subnet IDs	*<Requires input>*	Select at least two subnets that have access to the OpenSearch domain. The log processing Lambda will reside in the subnets. Make sure that the subnets have access to the Amazon S3 service.
Security Group ID	*<Requires input>*	Select a Security Group that will be associated with the log processing Lambda. Make sure that the Security Group has access to the OpenSearch domain.
Number Of Shards	5	Number of shards to distribute the index evenly across all data nodes. Keep the size of each shard between 10-50 GB.
Number of Replicas	1	Number of replicas for OpenSearch Index. Each replica is a full copy of an index. If the OpenSearch option is set to Domain with standby, you need to configure it to 2.

Parameter	Default	Description
Age to Warm Storage	*Optional input*	The age required to move the index into warm storage (for example, 7d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when warm storage is enabled in OpenSearch.
Age to Cold Storage	*Optional input*	The age required to move the index into cold storage (for example, 30d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when cold storage is enabled in OpenSearch.
Age to Retain	*Optional input*	The age to retain the index (for example, 180d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). If the value is "", the index will not be deleted.
Rollover Index Size	*Optional input*	The minimum size of the shard storage required to roll over the index (for example, 30GB).

Parameter	Default	Description
Index Suffix	`yyyy-MM-dd`	The common suffix format of OpenSearch index for the log(Example: yyyy-MM-dd, yyyy-MM-dd-HH). The index name will be <Index Prefix>-<Log Type>-<Index Suffix>-000001.
Compression type	`best_compression`	The compression type to use to compress stored data. Available values are best_compression and default.
Refresh Interval	`1s`	How often the index should refresh, which publishes its most recent changes and makes them available for searching. Can be set to -1 to disable refreshing. Default is 1s.
Plugins	*Optional input*	List of plugins delimited by comma. Leave it blank if there are no available plugins to use. Valid inputs are user_agent, geo_ip.
EnableS3Notification	`True`	An option to enable or disable notifications for Amazon S3 buckets. The default option is recommended for most cases.

Parameter	Default	Description
LogProcessorRoleName	*Optional input*	Specify a role name for the log processor. The name should NOT duplicate an existing role name. If no name is specified, a random name is generated.
QueueName	*Optional input*	Specify a queue name for an SQS. The name should NOT duplicate an existing queue name. If no name is given, a random name will be generated.

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

View dashboard

The dashboard includes the following visualizations.

Visualization Name	Source Field	Description
Total Requests	* log event	Displays the total number of viewer requests received by the Amazon CloudFron t, for all HTTP methods and for both HTTP and HTTPS requests.

Visualization Name	Source Field	Description
Edge Locations	* x-edge-location	Shows a pie chart represent ing the proportion of the locations of CloudFront edge servers.
Request History	* log event	Presents a bar chart that displays the distribution of events over time.
Unique Visitors	* c-ip	Displays unique visitors identified by client IP address.
Cache Hit Rate	* sc-bytes	Shows the proportion of your viewer requests that are served directly from the CloudFront cache instead of going to your origin servers for content.

Visualization Name	Source Field	Description
Result Type	* x-edge-response-result-type	Shows the percentage of hits, misses, and errors to the total viewer requests for the selected CloudFront distribut ion: * Hit - A viewer request for which the object is served from a CloudFront edge cache. In access logs, these are requests for which the value of x-edge-response-result-type is Hit * Miss - A viewer request for which the object isn't currently in an edge cache, so CloudFront must get the object from your origin. In access logs, these are requests for which the value of x-edge-response-re sult-type is Miss. * Error - A viewer request that resulted in an error, so CloudFront didn't serve the object. In access logs, these are requests for which the value of x-edge-response-result-type is Error, LimitExceeded, or CapacityE xceeded. The chart does not include refresh hits—requests for objects that are in the edge cache but that have expired. In access logs, refresh hits are

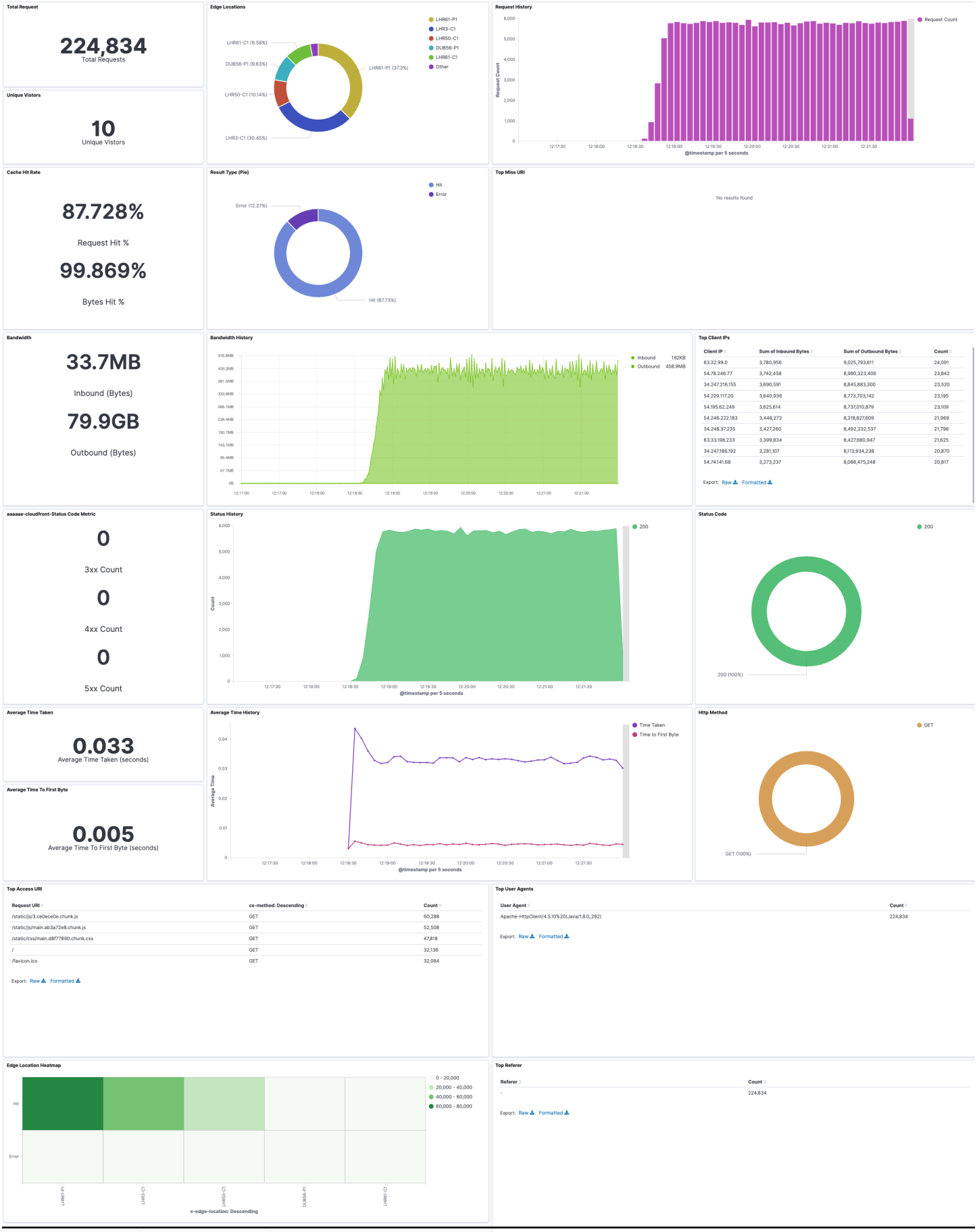
Visualization Name	Source Field	Description
		requests for which the value of x-edge-response-result-type is RefreshHit.
Top Miss URI	* cs-uri-stem * cs-method	Shows top 10 of the requested objects that are not in the cache.
Bandwidth	* cs-bytes * sc-bytes	Provides insights into data transfer activities from the locations of CloudFront edge.
Bandwidth History	* cs-bytes * sc-bytes	Shows the historical trend of the data transfer activitie s from the locations of CloudFront edge.
Top Client IPs	* c-ip	Provides the top 10 IP address accessing your Amazon CloudFront.
Status Code Count	* sc-status	Displays the count of requests made to the Amazon CloudFront, grouped by HTTP status codes(e.g., 200, 404, 403, etc.).
Status History	* @timestamp * sc-status	Shows the historical trend of HTTP status codes returned by the Amazon CloudFront over a specific period of time.
Status Code	* sc-status	Identifies the users or IAM roles responsible for changes to EC2 resources, assisting in accountability and tracking of modifications.

Visualization Name	Source Field	Description
Average Time Taken	* time-taken	This visualization calculates and presents the average time taken for various operations in the Amazon CloudFront (e.g., average time for GET, PUT requests, etc.).
Average Time History	* time-taken * time-to-first-byte * @timestamp	Shows the historical trend of the average time taken for various operations in the Amazon CloudFront.
Http Method	* cs-method	Displays the count of requests made to the Amazon CloudFront using a pie chart, grouped by HTTP request method names (for example, POST, GET, HEAD).
Average Time To First Byte	* time-to-first-byte	Provides the average time taken in seconds by the origin server to respond back with the first byte of the response.
Top Request URIs	* cs-uri-stem * cs-method	Provides the top 10 request URIs accessing your CloudFront.
Top User Agents	* cs-user-agent	Provides the top 10 user agents accessing your CloudFront.
Edge Location Heatmap	* x-edge-location * x-edge-result-type	Shows a heatmap representing the result type of each edge location.

Visualization Name	Source Field	Description
Top Referrers	* cs-referer	Top 10 referrers with the Amazon CloudFront access.
Top Countries or Regions	* c_country	Top 10 countries with the Amazon CloudFront access.

You can access the built-in dashboard in Amazon OpenSearch Service to view log data. For more information, see the Access Dashboard.

CloudFront logs sample dashboard.


Total Request
224,834
Total Requests
Edge Locations
LHR61-P1
LHR3-C1
LHR50-C1
DUB56-P1
LHR61-C1
Other
LHR61-C1 (9.59%)
DUB56-P1 (9.63%)
LHR50-C1 (10.14%)
LHR3-C1 (30.45%)
LHR61-P1 (37.3%)
Request History
Request Count
@timestamp per 5 seconds
Unique Vistors
10
Unique Vistors
Cache Hit Rate
87.728%
Request Hit %
99.869%
Bytes Hit %
Result Type (Pie)
Hit
Error
Error (12.27%)
Hit (87.73%)
Top Miss URI
No results found
Bandwidth
33.7MB
Inbound (Bytes)
79.9GB
Outbound (Bytes)
Bandwidth History
Inbound 192KB
Outbound 458.9MB
Top Client IPs
Client IP
Sum of Inbound Bytes
Sum of Outbound Bytes
Count
63.32.99.0 3,780,956 9,025,793,611 24,091
54.78.246.77 3,742,458 8,980,323,406 23,842
34.247.216.155 3,690,591 8,845,883,300 23,520
54.229.117.20 3,640,936 8,773,703,142 23,195
54.195.62.249 3,625,614 8,737,010,879 23,109
54.246.222.183 3,448,273 8,318,827,609 21,969
34.248.37.235 3,427,260 8,492,332,537 21,796
63.33.198.233 3,399,834 8,427,680,947 21,625
34.247.186.192 3,281,107 8,113,934,238 20,870
54.74.141.68 3,273,237 8,088,475,248 20,817
Export: Raw Formatted
aaaaaa-cloudfront-Status Code Metric
0
3xx Count
0
4xx Count
0
5xx Count
Status History
200
Count
@timestamp per 5 seconds
Status Code
200
200 (100%)
Average Time Taken
0.033
Average Time Taken (seconds)
Average Time To First Byte
0.005
Average Time To First Byte (seconds)
Average Time History
Time Taken
Time to First Byte
Average Time
@timestamp per 5 seconds
Http Method
GET
GET (100%)
Top Access URI
Request URI
cs-method: Descending
Count
/static/js/3.ce0ece0e.chunk.js GET 60,288
/static/js/main.ab3a72e8.chunk.js GET 52,508
/static/css/main.d8f77890.chunk.css GET 47,818
/ GET 32,136
/favicon.ico GET 32,084
Export: Raw Formatted
Top User Agents
User Agent
Count
Apache-HttpClient/4.5.10%20(Java/1.8.0_292) 224,834
Export: Raw Formatted
Edge Location Heatmap
0 - 20,000
20,000 - 40,000
40,000 - 60,000
60,000 - 80,000
Hit
Error
LHR61-P1
LHR3-C1
LHR50-C1
DUB56-P1
LHR61-C1
x-edge-location: Descending
Top Referer
Referer
Count
- 224,834
Export: Raw Formatted

Create log ingestion (Light Engine)

Using the Centralized Logging with OpenSearch Console

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose the Create a log ingestion button.
4. In the AWS Services section, choose Amazon CloudFront.
5. Choose **Next**.
6. Under **Specify settings**, choose **Automatic** or **Manual** for **CloudFront logs enabling**. The automatic mode will detect the CloudFront log location automatically.
   - For **Automatic mode**, choose the CloudFront distribution and Log Type from the dropdown list.
   - For Standard Log, the solution will automatically detect the log location if logging is enabled.
   - For Real-time log, the solution will prompt you for confirmation to create or replace the CloudFront real-time log configuration.
   - For **Manual mode**, enter the **CloudFront Distribution ID** and **CloudFront Standard Log location**. (Note that CloudFront real-time log is not supported in Manual mode)
   - (Optional) If you are ingesting CloudFront logs from another account, select a linked account from the **Account** dropdown list first.
7. Choose **Next**.
8. Choose **Log Processing Enriched fields** if needed. The available plugins are **location** and **OS/User Agent**. Enabling rich fields increases data processing latency and processing costs. By default, it is not selected.
9. In the **Specify Light Engine Configuration** section, if you want to ingest associated templated Grafana dashboards, select **Yes** for the sample dashboard.
10. You can choose an existing Grafana, or if you must import a new one, you can go to Grafana for configuration.
11. Select an S3 bucket to store partitioned logs and define a name for the log table. We have provided a predefined table name, but you can modify it according to your business needs.
12. If needed, change the log processing frequency, which is set to **5** minutes by default, with a minimum processing frequency of **1** minute.

13In the **Log Lifecycle** section, enter the log merge time and log archive time. We have provided default values, but you can adjust them based on your business requirements.

14Select **Next**.

15If desired, add tags.

16Select **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - CloudFront Log Ingestion* solution in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Regions		Template
AWS China Regions		Template

1. Log in to the AWS Management Console and select the preceding button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.
2. To launch the stack 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.

   a. Parameters for **Pipeline settings**

Parameter	Default	Description
Pipeline Id	*<Requires input>*	The unique identifier for the pipeline is essential if you must create multiple Application Load Balancer pipelines and write different Application Load Balancer logs into separate tables. To ensure uniqueness, you can generate a unique pipeline identifier using uuidgenerator.
Staging Bucket Prefix	`AWSLogs/CloudFrontLogs`	The storage directory for logs in the temporary storage area should ensure the uniqueness and non-overlapping of the Prefix for different pipelines.

b. Parameters for **Destination settings**

Parameter	Default	Description
Centralized Bucket Name	*<Requires input>*	Centralized S3 bucket name. For example, centralized-logging-bucket.
Centralized Bucket Prefix	`datalake`	Centralized bucket prefix. By default, the data base location is s3://{Centralized Bucket Name}/{Centralized Bucket Prefix}/amazon_cl_centralized.

Parameter	Default	Description
Centralized Table Name	`CloudFront`	Table name for writing data to the centralized database. You can modify it if needed.

c. Parameters for **Scheduler settings**

Parameter	Default	Description
LogProcessor Schedule Expression	`rate(5 minutes)`	Task scheduling expressio n for performing log processing, with a default value of executing the LogProcessor every 5 minutes. Configuration for reference.
LogMerger Schedule Expression	`cron(0 1 * ? )`	Task scheduling expression for performing log merging, with a default value of executing the LogMerger at 1 AM every day. Configura tion for reference.
LogArchive Schedule Expression	`cron(0 2 * ? )`	Task scheduling expression for performing log archiving , with a default value of executing the LogArchive at 2 AM every day. Configura tion for reference.

Parameter	Default	Description
Age to Merge	7	Small file retention days, with a default value of 7, indicating that logs older than 7 days will be merged into small files. It can be adjusted as needed.
Age to Archive	30	Log retention days, with a default value of 30, indicating that data older than 30 days will be archived and deleted. It can be adjusted as needed.

d. Parameters for **Notification settings**

Parameter	Default	Description
Notification Service	SNS	Notification method for alerts. If your main stack is using China, you can only choose the SNS method. If your main stack is using Global, you can choose either the SNS or SES method.

Parameter	Default	Description
Recipients	*<Requires input>*	Alert notification: If the Notification Service is SNS, enter the SNS Topic ARN here to have the necessary permissions. If the Notification Service is SES, enter the email addresses separated by commas here, ensuring that the email addresses are already Verified Identities in SES. The adminEmail provided during the creation of the main stack will receive a verification email by default.

e. Parameters for **Dashboard settings**

Parameter	Default	Description
Import Dashboards	FALSE	Whether to import the Dashboard into Grafana, with a default value of false. If set to true, you must provide the Grafana URL and Grafana Service Account Token.
Grafana URL	*<Requires input>*	Grafana access URL for example: https://alb-72277319.us-west-2.elb.amazonaws.com.

Parameter	Default	Description
Grafana Service Account Token	*<Requires input>*	Grafana Service Account Token Service Account Token created in Grafana.

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

View dashboard

The dashboard includes the following visualizations.

Visualization Name	Source Field	Description
Filters	Filters	The following data can be filtered by query filter conditions.
Total Requests	log event	Displays the total number of viewer requests received by the Amazon CloudFron t, for all HTTP methods and for both HTTP and HTTPS requests.
Unique Visitors	c-ip	Displays unique visitors identified by client IP address.

Visualization Name	Source Field	Description
Requests History	log event	Presents a bar chart that displays the distribution of events over time.
Request By Edge Location	x-edge-location	Shows a pie chart represent ing the proportion of the locations of CloudFront edge servers.
HTTP Status Code	sc-status	Displays the count of requests made to the Amazon CloudFront, grouped by HTTP status codes (e.g., 200, 404, 403, etc.).
Status Code History	sc-status	Shows the historical trend of HTTP status codes returned by the Amazon CloudFront over a specific period of time.
Status Code Pie	sc-status	Represents the distribution of requests based on different HTTP status codes using a pie chart.
Average Processing Time	time-taken time-to-first-byte	This visualization calculate s and presents the average time taken for various operations in the Amazon CloudFront (for example, average time for GET, PUT requests).

Visualization Name	Source Field	Description
Avg. Processing Time History	time-taken time-to-first-byte	Shows the historical trend of the average time taken for various operations in the Amazon CloudFront.
Avg. Processing Time History	time-taken time-to-first-byte	Shows the historical trend of the average time taken for various operations in the Amazon CloudFront.
HTTP Method	cs-method	Displays the count of requests made to the Amazon CloudFront using a pie chart, grouped by HTTP request method names (for example, POST, GET, HEAD).
Total Bytes	cs-bytes sc-bytes	Provides insights into data transfer activities, including the total bytes transferred.
Response Bytes History	cs-bytes sc-bytes	Displays the historical trend of the received bytes, send bytes.

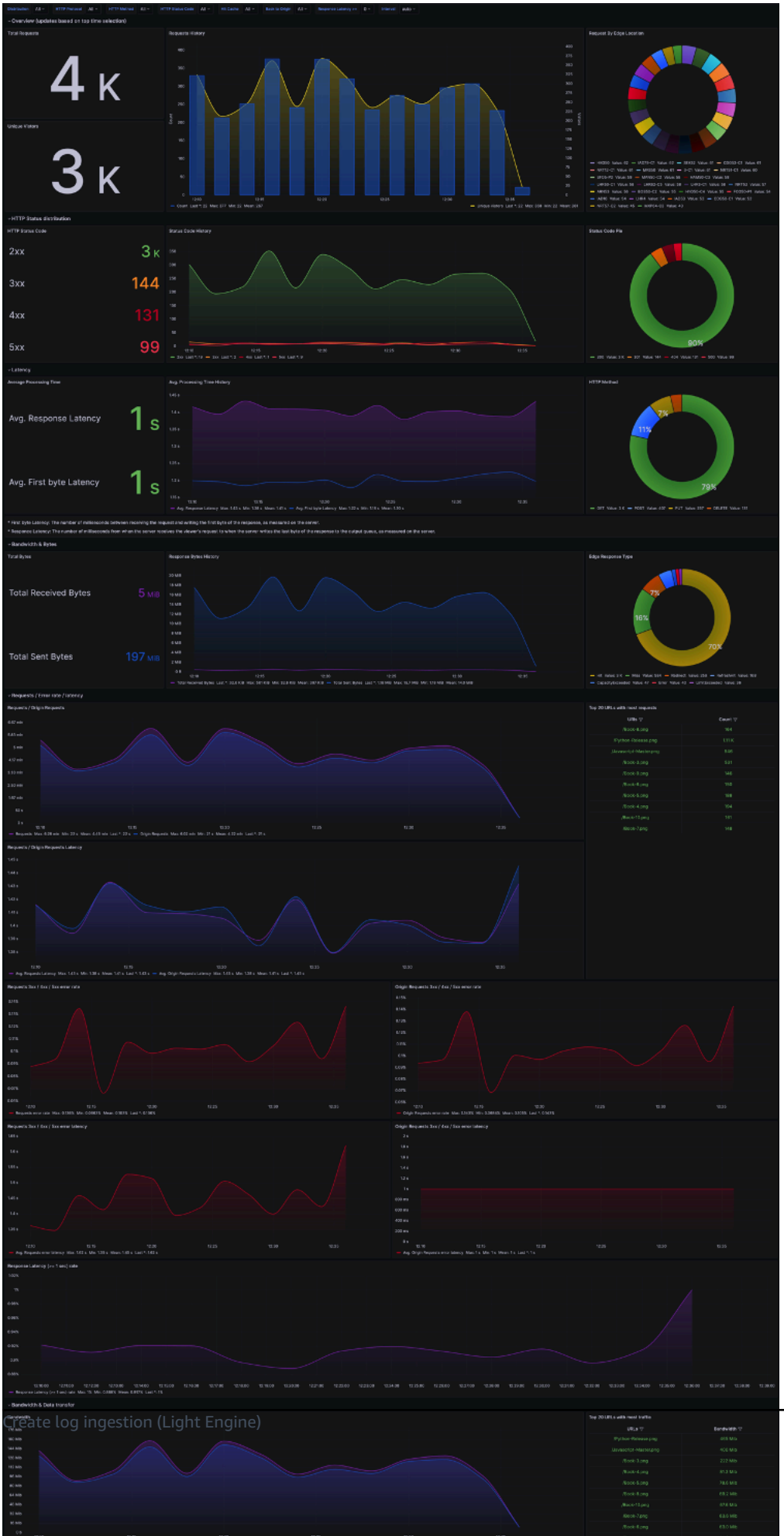
Visualization Name	Source Field	Description
Edge Response Type	x-edge-response-result-type	Shows the percentage of hits, misses, and errors to the total viewer requests for the selected CloudFront distribution: - Hit - A viewer request for which the object is served from a CloudFron t edge cache. In access logs, these are requests for which the value of x-edge-response-result-type is Hit. - Miss - A viewer request for which the object isn't currently in an edge cache, so CloudFront must get the object from your origin. In access logs, these are requests for which the value of x-edge-response-re sult-type is Miss. - Error - A viewer request that resulted in an error, so CloudFront didn't serve the object. In access logs, these are requests for which the value of x-edge-response-result-type is Error, LimitExceeded, or CapacityE xceeded. The chart does not include refresh hits—requests for objects that are in the edge cache but that have expired. In access logs, refresh hits are requests for which the value of x-edge-response-re sult-type is RefreshHit.

Visualization Name	Source Field	Description
Requests / Origin Requests	log event	Displays the number of requests made to CloudFront and the number of requests back to the origin.
Requests / Origin Requests Latency	log event time-taken	Displays the request latency from the client to CloudFront and the request latency back to the origin.
Top 20 URLs with most requests	log event	Top 20 URLs based on the number of requests.
Requests 3xx / 4xx / 5xx error rate	log event sc-status	Displays the ratio of 3xx/4xx/5xx status codes from the client to CloudFront.
Origin Requests 3xx / 4xx / 5xx error rate	log event sc-status x-edge-detailed-result-type	Display the proportion of 3xx/4xx/5xx status codes returned to the origin.
Requests 3xx / 4xx / 5xx error latency	log event sc-status time-taken	Displays the latency from the client to CloudFront for 3xx/4xx/5xx status codes.
Origin Requests 3xx / 4xx / 5xx error latency	log event sc-status x-edge-detailed-result-type time-taken	Displays the delay in returning to the source 3xx/4xx/5xx status code.
Response Latency (>= 1sec) rate	log event time-taken	Display the proportion of delay above 1s.
Bandwidth	sc-bytes	Displays the bandwidth from the client to CloudFront and the bandwidth back to the origin.

Visualization Name	Source Field	Description
Data transfer	sc-bytes	Display the response traffic.
Top 20 URLs with most traffic	cs-uri-stem sc-bytes	Top 20 URLs calculated by traffic.
Cache hit rate (calculated using requests)	log event x-edge-result-type	Displays the cache hit ratio calculated by the number of requests.
Cache hit rate (calculated using bandwidth)	log event sc-bytes x-edge-result-type	Displays the cache hit ratio calculated by bandwidth.
Cache Result	log event x-edge-result-type	Displays the number of requests of various x-edge-result-types, such as the number of requests that hit the cache and the number of requests that missed the cache.
Cache Result Latency	log event sc-bytes x-edge-result-type	Displays the request latency of various x-edge-result-types, such as the request latency that hits the cache and the request latency that misses the cache.
Requests by OS	ua_os	Displays the count of requests made to the Application Load Balancer, grouped by user agent OS.
Requests by Device	ua_device	Displays the count of requests made to the Application Load Balancer, grouped by user agent device.

Visualization Name	Source Field	Description
Requests by Browser	ua_browser	Displays the count of requests made to the Application Load Balancer, grouped by user agent browser.
Requests by Category	ua_category	Displays the count of categories made to the Application Load Balancer, grouped by user agent category (for example, PC, Mobile, Tablet).
Requests by Countries or Regions	geo_iso_code	Displays the count of requests made to the Application Load Balancer (grouped by the corresponding country or Region resolved by the client IP).
Top Countries or Regions	geo_country	Top 10 countries with the Application Load Balancer Access.
Top Cities	geo_city	Top 10 cities with Application Load Balancer Access.

CloudFront logs sample dashboard.


Overview (updates based on top time selection)
Total Requests
4 K
Unique Visitors
3 K
Requests History
Request By Edge Location
HTTP Status distribution
HTTP Status Code
2xx 3 K
3xx 144
4xx 131
5xx 99
Status Code History
Status Code Pie
90%
Latency
Average Processing Time
Avg. Response Latency 1 s
Avg. First byte Latency 1 s
Avg. Processing Time History
HTTP Method
11%
7%
79%
Bandwidth & Bytes
Total Bytes
Total Received Bytes 5 MiB
Total Sent Bytes 197 MiB
Response Bytes History
Edge Response Type
7%
16%
70%
Requests / Error rate / Latency
Requests / Origin Requests
Top 20 URLs with most requests
Requests / Origin Requests Latency
Requests 3xx / 4xx / 5xx error rate
Origin Requests 3xx / 4xx / 5xx error rate
Requests 3xx / 4xx / 5xx error latency
Origin Requests 3xx / 4xx / 5xx error latency
Response Latency [<= 1 sec] rate
Bandwidth & Data transfer
Top 20 URLs with most traffic

AWS Lambda logs

AWS Lambda automatically monitors Lambda functions on your behalf and sends function metrics to Amazon CloudWatch.

You can create a log ingestion into Amazon OpenSearch Service either by using the Centralized Logging with OpenSearch console or by deploying a standalone CloudFormation stack.

> **Important**
>
> The Lambda Region must be the same as the Centralized Logging with OpenSearch solution.
> The Amazon OpenSearch Service index is rotated on a daily basis by default, and you can adjust the index in the Additional Settings.

Create log ingestion (OpenSearch Engine)

Using the Centralized Logging with OpenSearch Console

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose the Create a log ingestion button.
4. In the **AWS Services** section, choose **AWS Lambda**.
5. Choose **Next**.
6. Under **Specify settings**, choose the Lambda function from the dropdown list. (Optional) If you are ingesting logs from another account, select a linked account from the **Account** dropdown first.
7. Choose **Next**.
8. In the **Specify OpenSearch domain** section, select an imported domain for the Amazon OpenSearch Service domain.
9. Choose **Yes** for **Sample dashboard** if you want to ingest an associated templated Amazon OpenSearch Service dashboard.
10. You can change the **Index Prefix** of the target Amazon OpenSearch Service index if needed. The default prefix is the Lambda function name.

11 In the **Log Lifecycle** section, input the number of days to manage the Amazon OpenSearch Service index lifecycle. The Centralized Logging with OpenSearch will create the associated Index State Management (ISM) policy automatically for this pipeline.

12 In the **Log processor settings** section, choose **Log processor type**, configure the Lambda concurrency if needed, and then choose **Next**.

13 Add tags if needed.

14 Choose **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - Lambda Log Ingestion* solution in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Regions		Template
AWS China Regions		Template

1. Log in to the AWS Management Console and select the button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.
2. To launch the Centralized Logging with OpenSearch 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.

Parameter	Default	Description
Log Bucket Name	*<Requires input>*	The S3 bucket name to export the logs.
Log Bucket Prefix	*<Requires input>*	The S3 bucket path prefix thatthat stores the logs.
Log Source Account ID	*Optional input*	The AWS Account ID of the CloudWatch log group. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at **Step 1** will be used.
Log Source Region	*Optional input*	The AWS Region of the CloudWatch log group. By default, the Region you selected at **Step 2** will be used.
Log Source Account Assume Role	*Optional input*	The IAM Role ARN used for cross-account log ingestion . Required for cross-account log ingestion (add a member account first).
KMS-CMK ARN	*Optional input*	The KMS-CMK ARN for Amazon SQS encryption. Leave it blank to create a new AWS KMS key.
Enable OpenSearch Ingestion as processor	*Optional input*	Ingestion table ARN. Leave empty if you do not use OSI as Processor.

Parameter	Default	Description
Log Group Names	*<Requires input>*	The names of the CloudWatch log group for the logs.
S3 Backup Bucket	*<Requires input>*	The S3 backup bucket name to store the failed ingestion logs.
Engine Type	OpenSearch	The engine type of the OpenSearch.
OpenSearch Domain Name	*<Requires input>*	The domain name of the Amazon OpenSearch Service cluster.
OpenSearch Endpoint	*<Requires input>*	The OpenSearch endpoint URL. For example, vpc-your_opensearch_domain_name-xcvgw6uu2o6zafsiefxubwuohe.us-east-1.es.amazonaws.com
Index Prefix	*<Requires input>*	The common prefix of OpenSearch index for the log. The index name will be <Index Prefix>-<Log Type>-<Other Suffix>.
Create Sample Dashboard	Yes	Whether to create a sample OpenSearch dashboard.
VPC ID	*<Requires input>*	Select a VPC that has access to the OpenSearch domain. The log processing Lambda will be resides in the selected VPC.

Parameter	Default	Description
Subnet IDs	*<Requires input>*	Select at least two subnets that has access to the OpenSearch domain. The log processing Lambda will reside in the subnets. Make sure that the subnets have access to the Amazon S3 service.
Security Group ID	*<Requires input>*	Select a Security Group that will be associated to the log processing Lambda. Make sure that the Security Group has access to the OpenSearch domain.
Number Of Shards	5	Number of shards to distribute the index evenly across all data nodes. Keep the size of each shard between 10-50 GiB.
Number of Replicas	1	Number of replicas for OpenSearch Index. Each replica is a full copy of an index. If the OpenSearch option is set to Domain with standby, you need to configure it to 2.

Parameter	Default	Description
Age to Warm Storage	*Optional input*	The age required to move the index into warm storage (for example, 7d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when warm storage is enabled in OpenSearch.
Age to Cold Storage	*Optional input*	The age required to move the index into cold storage (for example, 30d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when cold storage is enabled in OpenSearch.
Age to Retain	*Optional input*	The age to retain the index (for example, 180d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). If the value is "", the index will not be deleted.
Rollover Index Size	*Optional input*	The minimum size of the shard storage required to roll over the index (for example, 30GB).

Parameter	Default	Description
Index Suffix	`yyyy-MM-dd`	The common suffix format of OpenSearch index for the log(for example, yyyy-MM-dd, yyyy-MM-dd-HH). The index name will be <Index Prefix>-<Log Type>-<Index Suffix>-000001.
Compression type	`best_compression`	The compression type to use to compress stored data. Available values are best_compression and default.
Refresh Interval	`1s`	How often the index should refresh, which publishes its most recent changes and makes them available for searching. Can be set to -1 to disable refreshing. Default is 1s.
EnableS3Notification	`True`	An option to enable or disable notifications for Amazon S3 buckets. The default option is recommend ed for most cases.
LogProcessorRoleName	*Optional input*	Specify a role name for the log processor. The name should NOT duplicate an existing role name. If no name is specified, a random name is generated.

Parameter	Default	Description
QueueName	*Optional input*	Specify a queue name for an Amazon SQS queue. The name should NOT duplicate an existing queue name. If no name is given, a random name will be generated.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review and create** page, review and confirm the settings. Check the box acknowledging that the template creates AWS Identity and Access Management (IAM) resources.
9. Choose **Submit** 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 15 minutes.

View dashboard

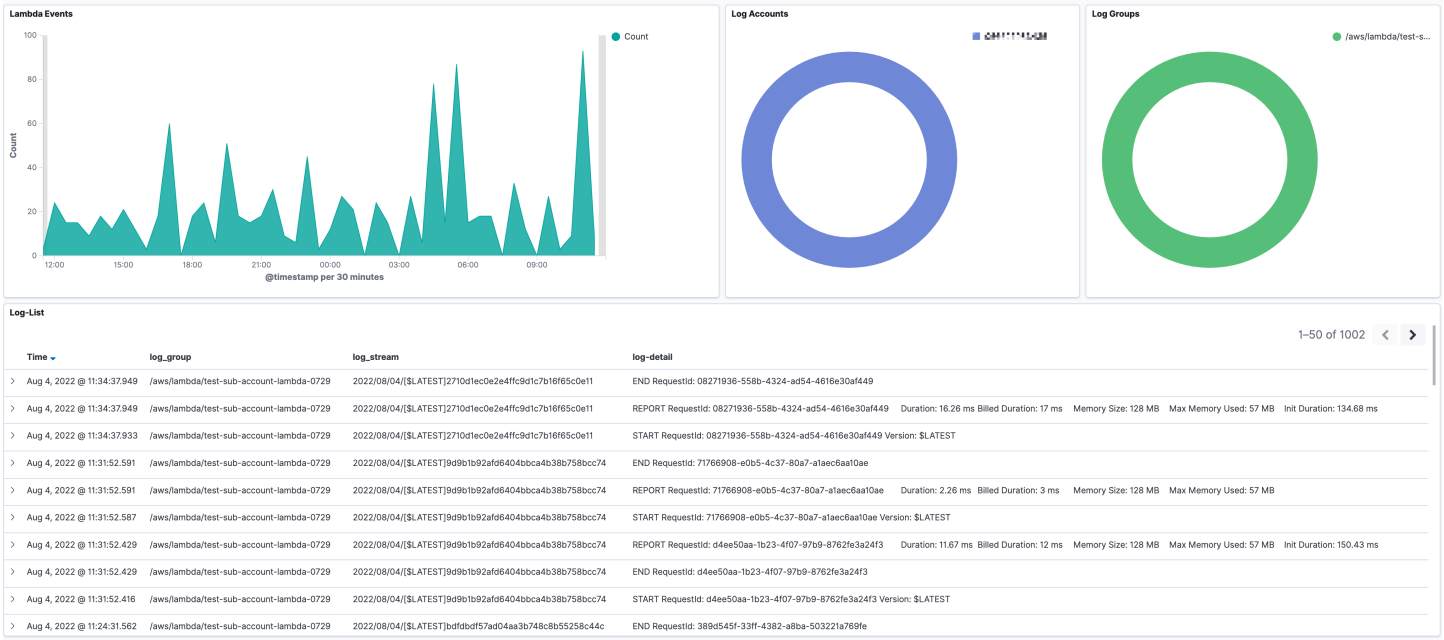
The dashboard includes the following visualizations.

Visualization Name	Source Field	Description
Lambda Events	* log event	Presents a chart that displays the distribution of events over time.
Log Accounts	* owner	Shows a pie chart represent ing the proportion of log events from different AWS accounts (owners).
Log Groups	* log_group	Displays a pie chart depicting the distribution of log events among various log groups in the Lambda environment.

Visualization Name	Source Field	Description
Log-List	* time * log_group * log_stream * log_detail	Provides a detailed list of log events, including timestamps, log groups, log streams, and log details.

You can access the built-in dashboard in Amazon OpenSearch Service to view log data. For more information, see the Access Dashboard.

Lambda logs sample dashboard.



Application Load Balancer logs

Application Load Balancer access logs capture detailed information about requests sent to your load balancer. Application Load Balancer publishes a log file for each load balancer node every 5 minutes.

You can create a log ingestion into Amazon OpenSearch Service either by using the Centralized Logging with OpenSearch console or by deploying a standalone CloudFormation stack.

> **Important**
>
> The Elastic Load Balancing logging bucket must be the same as the Centralized Logging with OpenSearch solution.
> The Amazon OpenSearch Service index is rotated on a daily basis by default, and you can adjust the index in the Additional Settings.

Create log ingestion (OpenSearch Engine)

Using the Centralized Logging with OpenSearch Console

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose the Create a log ingestion button.
4. In the AWS Services section, choose **Elastic Load Balancer**.
5. Choose **Next**.
6. Under Specify settings, choose Automatic or Manual.
   - For **Automatic** mode, choose an Application Load Balancer in the dropdown list. (If the selected Application Load Balancer access log is not enabled, choose **Enable** to enable the Application Load Balancer access log.)
   - For Manual mode, enter the Application Load Balancer identifier and Log location.
   - (Optional) If you are ingesting logs from another account, select a linked account from the **Account** dropdown first.
7. Choose **Next**.
8. In the Specify OpenSearch domain section, select an imported domain for the Amazon OpenSearch Service domain.
9. Choose **Yes** for **Sample dashboard** if you want to ingest an associated templated Amazon OpenSearch Service dashboard.
10. You can change the **Index Prefix** of the target Amazon OpenSearch Service index if needed. The default prefix is the Load Balancer Name.
11. In the **Log Lifecycle** section, input the number of days to manage the Amazon OpenSearch Service index lifecycle. The Centralized Logging with OpenSearch will create the associated Index State Management (ISM) policy automatically for this pipeline.

12. In the **Log Lifecycle** section, enter the number of days to manage the Amazon OpenSearch Service index lifecycle. Centralized Logging with OpenSearch will create the associated Index State Management (ISM) policy automatically for this pipeline.
13. In the **Select log processor** section, choose the log processor.
    a. When selecting Lambda as a log processor, you can configure the Lambda concurrency if needed.
    b. (Optional) OSI as log processor is now supported in these Regions. When OSI is selected, type in the minimum and maximum number of OCU. See more information here.
14. Choose **Next**.
15. Add tags if needed.
16. Choose **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - ELB Log Ingestion* solution in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Regions		Template
AWS China Regions		Template

1. Log in to the AWS Management Console and select the preceding button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.
2. To launch the stack 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.

Parameter	Default	Description
Log Bucket Name	`<Requires input>`	The S3 bucket name that stores the logs.
Log Bucket Prefix	`<Requires input>`	The S3 bucket path prefix that stores the logs.
Log Source Account ID	*Optional input*	The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at **Step 1** will be used.
Log Source Region	*Optional input*	The AWS Region of the S3 bucket. By default, the Region you selected at **Step 2** will be used.
Log Source Account Assume Role	*Optional input*	The IAM Role ARN used for cross-account log ingestion . Required for cross-account log ingestion (add a member account first).
KMS-CMK ARN	*Optional input*	The KMS-CMK ARN for encryption. Leave it blank to create a new AWS KMS key.
Enable OpenSearch Ingestion as processor	*Optional input*	Ingestion table ARN. Leave empty if you do not use OSI as Processor.

Parameter	Default	Description
S3 Backup Bucket	*<Requires input>*	The S3 backup bucket name to store the failed ingestion logs.
Engine Type	OpenSearch	The engine type of the OpenSearch.
OpenSearch Domain Name	*<Requires input>*	The domain name of the Amazon OpenSearch Service cluster.
OpenSearch Endpoint	*<Requires input>*	The OpenSearch endpoint URL. For example, vpc-your_opensearch_domain_name-xcvgw6uu2o6zafsiefxubwuohe.us-east-1.es.amazonaws.com
Index Prefix	*<Requires input>*	The common prefix of OpenSearch index for the log. The index name will be <Index Prefix>-<Log Type>-<Other Suffix>.
Create Sample Dashboard	Yes	Whether to create a sample OpenSearch dashboard.
VPC ID	*<Requires input>*	Select a VPC that has access to the OpenSearch domain. The log processing Lambda will reside in the selected VPC.

Parameter	Default	Description
Subnet IDs	*<Requires input>*	Select at least two subnets that have access to the OpenSearch domain. The log processing Lambda will reside in the subnets. Make sure that the subnets have access to the Amazon S3 service.
Security Group ID	*<Requires input>*	Select a Security Group that will be associated with the log processing Lambda. Make sure that the Security Group has access to the OpenSearch domain.
Number Of Shards	5	Number of shards to distribute the index evenly across all data nodes. Keep the size of each shard between 10-50 GB.
Number of Replicas	1	Number of replicas for OpenSearch Index. Each replica is a full copy of an index. If the OpenSearch option is set to Domain with standby, you need to configure it to 2.

Parameter	Default	Description
Age to Warm Storage	*Optional input*	The age required to move the index into warm storage (for example, 7d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when warm storage is enabled in OpenSearch.
Age to Cold Storage	*Optional input*	The age required to move the index into cold storage (for example, 30d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when cold storage is enabled in OpenSearch.
Age to Retain	*Optional input*	The age to retain the index (for example, 180d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). If the value is "", the index will not be deleted.
Rollover Index Size	*Optional input*	The minimum size of the shard storage required to roll over the index (for example, 30GB).

Parameter	Default	Description
Index Suffix	`yyyy-MM-dd`	The common suffix format of OpenSearch index for the log (for example, yyyy-MM-dd, yyyy-MM-dd-HH). The index name will be <Index Prefix>-<Log Type>-<Index Suffix>-000001.
Compression type	`best_compression`	The compression type to use to compress stored data. Available values are best_compression and default.
Refresh Interval	`1s`	How often the index should refresh, which publishes its most recent changes and makes them available for searching. Can be set to -1 to disable refreshing. Default is 1s.
Plugins	*Optional input*	List of plugins delimited by comma. Leave it blank if there are no available plugins to use. Valid inputs are user_agent, geo_ip.
EnableS3Notification	`True`	An option to enable or disable notifications for Amazon S3 buckets. The default option is recommended for most cases.

Parameter	Default	Description
LogProcessorRoleName	*Optional input*	Specify a role name for the log processor. The name should NOT duplicate an existing role name. If no name is specified, a random name is generated.
QueueName	*Optional input*	Specify a queue name for an Amazon SQS queue. The name should NOT duplicate an existing queue name. If no name is given, a random name will be generated.

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

View dashboard

The dashboard includes the following visualizations.

Visualization Name	Source Field	Description
Total Requests	* log event	Displays aggregated events based on a specified time interval.

Visualization Name	Source Field	Description
Request History	* log event	Presents a bar chart that displays the distribution of events over time.
Request By Target	* log event * target_ip	Presents a bar chart that displays the distribution of events over time and IP.
Unique Visitors	* client_ip	Displays unique visitors identified by client IP address.
Status Code	* elb_status_code	Displays the count of requests made to the Application Load Balancer, grouped by HTTP status codes (for example, 200, 404, 403).
Status History	* elb_status_code	Shows the historical trend of HTTP status codes returned by the Application Load Balancer over a specific period of time.
Status Code Pipe	* elb_status_code	Represents the distribution of requests based on different HTTP status codes using a pie chart.
Average Processing Time	* request_processing_time * response_processing_time * target_processing_time	This visualization calculate s and presents the average time taken for various operations in the Application Load Balancer.

Visualization Name	Source Field	Description
Avg. Processing Time History	* request_processing_time * response_processing_time * target_processing_time	Displays the historical trend of the average time-cons uming of each operation returned by the Applicati on Load Balancer within a specific period of time.
Request Verb	* request_verb	Displays the count of requests made to the Application Load Balancer using a pie chart, grouped by HTTP request method names (for example, POST, GET, HEAD).
Total Bytes	* received_bytes * sent_bytes	Provides insights into data transfer activities, including the total bytes transferred.
Sent and Received Bytes History	* received_bytes * sent_bytes	Displays the historical trend of the received bytes, send bytes
SSL Protocol	* ssl_protocol	Displays the count of requests made to the Application Load Balancer, grouped by SSL Protocol
Top Request URLs	* request_url	The web requests view enables you to analyze the top web requests.
Top Client IPs	* client_ip	Provides the top 10 IP address accessing your Application Load Balancer.

Visualization Name	Source Field	Description
Top User Agents	* user_agent	Provides the top 10 user agents accessing your Application Load Balancer.
Target Status	* target_ip * target_st atus_code	Displays the HTTP status code request count for targets in the Application Load Balancer target group.
Abnormal Requests	* @timestamp * client_ip * target_ip * elb_status_code * error_reason * request_verb * target_status_code * target_st atus_code_list * request_url * request_proto * trace_id	Provides a detailed list of log events, including timestamps, client ip, and target ip.
Requests by OS	* ua_os	Displays the count of requests made to the Application Load Balancer, grouped by user agent OS
Request by Device	* ua_device	Displays the count of requests made to the Application Load Balancer, grouped by user agent device.
Request by Browser	* ua_browser	Displays the count of requests made to the Application Load Balancer, grouped by user agent browser.

Visualization Name	Source Field	Description
Request by Category	* ua_category	Displays the count of categories made to the Application Load Balancer, grouped by user agent category (for example, PC, Mobile, Tablet).
Requests by Countries or Regions	* geo_iso_code	Displays the count of requests made to the Application Load Balancer (grouped by the corresponding country or Region resolved by the client IP).
Top Countries or Regions	* geo_country	Top 10 countries with the Application Load Balancer Access.
Top Cities	* geo_city	Top 10 cities with Application Load Balancer Access

You can access the built-in dashboard in Amazon OpenSearch Service to view log data. For more information, see the Access Dashboard.

Application Load Balancer logs sample dashboard.


Total Requests
296,314
Total Requests
Request History
Count
@timestamp per 10 seconds
Request By Target
Count
target_ip: Descending
10.255.2.27
74,095
10.255.2.205
74,085
10.255.3.173
74,077
10.255.3.110
74,057
Unique Vistors
21
Unique Vistors
Status Code
84,742
3xx Count
1
4xx Count
0
5xx Count
Status Code History
200
302
404
@timestamp per 10 seconds
Status Code Pie
404 (0%)
302 (28.6%)
200 (71.4%)
Average Processing Time
0
Avg. Request Processing Time
0
Avg. Response Processing Time
1.739
Avg. Target Processing Time
Avg. Processing Time History
Avg. Request Proce...
Avg. Response Proc...
Avg. Target Process...
Request Time (s)
@timestamp per 10 seconds
Request Verb
GET
POST
POST (0%)
GET (100%)
Total Bytes
4.9GB
Total Sent Bytes
43.6MB
Total Received Bytes
Sent and Received Bytes History
Received Bytes 315.2KB
Sent Bytes 35.3MB
per 5 seconds
SSL Protocol
TLSv1.2
TLSv1.2 (100%)
Top Request URLs
Request URI
Count
https://nginx.awssolution.xyz:443/ 42,376
https://nginx.awssolution.xyz:443/_dashboards 42,366
https://nginx.awssolution.xyz:443/_dashboards/app/login?nextUrl=%2F_dashboards%2F 42,341
https://nginx.awssolution.xyz:443/_dashboards/ui/favicons/favicon-32x32.png 42,314
https://nginx.awssolution.xyz:443/_dashboards/bootstrap.js 42,308
https://nginx.awssolution.xyz:443/_dashboards/ui/favicons/favicon.ico 42,305
https://nginx.awssolution.xyz:443/_dashboards/ui/favicons/favicon-16x16.png 42,303
https://63.35.133.122:443/vendor/phpunit/phpunit/src/Util/PHP/eval-stdin.php 1
Export: Raw Formatted
Top Client IPs
Client IP
Count
54.229.97.55 14,988
52.214.34.112 14,925
52.209.96.102 14,906
34.244.230.211 14,902
52.211.0.83 14,884
34.255.194.0 14,871
52.18.32.201 14,864
54.229.53.80 14,824
34.247.38.70 14,818
54.74.39.12 14,812
Export: Raw Formatted
Top User Agents
User Agent
Count
Apache-HttpClient/4.5.10 (Java/1.8.0_292) 84,742
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.108 Safari/537.36 1
Export: Raw Formatted
Target Status
200
302
404
21,389
21,038
21,333
21,182
52,906
53,046
52,744
52,875
Count
Target IP
Abnormal Requests
1–50 of 84743
Time
client_ip
target_ip
elb_status_code
error_reason
request_verb
target_status_code
target_status_code_list
request_url
request_proto
trace_id
Nov 22, 2021 @ 17:19:59.996 34.255.194.0 10.255.3.173 302 - GET 302 302 https://nginx.awssolution.xyz:443/_dashboards HTTP/1.1 Root=1-619b60be-391884ff2e726c5b155b1675
Nov 22, 2021 @ 17:19:59.963 54.216.184.75 10.255.3.110 302 - GET 302 302 https://nginx.awssolution.xyz:443/ HTTP/1.1 Root=1-619b60bf-6135e358240f83282ef455bb
Nov 22, 2021 @ 17:19:59.930 52.214.34.112 10.255.2.27 302 - GET 302 302 https://nginx.awssolution.xyz:443/ HTTP/1.1 Root=1-619b60bf-0bc31b934cbf1e28232cdba7
Nov 22, 2021 @ 17:19:59.908 52.18.32.201 10.255.2.205 302 - GET 302 302 https://nginx.awssolution.xyz:443/ HTTP/1.1 Root=1-619b60bf-528b5918f1080a090676e4c3
Nov 22, 2021 @ 17:19:59.908 52.209.96.102 10.255.2.27 302 - GET 302 302 https://nginx.awssolution.xyz:443/ HTTP/1.1 Root=1-619b60bf-3783b8fd43b2dd242bee2a64
Nov 22, 2021 @ 17:19:59.906 46.137.143.120 10.255.3.110 302 - GET 302 302 https://nginx.awssolution.xyz:443/ HTTP/1.1 Root=1-619b60bf-0e2a9fc035e3ff02050a8af7
Nov 22, 2021 @ 17:19:59.904 18.203.162.158 10.255.2.27 302 - GET 302 302 https://nginx.awssolution.xyz:443/ HTTP/1.1 Root=1-619b60bf-795ce0e22f17f1496c92fbc3
Nov 22, 2021 @ 17:19:59.904 52.211.0.83 10.255.2.205 302 - GET 302 302 https://nginx.awssolution.xyz:443/ HTTP/1.1 Root=1-619b60bf-2c66e60e280112ab7b150199
Nov 22, 2021 @ 17:19:59.904 54.154.108.42 10.255.2.27 302 - GET 302 302 https://nginx.awssolution.xyz:443/ HTTP/1.1 Root=1-619b60bf-1bc665720e6483ff23f32bf6
Nov 22, 2021 @ 17:19:59.891 46.137.143.120 10.255.3.173 302 - GET 302 302 https://nginx.awssolution.xyz:443/ HTTP/1.1 Root=1-619b60bf-3c6a2755406e5e545e21591c

Create log ingestion (Light Engine)

Using the Centralized Logging with OpenSearch Console

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose the Create a log ingestion button.
4. In the AWS Services section, choose **Elastic Load Balancer**.
5. Choose **Next**.
6. Under **Specify settings**, choose **Automatic** or **Manual** for **CloudFront logs enabling**. The automatic mode will detect the CloudFront log location automatically.
   - For **Automatic** mode, choose an Application Load Balancer in the dropdown list. (If the selected Application Load Balancer access log is not enabled, choose **Enable** to enable the Application Load Balancer access log.)
   - For Manual mode, enter the Application Load Balancer identifier and Log location.
   - (Optional) If you are ingesting CloudFront logs from another account, select a linked account from the **Account** dropdown list first.
7. Choose **Next**.
8. Choose **Log Processing Enriched fields** if needed. The available plugins are **location** and **OS/User Agent**. Enabling rich fields increases data processing latency and processing costs. By default, it is not selected.
9. In the **Specify Light Engine Configuration** section, if you want to ingest associated templated Grafana dashboards, select **Yes** for the sample dashboard.
10. You can choose an existing Grafana, or if you must import a new one, you can go to Grafana for configuration.
11. Select an S3 bucket to store partitioned logs and define a name for the log table. We have provided a predefined table name, but you can modify it according to your business needs.
12. If needed, change the log processing frequency, which is set to **5** minutes by default, with a minimum processing frequency of **1** minute.
13. In the **Log Lifecycle** section, enter the log merge time and log archive time. We have provided default values, but you can adjust them based on your business requirements.
14. Select **Next**.
15. If desired, add tags.

16Select **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - ELB Log Ingestion* solution in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Regions		Template
AWS China Regions		Template

1. Log in to the AWS Management Console and select the preceding button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.
2. To launch the stack 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.
   a. Parameters for **Pipeline settings**

Parameter	Default	Description
Pipeline Id	*<Requires input>*	The unique identifier for the pipeline is essential if you must create multiple

Parameter	Default	Description
		Application Load Balancer pipelines and write different Application Load Balancer logs into separate tables. For uniqueness, you can generate a unique pipeline identifier using uuidgenerator.
Staging Bucket Prefix	`AWSLogs/ALBLogs`	The storage directory for logs in the temporary storage area should ensure the uniqueness and non-overlapping of the Prefix for different pipelines.

b. Parameters for **Destination settings**

Parameter	Default	Description
Centralized Bucket Name	*`<Requires input>`*	Centralized S3 bucket name. For example, centralized-logging-bucket.
Centralized Bucket Prefix	`datalake`	Centralized bucket prefix. By default, the data base location is s3://{Centralized Bucket Name}/{Centralized Bucket Prefix}/amazon_cl_centralized.
Centralized Table Name	`ALB`	Table name for writing data to the centralized database. You can modify it if needed.

c. Parameters for **Scheduler settings**

Parameter	Default	Description
LogProcessor Schedule Expression	`rate(5 minutes)`	Task scheduling expression for performing log processing, with a default value of executing the LogProcessor every 5 minutes. Configuration for reference.
LogMerger Schedule Expression	`cron(0 1 * ? )`	Task scheduling expression for performing log merging, with a default value of executing the LogMerger at 1 AM every day. Configuration for reference.
LogArchive Schedule Expression	`cron(0 2 * ? )`	Task scheduling expression for performing log archiving, with a default value of executing the LogArchive at 2 AM every day. Configuration for reference.
Age to Merge	7	Small file retention days, with a default value of 7, indicating that logs older than 7 days will be merged into small files. It can be adjusted as needed.

Parameter	Default	Description
Age to Archive	30	Log retention days, with a default value of 30, indicating that data older than 30 days will be archived and deleted. It can be adjusted as needed.

d. Parameters for **Notification settings**

Parameter	Default	Description
Notification Service	SNS	Notification method for alerts. If your main stack is using China, you can only choose the SNS method. If your main stack is using Global, you can choose either the SNS or SES method.

Parameter	Default	Description
Recipients	*<Requires input>*	Alert notification: If the Notification Service is SNS, enter the SNS Topic ARN here so that you have the necessary permissions. If the Notification Service is SES, enter the email addresses separated by commas here, ensuring that the email addresses are already Verified Identitie s in SES. The adminEmai l provided during the creation of the main stack will receive a verification email by default.

e. Parameters for **Dashboard settings**

Parameter	Default	Description
Import Dashboards	FALSE	Whether to import the Dashboard into Grafana, with a default value of false. If set to true, you must provide the Grafana URL and Grafana Service Account Token.
Grafana URL	*<Requires input>*	Grafana access URL for example: https:// alb-72277319.us-wes t-2.elb.amazonaws.com.

Parameter	Default	Description
Grafana Service Account Token	*<Requires input>*	Grafana Service Account Token Service Account Token created in Grafana.

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

View dashboard

The dashboard includes the following visualizations.

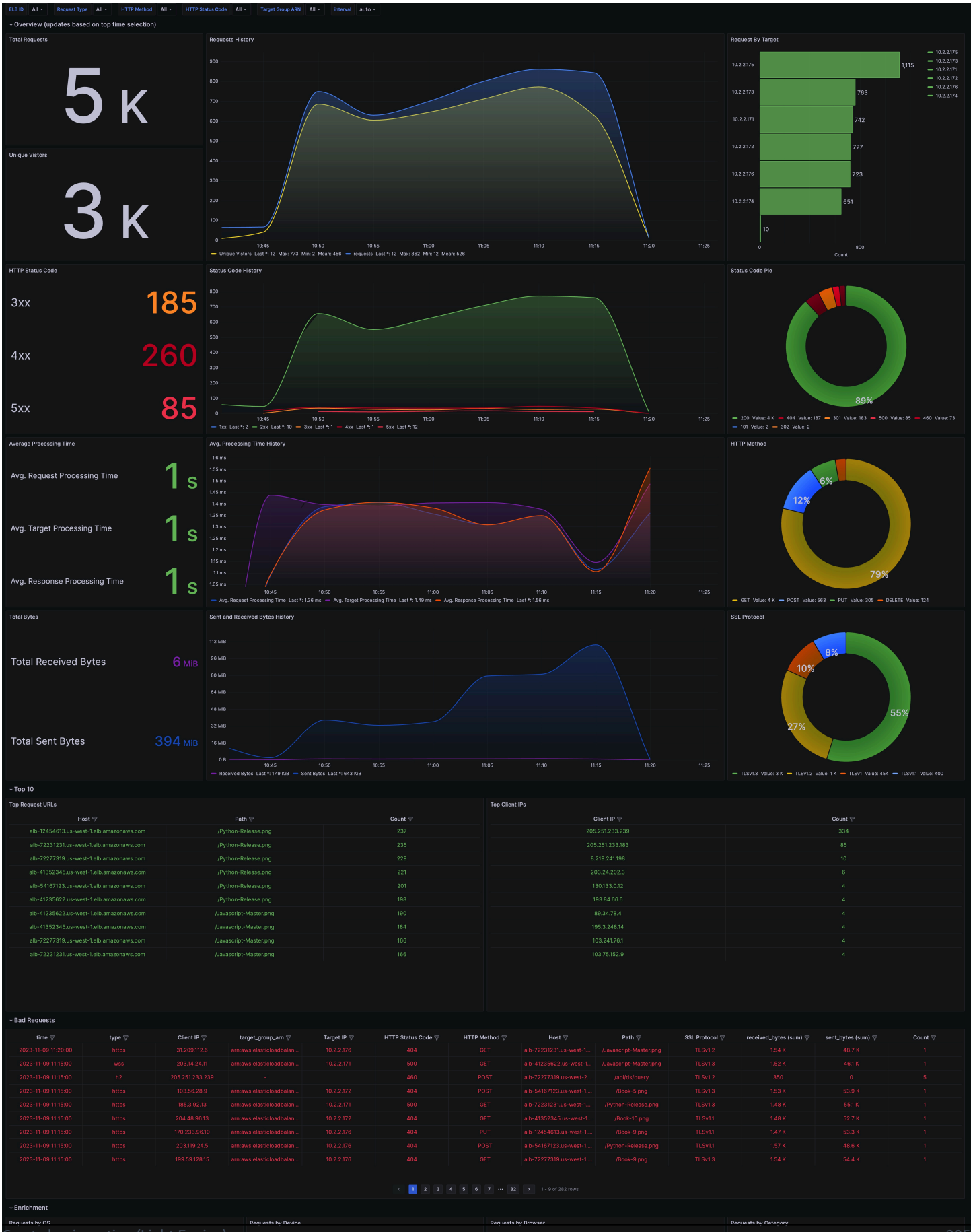
Visualization Name	Source Field	Description
Filters	Filters	The following data can be filtered by query filter conditions.
Total Requests	log event	Displays aggregated events based on a specified time interval.
Unique Visitors	client_ip	Displays unique visitors identified by client IP address.
Requests History	log event	Presents a bar chart that displays the distribution of events over time.

Visualization Name	Source Field	Description
Request By Target	log event target_ip	Presents a bar chart that displays the distribution of events over time and IP.
HTTP Status Code	elb_status_code	Displays the count of requests made to the Application Load Balancer, grouped by HTTP status codes (for example, 200, 404, 403).
Status Code History	elb_status_code	Shows the historical trend of HTTP status codes returned by the Application Load Balancer over a specific period of time.
Status Code Pie	elb_status_code	Represents the distribution of requests based on different HTTP status codes using a pie chart.
Average Processing Time	request_processing_time response_processing_time target_processing_time	This visualization calculate s and presents the average time taken for various operations in the Application Load Balancer.
Avg. Processing Time History	request_processing_time response_processing_time target_processing_time	Displays the historical trend of the average time-cons uming of each operation returned by the Applicati on Load Balancer within a specific period of time.

Visualization Name	Source Field	Description
HTTP Method	request_verb	Displays the count of requests made to the Application Load Balancer using a pie chart, grouped by HTTP request method names (for example, POST, GET, HEAD).
Total Bytes	received_bytes sent_bytes	Provides insights into data transfer activities, including the total bytes transferred.
Sent and Received Bytes History	received_bytes sent_bytes	Displays the historical trend of the received bytes, send bytes.
SSL Protocol	ssl_protocol	Displays the count of requests made to the Application Load Balancer, grouped by SSL Protocol.
Top Request URLs	request_url	The web requests view enables you to analyze the top web requests.
Top Client IPs	client_ip	Provides the top 10 IP addresses accessing your Application Load Balancer.
Bad Requests	type client_ip target_gr oup_arn target_ip elb_statu s_code request_verb request_url ssl_protocol received_bytes sent_bytes	Provides a detailed list of log events, including timestamps, client IP, and target IP.

Visualization Name	Source Field	Description
Requests by OS	ua_os	Displays the count of requests made to the Application Load Balancer, grouped by user agent OS.
Requests by Device	ua_device	Displays the count of requests made to the Application Load Balancer, grouped by user agent device.
Requests by Browser	ua_browser	Displays the count of requests made to the Application Load Balancer, grouped by user agent browser.
Requests by Category	ua_category	Displays the count of categories made to the Application Load Balancer, grouped by user agent category (for example, PC, Mobile, Tablet).
Requests by Countries or Regions	geo_iso_code	Displays the count of requests made to the Application Load Balancer (grouped by the corresponding country or Region resolved by the client IP).
Top Countries or Regions	geo_country	Top 10 countries with the Application Load Balancer Access.

Application Load Balancer logs sample dashboard.



Top Request URLs

Host	Path	Count
alb-12454613.us-west-1.elb.amazonaws.com	/Python-Release.png	237
alb-72231231.us-west-1.elb.amazonaws.com	/Python-Release.png	235
alb-72277319.us-west-1.elb.amazonaws.com	/Python-Release.png	229
alb-41352345.us-west-1.elb.amazonaws.com	/Python-Release.png	221
alb-54167123.us-west-1.elb.amazonaws.com	/Python-Release.png	201
alb-41235622.us-west-1.elb.amazonaws.com	/Python-Release.png	198
alb-41235622.us-west-1.elb.amazonaws.com	/Javascript-Master.png	190
alb-41352345.us-west-1.elb.amazonaws.com	/Javascript-Master.png	184
alb-72277319.us-west-1.elb.amazonaws.com	/Javascript-Master.png	166
alb-72231231.us-west-1.elb.amazonaws.com	/Javascript-Master.png	166

Top Client IPs

Client IP	Count
205.251.233.239	334
205.251.233.183	85
8.219.241.198	10
203.24.202.3	6
130.133.0.12	4
193.84.66.6	4
89.34.78.4	4
195.3.248.14	4
103.241.76.1	4
103.75.152.9	4

Bad Requests

time	type	Client IP	target_group_arn	Target IP	HTTP Status Code	HTTP Method	Host	Path	SSL Protocol	received_bytes (sum)	sent_bytes (sum)	Count
2023-11-09 11:20:00	https	31.209.112.6	arn:aws:elasticloadbalan...	10.2.2.176	404	GET	alb-72231231.us-west-1....	/Javascript-Master.png	TLSv1.2	1.54 K	48.7 K	1
2023-11-09 11:15:00	wss	203.14.24.11	arn:aws:elasticloadbalan...	10.2.2.171	500	GET	alb-41235622.us-west-1....	/Javascript-Master.png	TLSv1.3	1.52 K	46.1 K	1
2023-11-09 11:15:00	h2	205.251.233.239	-		460	POST	alb-72277319.us-west-2....	/api/ds/query	TLSv1.2	350	0	5
2023-11-09 11:15:00	https	103.56.28.9	arn:aws:elasticloadbalan...	10.2.2.172	404	POST	alb-54167123.us-west-1....	/Book-5.png	TLSv1.3	1.53 K	53.9 K	1
2023-11-09 11:15:00	https	185.3.92.13	arn:aws:elasticloadbalan...	10.2.2.171	500	GET	alb-72231231.us-west-1....	/Python-Release.png	TLSv1.3	1.48 K	55.1 K	1
2023-11-09 11:15:00	https	204.48.96.13	arn:aws:elasticloadbalan...	10.2.2.172	404	GET	alb-41352345.us-west-1....	/Book-10.png	TLSv1.1	1.48 K	52.7 K	1
2023-11-09 11:15:00	https	170.233.96.10	arn:aws:elasticloadbalan...	10.2.2.176	404	PUT	alb-12454613.us-west-1....	/Book-9.png	TLSv1.1	1.47 K	53.3 K	1
2023-11-09 11:15:00	https	203.119.24.5	arn:aws:elasticloadbalan...	10.2.2.176	404	POST	alb-54167123.us-west-1....	/Python-Release.png	TLSv1.1	1.57 K	48.6 K	1
2023-11-09 11:15:00	https	199.59.128.15	arn:aws:elasticloadbalan...	10.2.2.176	404	GET	alb-72277319.us-west-1....	/Book-9.png	TLSv1.3	1.54 K	54.4 K	1

AWS WAF logs

AWS WAF Access Logs provide detailed information about traffic that is analyzed by your web ACL. Logged information includes the time that AWS WAF received a web request from your AWS resource, detailed information about the request, and details about the rules that the request matched.

You can create a log ingestion into Amazon OpenSearch Service either by using the Centralized Logging with OpenSearch console or by deploying a standalone CloudFormation stack.

Important

- Deploy Centralized Logging with OpenSearch solution in the same Region as your Web ACLs, or you will not be able to create a AWS WAF pipeline. For example:
  - If your Web ACL is associated with Global CloudFront, you must deploy the solution in us-east-1.
  - If your Web ACL is associated with other resources in Regions like Ohio, your Centralized Logging with OpenSearch stack must also be deployed in that Region.
- The AWS WAF logging bucket must be the same as the Centralized Logging with OpenSearch solution.
- AWS WAF Classic logs are not supported in Centralized Logging with OpenSearch. Learn more about migrating rules from AWS WAF Classic to the new AWS WAF.
- The Amazon OpenSearch Service index is rotated on a daily basis by default, and you can adjust the index in the Additional Settings.

Create log ingestion (OpenSearch Engine)

Using the Centralized Logging with OpenSearch Console

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose the Create a log ingestion button.
4. In the **AWS Services** section, choose **AWS WAF**.
5. Choose **Next**.

6. Under Specify settings, choose Automatic or Manual.
   - For **Automatic** mode, choose a Web ACL in the dropdown list.
   - For **Manual** mode, enter the **Web ACL name**.
   - (Optional) If you are ingesting AWS WAF logs from another account, select a linked account from the **Account** dropdown list first.
7. Specify an Ingest Options. Choose between Sampled Request or Full Request.
   - For **Sampled Request**, enter how often you want to ingest sample requests in minutes.
   - For **Full Request**, if the Web ACL log is not enabled, choose **Enable** to enable the access log, or enter **Log location** in Manual mode. Note that Centralized Logging with OpenSearch will automatically enable logging with a Firehose stream as destination for your AWS WAF.
8. Choose **Next**.
9. In the Specify OpenSearch domain section, select an imported domain for the Amazon OpenSearch Service domain.
10. Choose **Yes** for **Sample dashboard** if you want to ingest an associated templated Amazon OpenSearch Service dashboard.
11. You can change the **Index Prefix** of the target Amazon OpenSearch Service index if needed. The default prefix is the Web ACL Name.
12. In the **Log Lifecycle** section, enter the number of days to manage the Amazon OpenSearch Service index lifecycle. The Centralized Logging with OpenSearch will create the associated Index State Management (ISM) policy automatically for this pipeline.
13. In the **Select log processor** section, choose the log processor.
    a. When selecting Lambda as a log processor, you can configure the Lambda concurrency if needed.
    b. (Optional) OSI as log processor is now supported in these Regions. When OSI is selected, type in the minimum and maximum number of OCU. See more information here.
14. Choose **Next**.
15. Add tags if needed.
16. Choose **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - AWS WAF Log Ingestion* solution in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Regions (Full Request)		Template
AWS China Regions (Full Request)		Template
AWS Regions (Sampled Request)		Template
AWS China Regions (Sampled Request)		Template

1. Log in to the AWS Management Console and select the button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.
2. To launch the stack 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.
   - Parameters for **Full Request** only

Parameter	Default	Description
Log Bucket Name	`<Requires input>`	The S3 bucket name that stores the logs.

Parameter	Default	Description
Log Bucket Prefix	*<Requires input>*	The S3 bucket path prefix that stores the logs.

- Parameters for **Sampled Request** only

Parameter	Default	Description
WebACL Names	*<Requires input>*	The list of Web ACL names, delimited by comma.
Interval	2	The default interval (in minutes) to get sampled logs. The value must be greater or equal to 2, and less or equal to 180.

- Common parameters

Parameter	**Default**	**Description**	**Log Source Account ID**	***Optional input***	**The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at Step 1 will be used.**
Log Source Region	*Optional input*	The AWS Region of the S3 bucket. By default, the Region you selected at **Step 2** will be used.	Log Source Account Assume Role	*Optional input*	The IAM Role ARN used for cross-account log ingestion. Required for cross-account log ingestion (add a member account first).

Parameter	Default	Description	Log Source Account ID	*Optional input*	**The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at Step 1 will be used.**
KMS-CMK ARN	*Optional input*	The KMS-CMK ARN for encryption. Leave it blank to create a new AWS KMS key.	Enable OpenSearch Ingestion as processor	*Optional input*	Ingestion table ARN. Leave empty if you do not use OSI as Processor.

Parameter	Default	Description	Log Source Account ID	*Optional input*	The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at Step 1 will be used.
S3 Backup Bucket	*<Requires input>*	The S3 backup bucket name to store the failed ingestion logs.	Engine Type	`OpenSearch`	The engine type of the OpenSearch. Select OpenSearch or OpenSearch.

Parameter	Default	Description	Log Source Account ID	*Optional input*	**The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at Step 1 will be used.**
OpenSearch Domain Name	*<Requires input>*	The domain name of the Amazon OpenSearch Service cluster.	OpenSearch Endpoint	*<Requires input>*	The OpenSearch endpoint URL. For example, vpc-your_opensearch_domain_name-xcvgw6uu2o6afsiefxubwuohe.us-east-1.es.amazonaws.com

Parameter	Default	Description	Log Source Account ID	*Optional input*	**The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at Step 1 will be used.**
Index Prefix	*<Requires input>*	The common prefix of OpenSearch index for the log. The index name will be <Index Prefix>-<log-type>-<YYYY-MM-DD>.	Create Sample Dashboard	Yes	Whether to create a sample OpenSearch dashboard.

Parameter	Default	Description	Log Source Account ID	*Optional input*	**The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at Step 1 will be used.**
VPC ID	*<Requires input>*	Select a VPC that has access to the OpenSearch domain. The log processing Lambda will reside in the selected VPC.	Subnet IDs	*<Requires input>*	Select at least two subnets that have access to the OpenSearch domain. The log processing Lambda will reside in the subnets. Make sure that the subnets have access to the

Parameter	**Default**	**Description**	**Log Source Account ID**	***Optional input***	**The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at Step 1 will be used.**
					Amazon S3 service.
Security Group ID	*<Requires input>*	Select a Security Group that will be associated with the log processing Lambda. Make sure that the Security Group has access to the OpenSearch domain.	Number Of Shards	5	Number of shards to distribute the index evenly across all data nodes. Keep the size of each shard between 10-50 GB.

Parameter	Default	Description	Log Source Account ID	*Optional input*	**The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at Step 1 will be used.**
Number of Replicas	1	Number of replicas for OpenSearch Index. Each replica is a full copy of an index. If the OpenSearch option is set to Domain with standby, you need to configure it to 2.	Age to Warm Storage	`0`	The number of days required to move the index into warm storage. This takes effect only when the value is larger than 0 and warm storage is enabled in OpenSearch.

Parameter	Default	Description	Log Source Account ID	*Optional input*	**The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at Step 1 will be used.**
Age to Cold Storage	0	The number of days required to move the index into cold storage. This takes effect only when the value is larger than 0 and cold storage is enabled in OpenSearch.	Age to Retain	0	The total number of days to retain the index. If the value is 0, the index will not be deleted.

Parameter	Default	Description	Log Source Account ID	*Optional input*	**The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at Step 1 will be used.**
Rollover Index Size	*Optional input*	The minimum size of the shard storage required to roll over the index (for example, 30GB).	Index Suffix	`yyyy-MM-dd`	The common suffix format of OpenSearch index for the log (for example, yyyy-MM-dd, yyyy-MM-dd-HH). The index name will be <Index Prefix>-<Log Type>-<Index

Parameter	Default	Description	Log Source Account ID	*Optional input*	The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at Step 1 will be used.
					Suffix>-000001.

Parameter	Default	Description	Log Source Account ID	*Optional input*	**The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at Step 1 will be used.**
Compression type	`best_compression`	The compression type to use to compress stored data. Available values are best_compression and default.	Refresh Interval	`1s`	How often the index should refresh, which publishes its most recent changes and makes them available for searching. Can be set to -1 to disable refreshing. Default is 1s.

Parameter	**Default**	**Description**	**Log Source Account ID**	***Optional input***	**The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at Step 1 will be used.**
Plugins	*Optional input*	List of plugins delimited by comma. Leave it blank if there are no available plugins to use. Valid inputs are user_agent, geo_ip.	EnableS3Notification	`True`	An option to enable or disable notifications for Amazon S3 buckets. The default option is recommended for most cases.

Parameter	Default	Description	Log Source Account ID	*Optional input*	**The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at Step 1 will be used.**
LogProcessorRoleName	*Optional input*	Specify a role name for the log processor. The name should NOT duplicate an existing role name. If no name is specified, a random name is generated.	QueueName	*Optional input*	Specify a queue name for an Amazon SQS queue. The name should NOT duplicate an existing queue name. If no name is given, a random name will be generated.

6. Choose **Next**.

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

View dashboard

The dashboard includes the following visualizations.

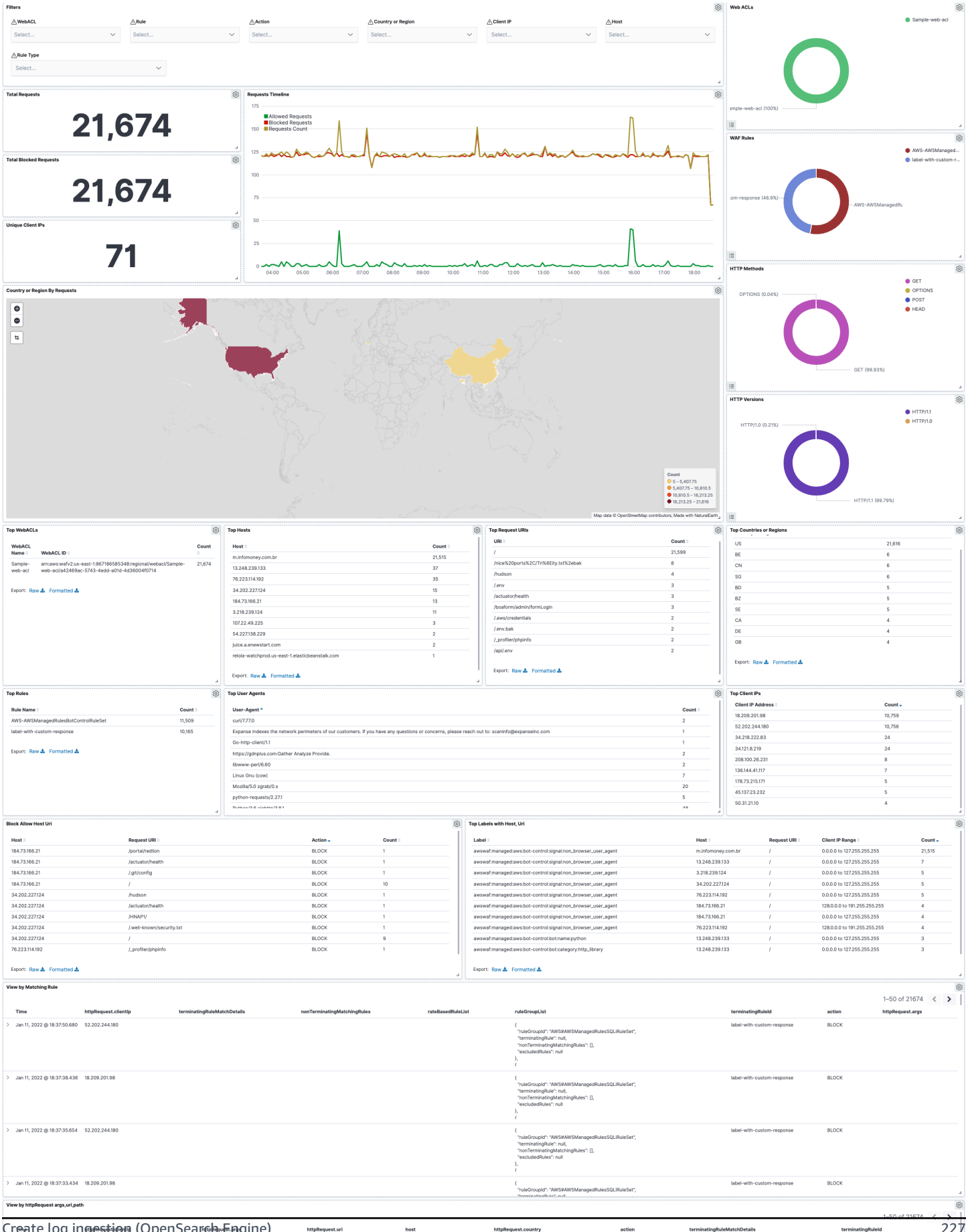
Visualization Name	Source Field	Description
Filters	* Filters	The following data can be filtered by query filter conditions.
Web ACLs	* log event * webaclName	Displays the count of requests made to the AWS WAF, grouped by Web ACL Names.
Total Requests	* log event	Displays the total number of web requests.
Request Timeline	* log event	Presents a bar chart that displays the distribution of events over time.
AWS WAF Rules	* terminatingRuleId	Presents a pie chart that displays the distribution of events over the AWS WAF rules in the Web ACL.
Total Blocked Requests	* log event	Displays the total number of blocked web requests.

Visualization Name	Source Field	Description
Unique Client IPs	* Request.ClientIP	Displays unique visitors identified by client IP.
Country or Region By Request	* Request.Country	Displays the count of requests made to the Web ACL (grouped by the correspon ding country or Region resolved by the client IP).
Http Methods	* Request.HTTPMethod	Displays the count of requests made to the Web ACL using a pie chart, grouped by HTTP request method names (for example, POST, GET, HEAD).
Http Versions	* Request.HTTPVersion	Displays the count of requests made to the Web ACL using a pie chart, grouped by HTTP protocol version (for example, HTTP/2.0, HTTP/1.1).
Top WebACLs	* webaclName * webaclId. keyword	The web requests view enables you to analyze the top web requests.
Top Hosts	* host	Lists the source IP addresses associated with events, enabling you to identify and investigate potentially suspicious or unauthorized activities.
Top Request URIs	* Request.URI	Top 10 request URIs.
Top Countries or Regions	* Request.country	Top 10 countries with the Web ACL Access.

Visualization Name	Source Field	Description
Top Rules	* terminatingRuleId	Top 10 rules in the web ACL that matched the request.
Top Client IPs	* Request.ClientIP	Provides the top 10 IP address.
Top User Agents	* userAgent	Provides the top 10 user agents
Block Allow Host Uri	* host * Request.URI * action	Provides blocked or allowed web requests.
Top Labels with Host, Uri	* labels.name * host * Request.URI	Top 10 detailed logs by labels with host, URI
View by Matching Rule	* sc-status	This visualization provides detailed logs by DQL "terminatingRuleId:*".
View by httpRequest args,uri, path	* sc-status	This visualization provides detailed logs by DQL.

You can access the built-in dashboard in Amazon OpenSearch Service to view log data. For more information, see the Access Dashboard.

AWS WAF logs sample dashboard.


Filters
WebACL
Rule
Action
Country or Region
Client IP
Host
Rule Type
Total Requests
21,674
Total Blocked Requests
21,674
Unique Client IPs
71
Requests Timeline
Allowed Requests
Blocked Requests
Requests Count
Web ACLs
Sample-web-acl
WAF Rules
AWS-AWSManaged...
label-with-custom-r...
HTTP Methods
GET
OPTIONS
POST
HEAD
OPTIONS (0.04%)
GET (99.93%)
HTTP Versions
HTTP/1.1
HTTP/1.0
HTTP/1.0 (0.21%)
HTTP/1.1 (99.79%)
Country or Region By Requests
Top WebACLs
Top Hosts
Top Request URIs
Top Countries or Regions
Top Rules
Top User Agents
Top Client IPs
Block Allow Host Uri
Top Labels with Host, Uri
View by Matching Rule
View by httpRequest args,uri,path

Create log ingestion (Light Engine)

Using the Centralized Logging with OpenSearch Console

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose the Create a log ingestion button.
4. In the **AWS Services** section, choose **AWS WAF**.
5. Choose **Light Engine**, choose **Next**.
6. Under Specify settings, choose Automatic or Manual.
   - For **Automatic** mode, choose a Web ACL from the dropdown list.
   - For **Manual** mode, enter the Web ACL name.
   - (Optional) If you are ingesting CloudFront logs from another account, select a linked account from the **Account** dropdown list first.
7. Choose **Next**.
8. Choose **Log Processing Enriched fields** if needed. The available plugins are **location** and **OS/User Agent**. Enabling rich fields increases data processing latency and processing costs. By default, it is not selected.
9. In the **Specify Light Engine Configuration** section, if you want to ingest associated templated Grafana dashboards, select **Yes** for the sample dashboard.
10. You can choose an existing Grafana, or if you must import a new one, you can go to Grafana for configuration.
11. Select an S3 bucket to store partitioned logs and define a name for the log table. We have provided a predefined table name, but you can modify it according to your business needs.
12. If needed, change the log processing frequency, which is set to **5** minutes by default, with a minimum processing frequency of **1** minute.
13. In the **Log Lifecycle** section, enter the log merge time and log archive time. We have provided default values, but you can adjust them based on your business requirements.
14. Select **Next**.
15. If desired, add tags.
16. Select **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - AWS WAF Log Ingestion* solution in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Regions		Template
AWS China Regions		Template

1. Log in to the AWS Management Console and select the preceding button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.
2. To launch the stack 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.
   a. Parameters for **Pipeline settings**

Parameter	Default	Description
Pipeline Id	`<Requires input>`	The unique identifier for the pipeline is essential if you must create multiple Application Load Balancer pipelines and write different Application Load Balancer

Parameter	Default	Description
		logs into separate tables. For uniqueness, you can generate a unique pipeline identifier using uuidgenerator.
Staging Bucket Prefix	`AWSLogs/WAFLogs`	The storage directory for logs in the temporary storage area should ensure the uniqueness and non-overlapping of the Prefix for different pipelines.

b. Parameters for **Destination settings**

Parameter	Default	Description
Centralized Bucket Name	*`<Requires input>`*	Centralized S3 bucket name. For example, centralized-logging-bucket.
Centralized Bucket Prefix	`datalake`	Centralized bucket prefix. By default, the data base location is s3://{Centralized Bucket Name}/{Centralized Bucket Prefix}/amazon_cl_centralized.
Centralized Table Name	`WAF`	Table name for writing data to the centralized database. You can modify it if needed.

c. Parameters for **Scheduler settings**

Parameter	Default	Description
LogProcessor Schedule Expression	`rate(5 minutes)`	Task scheduling expression for performing log processing, with a default value of executing the LogProcessor every 5 minutes. Configuration for reference.
LogMerger Schedule Expression	`cron(0 1 * _? )`	Task scheduling expression for performing log merging, with a default value of executing the LogMerger at 1 AM every day. Configuration for reference.
LogArchive Schedule Expression	`cron(0 2 * ? )`	Task scheduling expression for performing log archiving, with a default value of executing the LogArchive at 2 AM every day. Configuration for reference.
Age to Merge	7	Small file retention days, with a default value of 7, indicating that logs older than 7 days will be merged into small files. It can be adjusted as needed.

Parameter	Default	Description
Age to Archive	30	Log retention days, with a default value of 30, indicating that data older than 30 days will be archived and deleted. It can be adjusted as needed.

d. Parameters for **Notification settings**

Parameter	Default	Description
Notification Service	SNS	Notification method for alerts. If your main stack is using China, you can only choose the SNS method. If your main stack is using Global, you can choose either the SNS or SES method.

Parameter	Default	Description
Recipients	*<Requires input>*	Alert notification: If the Notification Service is SNS, enter the SNS Topic ARN here so that you have the necessary permissions. If the Notification Service is SES, enter the email addresses separated by commas here, ensuring that the email addresses are already Verified Identities in SES. The adminEmail provided during the creation of the main stack will receive a verification email by default.

e. Parameters for **Dashboard settings**

Parameter	Default	Description
Import Dashboards	FALSE	Whether to import the Dashboard into Grafana, with a default value of false. If set to true, you must provide the Grafana URL and Grafana Service Account Token.
Grafana URL	*<Requires input>*	Grafana access URL，for example: https://alb-72277319.us-west-2.elb.amazonaws.com.

Parameter	Default	Description
Grafana Service Account Token	*<Requires input>*	Grafana Service Account Token：Service Account Token created in Grafana.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review and create** page, review and confirm the settings. Check the box acknowledging that the template creates IAM resources.
9. Choose **Submit** 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.

View dashboard

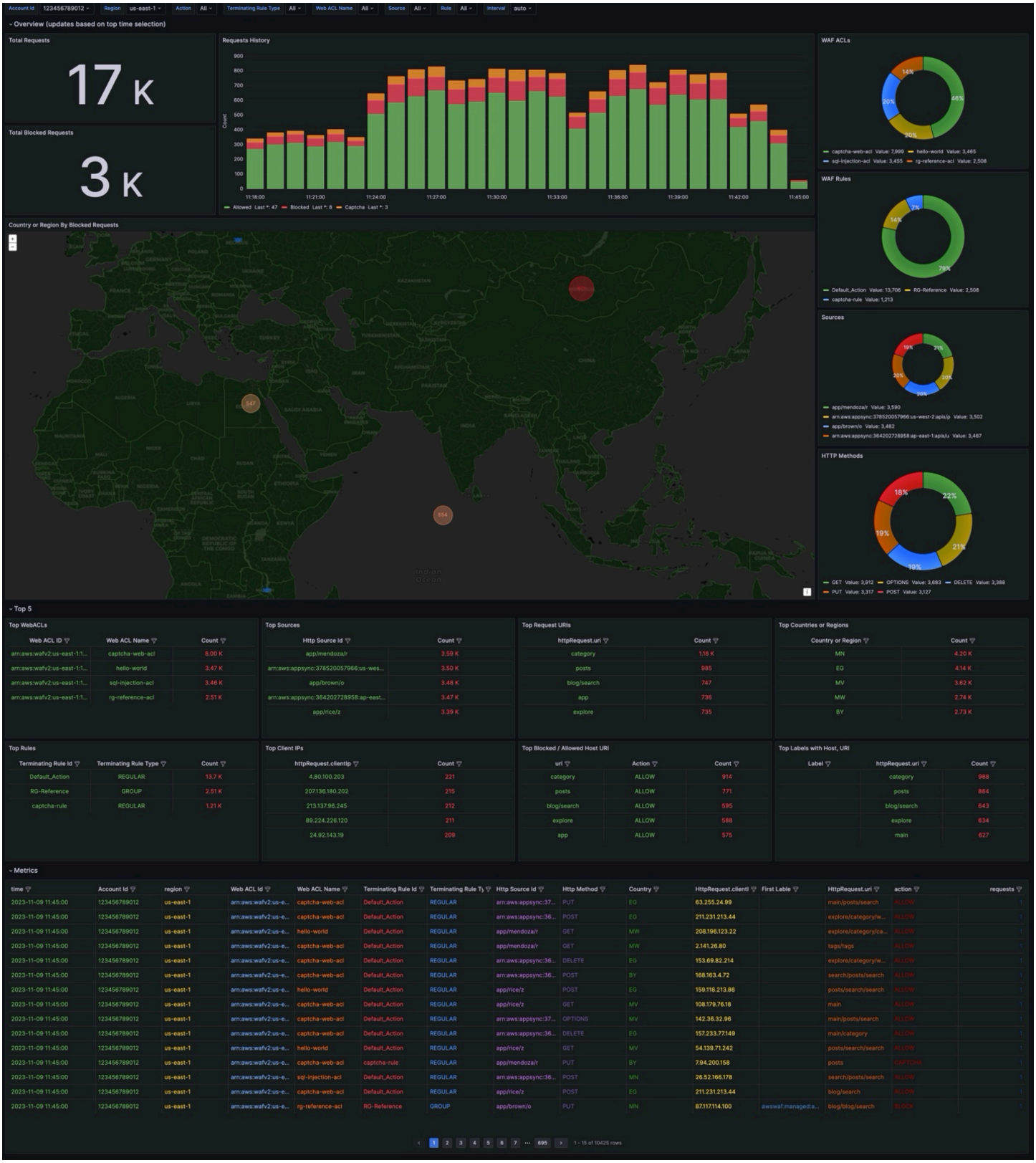
The dashboard includes the following visualizations.

Visualization Name	Source Field	Description
Filters	Filters	The following data can be filtered by query filter conditions.
Total Requests	log event	Displays the total number of web requests.
Total Blocked Requests	log event	Displays the total number of blocked web requests.
Requests History	log event	Presents a bar chart that displays the distribution of events over time.
AWS WAF ACLs	log event webaclName	Displays the count of requests made to the AWS WAF, grouped by Web ACL Names.

Visualization Name	Source Field	Description
AWS WAF Rules	terminatingRuleId	Presents a pie chart that displays the distribution of events over the AWS WAF rules in the Web ACL.
Sources	httpSourceId	Presents a pie chart that displays the distribution of events over the id of the associated resource.
HTTP Methods	httpRequest.HTTPMethod	Displays the count of requests made to the Web ACL using a pie chart, grouped by HTTP request method names (for example, POST, GET, HEAD).
Country or Region By Blocked Requests	HTTPRequest.Country	Displays the count of blocked web requests made to the Web ACL (grouped by the corresponding country or Region resolved by the client IP).
Top WebACLs	webaclName	The web requests view enables you to analyze the top web requests.
Top Sources	httpSourceId	Top 10 id of the associated resource.
Top Requests URIs	httpRequest.URI	Top 10 request URIs.
Top Countries or Regions	httpRequest.country	Top 10 countries with the Web ACL Access.
Top Rules	terminatingRuleId	Top 10 rules in the web ACL that matched the request.

Visualization Name	Source Field	Description
Top Client IPs	httpRequest.ClientIP	Provides the top 10 IP addresses.
Top Blocked / Allowed Hosts URI	host httpRequest.URI action	Provides blocked or allowed web requests.
Top Labels with Host, URI	labels host httpRequest.URI	Top 10 detailed logs by labels with host, URI.
Metrics	webaclId webaclName terminatingRuleId terminati ngRuleType httpSourc eId httpRequest.HTTPMe thod httpRequest.country httpRequest.ClientIP labels httpRequest.URI action	Provides a detailed list of log events, including timestamps, web ACL, and client IP.

AWS WAF logs sample dashboard.


Overview (updates based on top time selection)
Total Requests
17 K
Total Blocked Requests
3 K
Requests History
WAF ACLs
WAF Rules
Sources
HTTP Methods
Country or Region By Blocked Requests
Top 5
Top WebACLs
Top Sources
Top Request URIs
Top Countries or Regions
Top Rules
Top Client IPs
Top Blocked / Allowed Host URI
Top Labels with Host, URI
Metrics

VPC Flow Logs

VPC Flow Logs enable you to capture information about the IP traffic going to and from network interfaces in your VPC.

You can create a log ingestion into Amazon OpenSearch Service either by using the Centralized Logging with OpenSearch console or by deploying a standalone CloudFormation stack.

Important

- Centralized Logging with OpenSearch supports VPCs who publish the flow log data to an Amazon S3 bucket or a CloudWatch log group. When publishing to Amazon S3, the S3 bucket Region must be the same as the Centralized Logging with OpenSearch solution Region.
- The Amazon OpenSearch Service index is rotated on a daily basis by default, and you can adjust the index in the Additional Settings.

Create log ingestion (OpenSearch Engine)

Using the Centralized Logging with OpenSearch Console

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose the Create a log ingestion button.
4. In the **AWS Services** section, choose **VPC Flow Logs**.
5. Choose **Next**.
6. Under **Specify settings**, choose **Automatic** or **Manual** for **VPC Flow Log enabling**. The automatic mode will enable the VPC Flow Log and save the logs to a centralized S3 bucket if logging is not enabled yet.
   - For **Automatic mode**, choose the VPC from the dropdown list.
   - For Manual mode, enter the VPC Name and VPC Flow Logs location.
   - (Optional) If you are ingesting VPC Flow Logs from another account, select a linked account from the **Account** dropdown list first.
7. Under **Log Source**, select **S3** or **CloudWatch** as the source.

8. Choose **Next**.
9. In the Specify OpenSearch domain section, select an imported domain for the Amazon OpenSearch Service domain.
10. Choose **Yes** for **Sample dashboard** if you want to ingest an associated built-in Amazon OpenSearch Service dashboard.
11. You can change the **Index Prefix** of the target Amazon OpenSearch Service index if needed. The default prefix is your VPC name.
12. In the **Log Lifecycle** section, enter the number of days to manage the Amazon OpenSearch Service index lifecycle. The Centralized Logging with OpenSearch will create the associated Index State Management (ISM) policy automatically for this pipeline.
13. In the **Select log processor** section, choose the log processor.
    a. When selecting Lambda as a log processor, you can configure the Lambda concurrency if needed.
    b. (Optional) OSI as log processor is now supported in these Regions. When OSI is selected, please type in the minimum and maximum number of OCU. See more information here.
14. Choose **Next**.
15. Add tags if needed.
16. Choose **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - VPC Flow Logs Ingestion* solution in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Standard Regions		Template
AWS China Regions		Template

1. Log in to the AWS Management Console and select the preceding button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.
2. To launch the stack 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.

Parameter	Default	Description
Log Bucket Name	*<Requires input>*	The S3 bucket name that stores the logs.
Log Bucket Prefix	*<Requires input>*	The S3 bucket path prefix that stores the logs.
Log Source Account ID	*Optional input*	The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at **Step 1** will be used.
Log Source Region	*Optional input*	The AWS Region of the S3 bucket. By default, the Region you selected at **Step 2** will be used.
Log Source Account Assume Role	*Optional input*	The IAM Role ARN used for cross-account log ingestion . Required for cross-account

Parameter	Default	Description
		log ingestion (add a member account first).
KMS-CMK ARN	*Optional input*	The KMS-CMK ARN for encryption. Leave it blank to create a new AWS KMS key.
Enable OpenSearch Ingestion as processor	*Optional input*	Ingestion table ARN. Leave empty if you do not use OSI as Processor.
S3 Backup Bucket	*<Requires input>*	The S3 backup bucket name to store the failed ingestion logs.
Engine Type	OpenSearch	The engine type of the OpenSearch. Select OpenSearch or OpenSearch.
OpenSearch Domain Name	*<Requires input>*	The domain name of the Amazon OpenSearch Service cluster.
OpenSearch Endpoint	*<Requires input>*	The OpenSearch endpoint URL. For example, vpc-your_opensearch_domain_name-xcvgw6uu2o6zafsiefxubwuohe.us-east-1.es.amazonaws.com
Index Prefix	*<Requires input>*	The common prefix of OpenSearch index for the log. The index name will be <Index Prefix>-<Log Type>-<Other Suffix>.

Parameter	Default	Description
Create Sample Dashboard	Yes	Whether to create a sample OpenSearch dashboard.
VPC ID	*<Requires input>*	Select a VPC that has access to the OpenSearch domain. The log processing Lambda will reside in the selected VPC.
Subnet IDs	*<Requires input>*	Select at least two subnets that have access to the OpenSearch domain. The log processing Lambda will reside in the subnets. Make sure that the subnets have access to the Amazon S3 service.
Security Group ID	*<Requires input>*	Select a Security Group that will be associated with the log processing Lambda. Make sure that the Security Group has access to the OpenSearch domain.
Number Of Shards	5	Number of shards to distribute the index evenly across all data nodes. Keep the size of each shard between 10-50 GB.

Parameter	Default	Description
Number of Replicas	1	Number of replicas for OpenSearch Index. Each replica is a full copy of an index. If the OpenSearch option is set to Domain with standby, you need to configure it to 2.
Age to Warm Storage	*Optional input*	The age required to move the index into warm storage (for example, 7d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when warm storage is enabled in OpenSearch.
Age to Cold Storage	*Optional input*	The age required to move the index into cold storage (for example, 30d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when cold storage is enabled in OpenSearch.

Parameter	Default	Description
Age to Retain	*Optional input*	The age to retain the index (for example, 180d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). If the value is "", the index will not be deleted.
Rollover Index Size	*Optional input*	The minimum size of the shard storage required to roll over the index (for example, 30GB).
Index Suffix	`yyyy-MM-dd`	The common suffix format of OpenSearch index for the log (for example, yyyy-MM-dd, yyyy-MM-dd-HH). The index name will be <Index Prefix>-<Log Type>-<Index Suffix>-000001.
Compression type	`best_compression`	The compression type to use to compress stored data. Available values are best_compression and default.
Refresh Interval	`1s`	How often the index should refresh, which publishes its most recent changes and makes them available for searching. Can be set to -1 to disable refreshing. Default is 1s.

Parameter	Default	Description
EnableS3Notification	`True`	An option to enable or disable notifications for Amazon S3 buckets. The default option is recommend ed for most cases.
LogProcessorRoleName	*Optional input*	Specify a role name for the log processor. The name should NOT duplicate an existing role name. If no name is specified, a random name is generated.
QueueName	*Optional input*	Specify a queue name for an Amazon SQS queue. The name should NOT duplicate an existing queue name. If no name is given, a random name will be generated.

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

View dashboard

The dashboard includes the following visualizations.

Visualization Name	Source Field	Description
Global Filters	* account-id * Region * vpc-id * subnet-id * action * flow-direction * log-status * protocol-code * type	The charts are filtered according to Account ID, Region, VPC ID, and other conditions.
Total Requests	* log event	Shows the total number of network requests logged by VPC Flow Logs during a selected time period.
Request History	* log event	Presents a bar chart that displays the distribution of events over time.
Requests By VPC ID	* vpc-id	Displays the proportional breakdown of network requests by source VPC using a pie chart.
Total Requests By Action	* action	Displays the total volume of requests segmented by action over time.
Total Bytes	* bytes	Provides visibility into overall bandwidth usage and traffic patterns across the monitored VPCs, subnets, network interfaces, and security groups.
Total Packets	* packets	Displays total logged packets over time to visualize trends, surges, and dips.
Bytes Metric	* bytes * flow-direction	Shows the distribution of incoming (Ingress) and

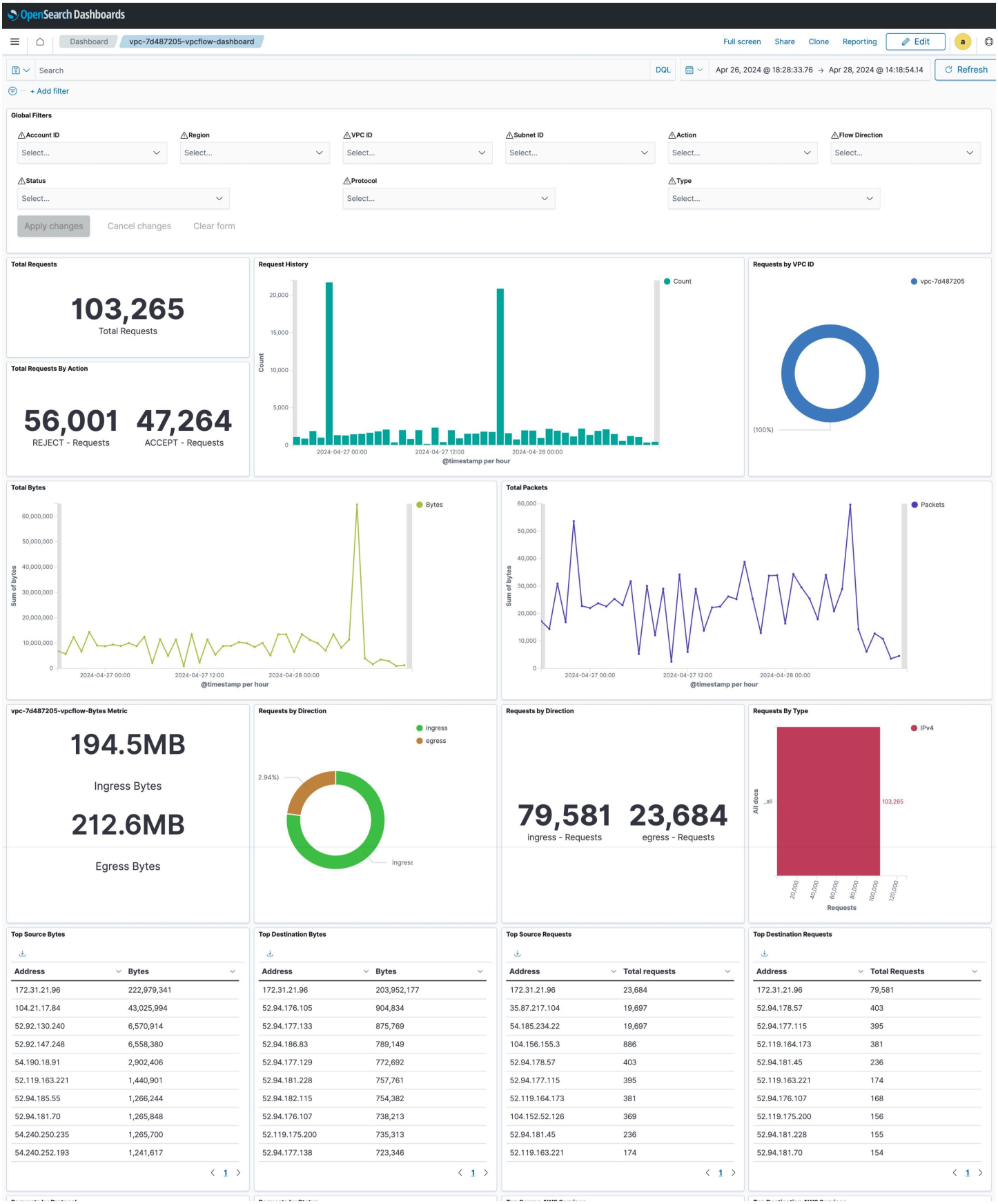
Visualization Name	Source Field	Description
		outgoing (Egress) network traffic volumes in bytes across the range of flows logged by VPC Flow Logs over a time period.
Requests By Direction	* flow-direction	Provides visibility into the proportional composition of incoming versus outgoing requests.
Requests By Direction	* flow-direction	Displays the total number of network flows logged by VPC Flow Logs segmented by traffic direction - Ingress vs Egress.
Requests By Type	* type	Shows the volume of flows for each type. This provides visibility into the protocol composition of network requests traversing the environment.
Top Source Bytes	* srcaddr * bytes	Displays the source IP addresses transmitting the highest outbound volume of data during the selected time period.
Top Destination Bytes	* dstaddr * bytes	Enables you to monitor and analyze outbound traffic from your VPC to external destinati ons.

Visualization Name	Source Field	Description
Top Source Requests	* srcaddr	Allows you to see which resources inside your VPC are initiating external requests.
Top Destination Requests	* dstaddr	Allows you to see which external hosts are being contacted most by your VPC resources.
Requests by Protocol	* protocol-code	Displays network flows logged by VPC Flow Logs segmented by traffic type - TCP, UDP, ICMP.
Requests by Status	* log-status	Provides a breakdown of network flows by their traffic status - Accepted, Rejected, or Other.
Top Sources AWS Services	* pkt-src-aws-service	Show the proportional distribution of flows originati ng from top AWS sources like Amazon S3, CloudFron t, Lambda, etc. during the selected time period.
Top Destination AWS Services	* pkt-dst-aws-service	Provide visibility into IP traffic going to and from AWS services located outside your VPC. By enabling flow logs on VPC subnets/interfaces and filtering on traffic with an ACCEPT action, you can view outbound flows from your VPC to various AWS services.

Visualization Name	Source Field	Description
Network Flow	* srcaddr * dstaddr	Allows you to view informati on about the IP traffic going to and from network interfaces in your VPC.
Heat Map	* srcaddr * dstaddr	Offers a visual summary of connections between source and destination IPs in your flow log data.
Egress Traffic Path	* traffic-path	Allows you to enable flow logging on VPC network interfaces to capture information about all IP traffic going to and from that interface.
Search	* @timestamp * account-i d * vpc-id * flow-direction * action * protocol-code * srcaddr * scaport * dstaddr * dstport * bytes * packets * log-status	Searching through the detailed flow log data allows pinpoint analysis of traffic around security events, network issues, changes in usage patterns, and more.

You can access the built-in dashboard in Amazon OpenSearch Service to view log data. For more information, see the Access Dashboard.

VPC Flow Logs sample dashboard.


OpenSearch Dashboards
Dashboard
vpc-7d487205-vpcflow-dashboard
Full screen
Share
Clone
Reporting
Edit
Search
DQL
Apr 26, 2024 @ 18:28:33.76 → Apr 28, 2024 @ 14:18:54.14
Refresh
+ Add filter
Global Filters
Account ID
Region
VPC ID
Subnet ID
Action
Flow Direction
Status
Protocol
Type
Select...
Apply changes
Cancel changes
Clear form
Total Requests
103,265
Total Requests
Total Requests By Action
56,001
REJECT - Requests
47,264
ACCEPT - Requests
Request History
Count
@timestamp per hour
Requests by VPC ID
vpc-7d487205
(100%)
Total Bytes
Bytes
Sum of bytes
@timestamp per hour
Total Packets
Packets
Sum of bytes
@timestamp per hour
vpc-7d487205-vpcflow-Bytes Metric
194.5MB
Ingress Bytes
212.6MB
Egress Bytes
Requests by Direction
ingress
egress
Requests by Direction
79,581
ingress - Requests
23,684
egress - Requests
Requests By Type
IPv4
All docs
103,265
Requests
Top Source Bytes
Address
Bytes
172.31.21.96 222,979,341
104.21.17.84 43,025,994
52.92.130.240 6,570,914
52.92.147.248 6,558,380
54.190.18.91 2,902,406
52.119.163.221 1,440,901
52.94.185.55 1,266,244
52.94.181.70 1,265,848
54.240.250.235 1,265,700
54.240.252.193 1,241,617
Top Destination Bytes
Address
Bytes
172.31.21.96 203,952,177
52.94.176.105 904,834
52.94.177.133 875,769
52.94.186.83 789,149
52.94.177.129 772,692
52.94.181.228 757,761
52.94.182.115 754,382
52.94.176.107 738,213
52.119.175.200 735,313
52.94.177.138 723,346
Top Source Requests
Address
Total requests
172.31.21.96 23,684
35.87.217.104 19,697
54.185.234.22 19,697
104.156.155.3 886
52.94.178.57 403
52.94.177.115 395
52.119.164.173 381
104.152.52.126 369
52.94.181.45 236
52.119.163.221 174
Top Destination Requests
Address
Total Requests
172.31.21.96 79,581
52.94.178.57 403
52.94.177.115 395
52.119.164.173 381
52.94.181.45 236
52.119.163.221 174
52.94.176.107 168
52.119.175.200 156
52.94.181.228 155
52.94.181.70 154

Create log ingestion (Light Engine)

Using the Centralized Logging with OpenSearch Console

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose the Create a log ingestion button.
4. In the **AWS Services** section, choose **AWS VPC Flow**.
5. Choose **Light Engine**, choose **Next**.
6. Under **Specify settings**, choose **Automatic** or **Manual** for **CloudFront logs enabling**. The automatic mode will detect the CloudFront log location automatically.
   - For **Automatic** mode, choose the VPC Flow from the dropdown list.
   - For Standard Log, the solution will automatically detect the log location if logging is enabled.
   - For Manual mode, enter the VPC Flow ID and VPC Flow Log location.
   - (Optional) If you are ingesting CloudFront logs from another account, select a linked account from the **Account** dropdown list first.
7. Choose **Next**.
8. In the **Specify Light Engine Configuration** section, if you want to ingest associated templated Grafana dashboards, select **Yes** for the sample dashboard.
9. You can choose an existing Grafana, or if you must import a new one, you can go to Grafana for configuration.
10. Select an S3 bucket to store partitioned logs and define a name for the log table. We have provided a predefined table name, but you can modify it according to your business needs.
11. If needed, change the log processing frequency, which is set to **5** minutes by default, with a minimum processing frequency of **1** minute.
12. In the **Log Lifecycle** section, enter the log merge time and log archive time. We have provided default values, but you can adjust them based on your business requirements.
13. Select **Next**.
14. If desired, add tags.
15. Select **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - VpcFlow Standard Log Ingestion* solution in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Regions		Template
AWS China Regions		Template

1. Log in to the AWS Management Console and select the preceding button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.
2. To launch the stack 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.

   a. Parameters for **Pipeline settings**

Parameter	Default	Description
Pipeline Id	`<Requires input>`	The unique identifier for the pipeline is essential if you must create multiple Application Load Balancer pipelines and write different Application Load Balancer

Parameter	Default	Description
		logs into separate tables. For uniqueness, you can generate a unique pipeline identifier using uuidgenerator.
Staging Bucket Prefix	`AWSLogs/VpcFlowLogs`	The storage directory for logs in the temporary storage area should ensure the uniqueness and non-overlapping of the Prefix for different pipelines.

b. Parameters for **Destination settings**

Parameter	Default	Description
Centralized Bucket Name	*`<Requires input>`*	Centralized S3 bucket name. For example, centralized-logging-bucket.
Centralized Bucket Prefix	`datalake`	Centralized bucket prefix. By default, the data base location is s3://{Centralized Bucket Name}/{Centralized Bucket Prefix}/amazon_cl_centralized.
Centralized Table Name	`WAF`	Table name for writing data to the centralized database. You can modify it if needed.

c. Parameters for **Scheduler settings**

Parameter	Default	Description
LogProcessor Schedule Expression	`rate(5 minutes)`	Task scheduling expression for performing log processing, with a default value of executing the LogProcessor every 5 minutes. Configuration for reference.
LogMerger Schedule Expression	`cron(0 1 * ? )`	Task scheduling expression for performing log merging, with a default value of executing the LogMerger at 1 AM every day. Configuration for reference.
LogArchive Schedule Expression	`cron(0 2 * ? )`	Task scheduling expression for performing log archiving, with a default value of executing the LogArchive at 2 AM every day. Configuration for reference.
Age to Merge	7	Small file retention days, with a default value of 7, indicating that logs older than 7 days will be merged into small files. It can be adjusted as needed.

Parameter	Default	Description
Age to Archive	30	Log retention days, with a default value of 30, indicating that data older than 30 days will be archived and deleted. It can be adjusted as needed.

d. Parameters for **Notification settings**

Parameter	Default	Description
Notification Service	SNS	Notification method for alerts. If your main stack is using China, you can only choose the SNS method. If your main stack is using Global, you can choose either the SNS or SES method.

Parameter	Default	Description
Recipients	*<Requires input>*	Alert notification: If the Notification Service is SNS, enter the SNS Topic ARN here so that you have the necessary permissions. If the Notification Service is SES, enter the email addresses separated by commas here, ensuring that the email addresses are already Verified Identities in SES. The adminEmail provided during the creation of the main stack will receive a verification email by default.

e. Parameters for **Dashboard settings**

Parameter	Default	Description
Import Dashboards	FALSE	Whether to import the Dashboard into Grafana, with a default value of false. If set to true, you must provide the Grafana URL and Grafana Service Account Token.
Grafana URL	*<Requires input>*	Grafana access URL，for example: https://alb-72277319.us-west-2.elb.amazonaws.com.

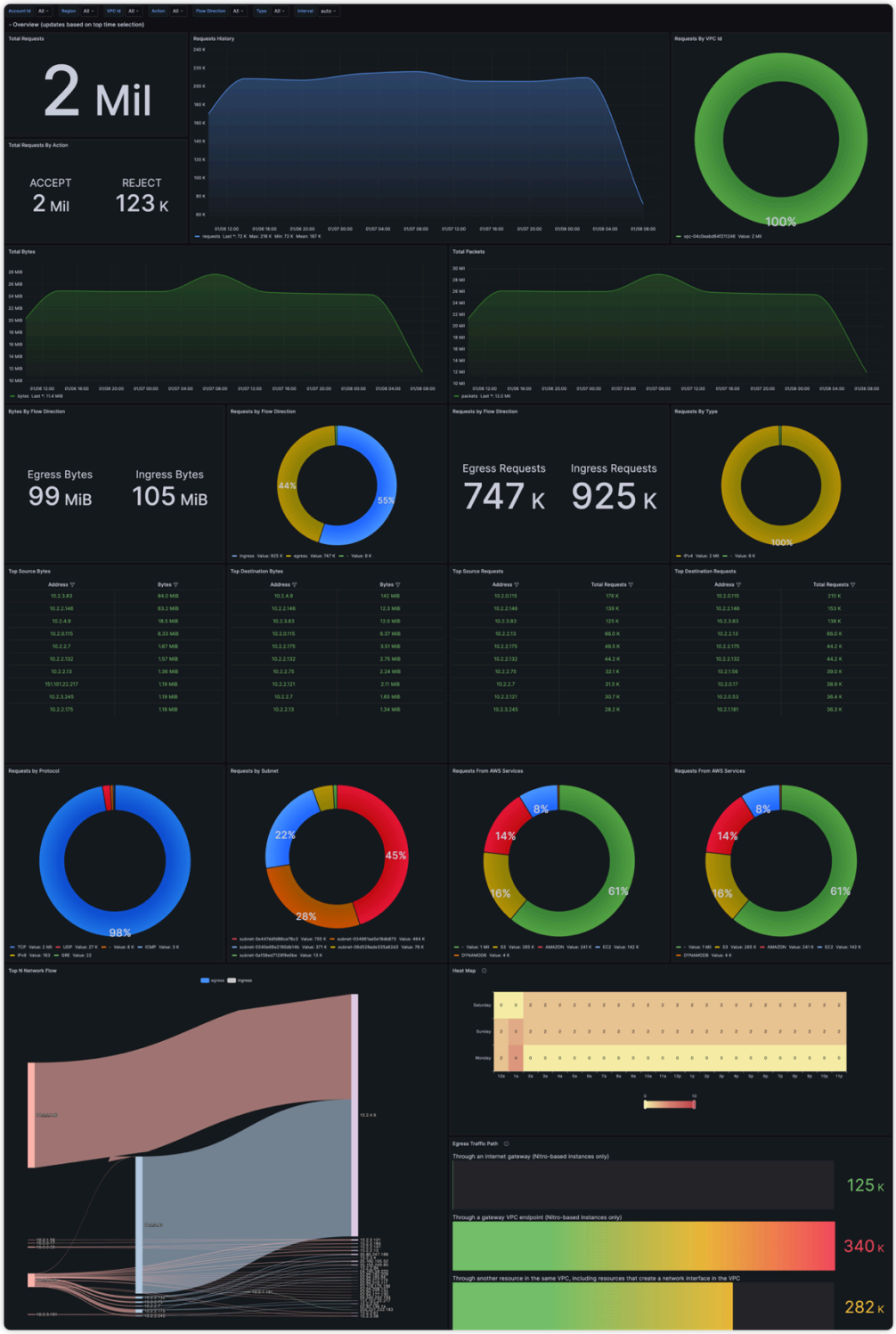
Parameter	Default	Description
Grafana Service Account Token	*<Requires input>*	Grafana Service Account Token：Service Account Token created in Grafana.

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

View dashboard

VPC Flow Logs sample dashboard.



AWS Config logs

By default, AWS Config delivers configuration history and snapshot files to your Amazon S3 bucket.

You can create a log ingestion into Amazon OpenSearch Service either by using the Centralized Logging with OpenSearch console or by deploying a standalone CloudFormation stack.

Important

- AWS Config must be enabled in the same Region as the Centralized Logging with OpenSearch solution.
- The Amazon S3 bucket Region must be the same as the Centralized Logging with OpenSearch solution.
- The Amazon OpenSearch Service index is rotated on a daily basis by default, and you can adjust the index in the Additional Settings.

Create log ingestion (OpenSearch Engine)

Using the Centralized Logging with OpenSearch Console

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the navigation pane, under **Log Analytics Pipelines**, choose **Service Log**.
3. Choose the Create a log ingestion button.
4. In the AWS Services section, choose AWS Config Logs.
5. Choose **Next**.
6. Under Specify settings, choose Automatic or Manual for Log creation.
   - For **Automatic mode**, make sure that the S3 bucket location is correct, and enter the **AWS Config Name**.
   - For Manual mode, enter the AWS Config Name and Log location.
   - (Optional) If you are ingesting VPC Flow Logs from another account, select a linked account from the **Account** dropdown list first.
7. Choose **Next**.
8. In the Specify OpenSearch domain section, select an imported domain for the Amazon OpenSearch Service domain.
9. Choose **Yes** for **Sample dashboard** if you want to ingest an associated built-in Amazon OpenSearch Service dashboard.

10. You can change the **Index Prefix** of the target Amazon OpenSearch Service index if needed. The default prefix is your VPC name.
11. In the **Log Lifecycle** section, enter the number of days to manage the Amazon OpenSearch Service index lifecycle. The Centralized Logging with OpenSearch will create the associated Index State Management (ISM) policy automatically for this pipeline.
12. Choose **Next**.
13. Add tags if needed.
14. Choose **Create**.

Using the CloudFormation Stack

This automated AWS CloudFormation template deploys the *Centralized Logging with OpenSearch - AWS Config Log Ingestion* solution in the AWS Cloud.

	Launch in AWS Management Console	Download Template
AWS Standard Regions		Template
AWS China Regions		Template

1. Log in to the AWS Management Console and select the preceding button to launch the AWS CloudFormation template. You can also download the template as a starting point for your own implementation.
2. To launch the stack 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.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following parameters.

Parameter	Default	Description
Log Bucket Name	*<Requires input>*	The S3 bucket name that stores the logs.
Log Bucket Prefix	*<Requires input>*	The S3 bucket path prefix that stores the logs.
Log Source Account ID	*Optional input*	The AWS Account ID of the S3 bucket. Required for cross-account log ingestion (add a member account first). By default, the Account ID you logged in at **Step 1** will be used.
Log Source Region	*Optional input*	The AWS Region of the S3 bucket. By default, the Region you selected at **Step 2** will be used.
Log Source Account Assume Role	*Optional input*	The IAM Role ARN used for cross-account log ingestion . Required for cross-account log ingestion (Please add a member account first).
KMS-CMK ARN	*Optional input*	The KMS-CMK ARN for encryption. Leave it blank to create a new AWS KMS key.
Enable OpenSearch Ingestion as processor	*Optional input*	Ingestion table ARN. Leave empty if you do not use OSI as Processor.

Parameter	Default	Description
S3 Backup Bucket	*<Requires input>*	The S3 backup bucket name to store the failed ingestion logs.
Engine Type	OpenSearch	The engine type of the OpenSearch.
OpenSearch Domain Name	*<Requires input>*	The domain name of the Amazon OpenSearch Service cluster.
OpenSearch Endpoint	*<Requires input>*	The OpenSearch endpoint URL. For example, vpc-your_opensearch_domain_name-xcvgw6uu2o6zafsiefxubwuohe.us-east-1.es.amazonaws.com
Index Prefix	*<Requires input>*	The common prefix of OpenSearch index for the log. The index name will be <Index Prefix>-<Log Type>-<Other Suffix>.
Create Sample Dashboard	Yes	Whether to create a sample OpenSearch dashboard.
VPC ID	*<Requires input>*	Select a VPC that has access to the OpenSearch domain. The log processing Lambda will reside in the selected VPC.

Parameter	Default	Description
Subnet IDs	*<Requires input>*	Select at least two subnets that have access to the OpenSearch domain. The log processing Lambda will reside in the subnets. Make sure that the subnets have access to the Amazon S3 service.
Security Group ID	*<Requires input>*	Select a Security Group that will be associated with the log processing Lambda. Make sure that the Security Group has access to the OpenSearch domain.
Number Of Shards	5	Number of shards to distribute the index evenly across all data nodes. Keep the size of each shard between 10-50 GB.
Number of Replicas	1	Number of replicas for OpenSearch Index. Each replica is a full copy of an index. If the OpenSearch option is set to Domain with standby, you need to configure it to 2.

Parameter	Default	Description
Age to Warm Storage	*Optional input*	The age required to move the index into warm storage (for example, 7d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when warm storage is enabled in OpenSearch.
Age to Cold Storage	*Optional input*	The age required to move the index into cold storage (for example, 30d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). This is only effective when cold storage is enabled in OpenSearch.
Age to Retain	*Optional input*	The age to retain the index (for example, 180d). Index age is the time between its creation and the present. Supported units are d (days) and h (hours). If the value is "", the index will not be deleted.
Rollover Index Size	*Optional input*	The minimum size of the shard storage required to roll over the index (for example, 30GB).

Parameter	Default	Description
Index Suffix	`yyyy-MM-dd`	The common suffix format of OpenSearch index for the log (for example, yyyy-MM-dd, yyyy-MM-dd-HH). The index name will be <Index Prefix>-<Log Type>-<Index Suffix>-000001.
Compression type	`best_compression`	The compression type to use to compress stored data. Available values are best_compression and default.
Refresh Interval	`1s`	How often the index should refresh, which publishes its most recent changes and makes them available for searching. Can be set to -1 to disable refreshing. Default is 1s.
EnableS3Notification	`True`	An option to enable or disable notifications for Amazon S3 buckets. The default option is recommended for most cases.
LogProcessorRoleName	*Optional input*	Specify a role name for the log processor. The name should NOT duplicate an existing role name. If no name is specified, a random name is generated.

Parameter	Default	Description
QueueName	*Optional input*	Specify a queue name for an Amazon SQS queue. The name should NOT duplicate an existing queue name. If no name is given, a random name will be generated.

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

View dashboard

The dashboard includes the following visualizations.

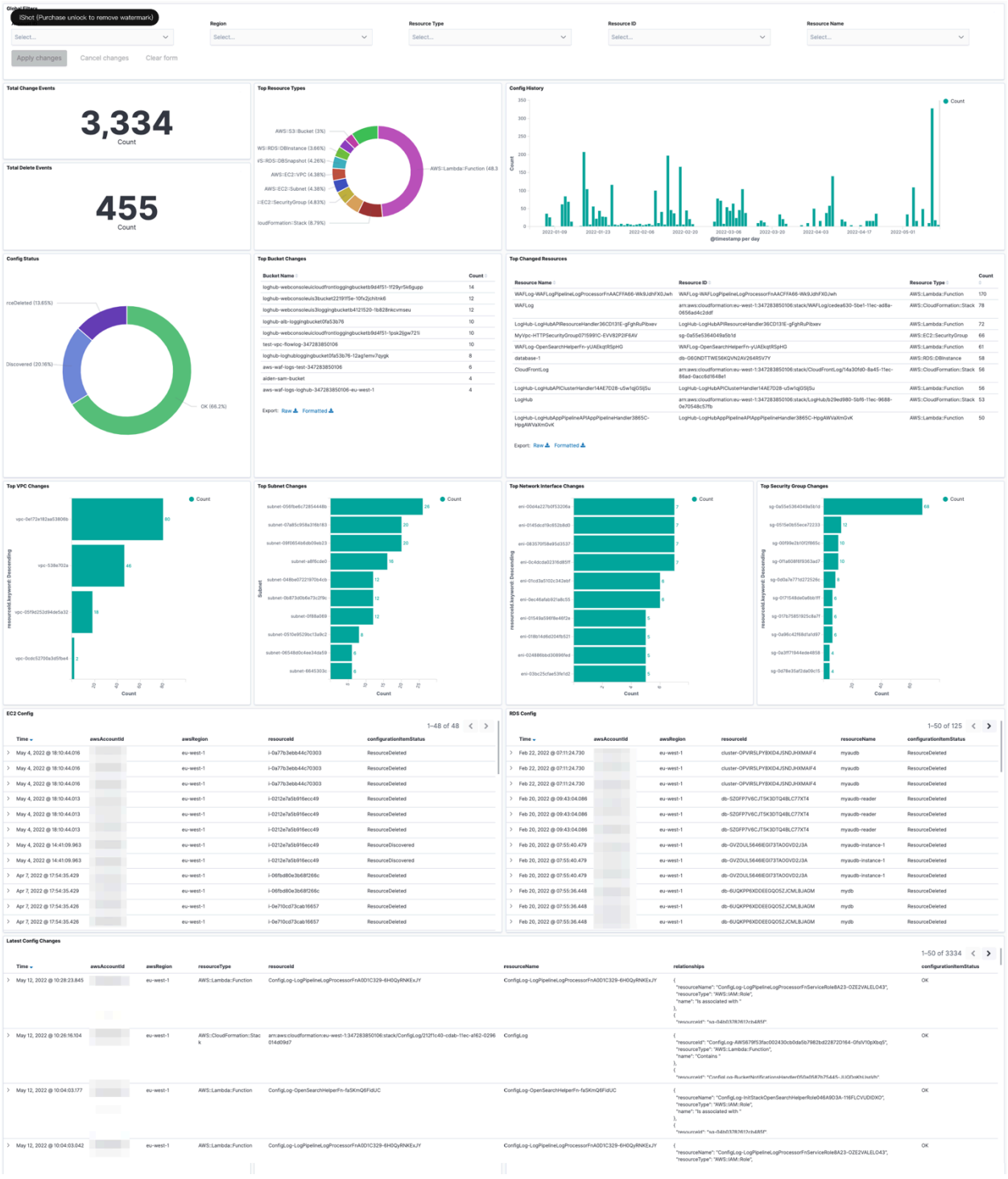
Visualization Name	Source Field	Description
Global Filters	awsAccountId awsRegion resourceType resourceId resourceName	The charts are filtered according to Account ID, Region, Resource Type, and other conditions.
Total Change Events	log event	Shows the number of configuration changes detected across all AWS resources during a selected time period.
Top Resource Types	resourceType	Displays the breakdown of configuration changes by the

Visualization Name	Source Field	Description
		most frequently modified AWS resource types during a selected time period.
Config History	log event	Presents a bar chart that displays the distribution of events over time.
Total Delete Events	log event	Shows the number of AWS resource deletion events detected by AWS Config during a selected time period.
Config Status	configurationItemStatus	Displays the operational state of the AWS Config service across monitored Regions and accounts.
Top S3 Changes	resourceName	Displays the Amazon S3 buckets undergoing the highest number of configura tion changes during a selected time period.
Top Changed Resources	resourceName resourceId resourceType	Displays the individual AWS resources undergoin g the highest number of configuration changes during a selected time period.
Top VPC Changes	resourceId	Presents a bar chart that Displays the Amazon VPCs undergoing the highest number of configuration changes during a selected time period.

Visualization Name	Source Field	Description
Top Subnet Changes	resourceId	Delivers targeted visibility into the subnets undergoin g the most transformation for governance, security, and stability.
Top Network Interface Changes	resourceId	Spotlights the Amazon VPC network interfaces seeing the most configuration changes during a selected period.
Top Security Group Changes	resourceId	Top 10 changed groups rank by total modification count.
EC2 Config	@timestamp awsAccoun tId awsRegion resourceId configurationItemStatus	Allows reconstructing the incremental changes applied to EC2 configurations over time for auditing.
RDS Config	@timestamp awsAccoun tId awsRegion resourceId resourceName configura tionItemStatus	Shows the configuration history and changes detected by AWS Config for RDS database resources
Latest Config Changes	@timestamp awsAccoun tId awsRegion resourceT ype resourceId resourceN ame relationships configura tionItemStatus	Offers an at-a-glance overview of infrastructure modifications.

You can access the built-in dashboard in Amazon OpenSearch Service to view log data. For more information, see the Access Dashboard.

AWS Config logs sample dashboard.


Global Filters
iShot (Purchase unlock to remove watermark)
Region
Resource Type
Resource ID
Resource Name
Apply changes
Cancel changes
Clear form
Total Change Events
3,334
Count
Total Delete Events
455
Count
Top Resource Types
Config History
Config Status
Top Bucket Changes
Top Changed Resources
Top VPC Changes
Top Subnet Changes
Top Network Interface Changes
Top Security Group Changes
EC2 Config
RDS Config
Latest Config Changes

Application logs

Centralized Logging with OpenSearch supports ingesting application logs from the following log sources:

- Instance Group: the solution automatically installs a log agent (Fluent Bit 1.9), collects application logs on EC2 instances and then sends logs into Amazon OpenSearch Service.
- Amazon EKS cluster: the solution generates all-in-one configuration file for customers to deploy the log agent (Fluent Bit 1.9) as a DaemonSet or Sidecar. After the log agent is deployed, the solution starts collecting pod logs and sends them to Amazon OpenSearch Service.
- Amazon S3: the solution either ingests logs in the specified Amazon S3 location continuously or performs one-time ingestion. You can also filter logs based on Amazon S3 prefix or parse logs with custom Log Config.
- Syslog: the solution collects syslog logs through UDP or TCP protocol.

After creating a log analytics pipeline, you can add more log sources to the log analytics pipeline. For more information, see add a new log source.

- Important If you are using Centralized Logging with OpenSearch to create an application log pipeline for the first time, you are recommended to learn the concepts and supported log formats and log sources.

Supported log formats and log sources

The table lists the log formats supported by each log source. For more information about how to create log ingestion for each log format, refer to Log Config.

Log Format	Instance Group	EKS Cluster	Amazon S3	Syslog
NGINX	Yes	Yes	Yes	No
Apache HTTP Server	Yes	Yes	Yes	No
JSON	Yes	Yes	Yes	Yes

Log Format	Instance Group	EKS Cluster	Amazon S3	Syslog
Single-line Text	Yes	Yes	Yes	Yes
Multi-line Text	Yes	Yes	Yes (Not support in Light Engine Mode)	No
Multi-line Text (Spring Boot)	Yes	Yes	Yes (Not support in Light Engine Mode)	No
Syslog RFC5424/RFC3164	No	No	No	Yes
Syslog Custom	No	No	No	Yes
Windows Event	Yes	No	No	No
IIS	Yes	No	No	No

Instance group

An instance group represents a group of EC2 Linux instances, which enables the solution to associate a Log Config with multiple EC2 instances quickly. Centralized Logging with OpenSearch uses Systems Manager Agent (SSM Agent) to install/configure Fluent Bit agent, and sends log data to Kinesis Data Streams.

This article guides you to create a log pipeline that ingests logs from an Instance Group.

Create a log analytics pipeline (OpenSearch Engine)

Prerequisites

Make sure you have imported an Amazon OpenSearch Service domain. For more information, see Domain operations.

Follow these steps:

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under Log Analytics Pipelines, choose Application Log.
3. Choose Create a pipeline.
4. Choose **Instance Group** as Log Source, choose **Amazon OpenSearch Service**, and choose **Next**.
5. Select an instance group. If you have no instance group yet, choose **Create Instance Group** at the top right corner, and follow the instructions to create an instance group. After that, choose **Refresh** and then select the newly created instance group.
6. (Auto Scaling group only) If your instance group is created based on an Auto Scaling group, after ingestion status become "Created", then you can find the generated Shell Script in the instance group's detail page. Copy the shell script and update the User Data of the Auto Scaling Launch configurations or Launch template.
7. Keep the default Permission grant method.
8. (Optional) If you choose, I will manually add the following required permissions after pipeline creation, continue to do the following:
9. Choose **Expand to view required permissions** and copy the provided JSON policy.
10. Go to AWS Management Console.
11. On the left navigation pane, choose **IAM**, and select **Policies** under **Access management**.
12. Choose **Create Policy**, choose **JSON**, and replace all the content inside the text block. Make sure to substitute <YOUR ACCOUNT ID> with your account id.
13. Choose **Next**, and then enter a name for this policy.
14. Attach the policy to your EC2 instance profile to grant the log agent permissions to send logs to the application log pipeline. If you are using the Auto Scaling group, you must update the IAM instance profile associated with the Auto Scaling group. If needed, you can follow the documentation to update your launch template or launch configuration.
15. Choose **Next**.

You have created a log source for the log analytics pipeline. Now you are ready to make further configurations for the log analytics pipeline with Amazon EC2 instance group as log source.

1. Select a log config. If you do not find the desired log config from the dropdown list, choose **Create New**, and follow instructions in Log Config.
2. Enter a **Log Path** to specify the location of logs to be collected. You can use , to separate multiple paths. Choose **Next**.

3. Specify **Index name** in lowercase.
4. In the **Buffer** section, choose **S3** or **Kinesis Data Streams**. If you don't want the buffer layer, choose **None**. Refer to the Log Buffer for more information about choosing the appropriate buffer layer.
   - Amazon S3 buffer parameters

Parameter	Default	Description
S3 Bucket	*A log bucket will be created by the solution.*	You can also select a bucket to store the log data.
S3 Bucket Prefix	`AppLogs/<index-prefix>/year=%Y/month=%m/day=%d`	The log agent appends the prefix when delivering the log files to the S3 bucket.
Buffer size	`50 MiB`	The maximum size of log data cached at the log agent side before delivering to Amazon S3. For more information, see Data Delivery Frequency.
Buffer interval	`60 seconds`	The maximum interval of the log agent to deliver logs to Amazon S3. For more information, see Data Delivery Frequency.
Compression for data records	`Gzip`	The log agent compresses records before delivering them to the S3 bucket.

   - Kinesis Data Streams buffer parameters

Parameter	Default	Description
Shard number	*<Requires input>*	The number of shards of the Kinesis Data Streams. Each shard can have up to 1,000 records per second and total data write rate of 1MB per second.
Enable auto scaling	No	This solution monitors the utilization of Kinesis Data Streams every 5 minutes, and scales in/out the number of shards automatic ally. The solution will scale in/out for a maximum of 8 times within 24 hours.
Maximum Shard number	*<Requires input>*	Required if auto scaling is enabled. The maximum number of shards.

> **Important**
>
> Important You may observe duplicate logs in OpenSearch if a threshold error occurs in Kinesis Data Streams (KDS). This is because the Fluent Bit log agent uploads logs in chunk (contains multiple records), and will retry the chunk if upload failed. Each KDS shard can support up to 1,000 records per second for writes, up to a maximum total data write rate of 1 MB per second. Estimate your log volume and choose an appropriate shard number.

5. Choose **Next**.
6. In the **Specify OpenSearch domain** section, select an imported domain for **Amazon OpenSearch Service domain**.

7. In the **Log Lifecycle** section, enter the number of days to manage the Amazon OpenSearch Service index lifecycle. The Centralized Logging with OpenSearch will create the associated Index State Management (ISM) policy automatically for this pipeline.
8. In the **Select log processor** section, choose the log processor.
   - When selecting Lambda as a log processor, you can configure the Lambda concurrency if needed.
   - (Optional) OSI as log processor is now supported in these Regions. When OSI is selected, enter the minimum and maximum number of OCU. For more information, see Scaling pipelines.
9. Choose **Next**.
10. Enable **Alarms** if needed and select an existing SNS topic. If you choose **Create a new SNS topic**, provide a name and an email address for the new SNS topic.
11. Add tags if needed.
12. Choose **Create**.
13. Wait for the application pipeline to turn to "Active" state.

Create a log analytics pipeline (Light Engine)

Follow these steps:

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Log Analytics Pipelines**, choose **Application Log**.
3. Choose **Create a pipeline**.
4. Choose **Instance Group** as Log Source, choose **Light Engine**, and choose **Next**.
5. Select an instance group. If you have no instance group yet, choose **Create Instance Group** at the top right corner, and follow the instructions to create an instance group. After that, choose **Refresh** and then select the newly created instance group.
6. (Auto Scaling group only) If your instance group is created based on an Auto Scaling group, after ingestion status become "Created", then you can find the generated Shell Script in the instance group's detail page. Copy the shell script and update the User Data of the Auto Scaling Launch configurations or Launch template.
7. Keep the default **Permission grant method**.
8. (Optional) If you choose **I will manually add the below required permissions after pipeline creation**, continue to do the following:

a. Choose **Expand to view required permissions** and copy the provided JSON policy.

b. Go to AWS Management Console.

c. On the left navigation pane, choose **IAM**, and select **Policies** under **Access management**.

d. Choose **Create Policy**, choose **JSON**, and replace all the content inside the text block. Make sure to substitute <YOUR ACCOUNT ID> with your account id.

e. Choose **Next**, and then enter a name for this policy.

f. Attach the policy to your EC2 instance profile to grant the log agent permissions to send logs to the application log pipeline. If you are using the Auto Scaling group, you must update the IAM instance profile associated with the Auto Scaling group. If needed, you can follow the documentation to update your launch template or launch configuration.

9. Choose **Next**.

You have created a log source for the log analytics pipeline. Now you are ready to make further configurations for the log analytics pipeline with Amazon EC2 instance group as log source.

1. Select a log config. If you do not find the desired log config from the dropdown list, choose **Create New**, and follow instructions in Log Config.
2. Enter a **Log Path** to specify the location of logs to be collected. You can use , to separate multiple paths. Choose **Next**.
3. In the **Buffer** section, configure Amazon S3 buffer parameters.

Parameter	Default	Description
S3 Bucket	*A log bucket will be created by the solution.*	You can also select a bucket to store the log data.
Buffer size	`50 MiB`	The maximum size of log data cached at the log agent side before deliverin g to Amazon S3. For more information, see Data Delivery Frequency.
Buffer interval	`60 seconds`	The maximum interval of the log agent to deliver

Parameter	Default	Description
		logs to Amazon S3. For more information, see Data Delivery Frequency.
Compression for data records	`Gzip`	The log agent compresses records before delivering them to the S3 bucket.

4. Choose **Next**.
5. In the **Specify Light Engine Configuration** section, if you want to ingest an associated templated Grafana dashboard, select **Yes** for the sample dashboard.
6. Choose an existing Grafana, or import a new one by making configurations in Grafana.
7. Select an Amazon S3 bucket to store partitioned logs and give a name to the log table. The solution provides a predefined table name, but you can modify it according to your needs.
8. Modify the log processing frequency if needed, which is set to **5** minutes by default with a minimum processing frequency of **1** minute.
9. In the **Log Lifecycle** section, enter the log merger time and lag archive time. The solution provides default values, which you can modify according to your needs.
10. Choose **Next**.
11. Enable **Alarms** if needed and select an existing SNS topic. If you choose **Create a new SNS topic**, provide a name and an email address for the new SNS topic.
12. Add tags if needed.
13. Choose **Create**.
14. Wait for the application pipeline to turn to "Active" state.

Amazon EKS cluster

For Amazon Elastic Kubernetes Service (Amazon EKS) clusters, Centralized Logging with OpenSearch generates an all-in-one configuration file for you to deploy the log agent (Fluent Bit 1.9) as a DaemonSet or Sidecar. After the log agent is deployed, the solution starts collecting pod logs.

The following guides you to create a log pipeline that ingests logs from an Amazon EKS cluster.

Create a log analytics pipeline (OpenSearch Engine)

Prerequisites

Make sure you have imported an Amazon OpenSearch Service domain. For more information, see Domain operations.

Follow these steps:

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Log Analytics Pipelines**, choose **Application Log**.
3. Choose **Create a pipeline**.
4. Choose **Amazon EKS** as Log Source, choose **Amazon OpenSearch Service**, and choose **Next**.
5. Choose the AWS account in which the logs are stored.
6. Choose an EKS Cluster. If no clusters are imported yet, choose **Import an EKS Cluster** and follow instructions to import an EKS cluster. After that, select the newly imported EKS cluster from the dropdown list.
7. Choose **Next**.

You have created a log source for the log analytics pipeline. Now you are ready to make further configurations for the log analytics pipeline with the Amazon EKS cluster as log source.

1. Select a log config. If you do not find the desired log config from the dropdown list, choose **Create New** and follow instructions in Log Config.
2. Enter a **Log Path** to specify the location of logs you want to collect. You can use, to separate multiple paths. Choose **Next**.
3. Specify **Index name** in lowercase.
4. In the **Buffer** section, choose **S3** or **Kinesis Data Streams**. If you don't want the buffer layer, choose **None**. Refer to the Log Buffer for more information about choosing the appropriate buffer layer.
   - Amazon S3 buffer parameters

Parameter	Default	Description
S3 Bucket	*A log bucket will be created by the solution.*	You can also select a bucket to store the log data.
S3 Bucket Prefix	`AppLogs/<index-prefix>/year=%Y/month=%m/day=%d`	The log agent appends the prefix when delivering the log files to the S3 bucket.
Buffer size	`50 MiB`	The maximum size of log data cached at the log agent side before delivering to Amazon S3. For more information, see Data Delivery Frequency.
Buffer interval	`60 seconds`	The maximum interval of the log agent to deliver logs to Amazon S3. For more information, see Data Delivery Frequency.
Compression for data records	`Gzip`	The log agent compresses records before delivering them to the S3 bucket.

- Kinesis Data Streams buffer parameters

Parameter	Default	Description
Shard number	*`<Requires input>`*	The number of shards of the Kinesis Data Streams. Each shard can have up to 1,000 records per second and total data write rate of 1MB per second.

Parameter	Default	Description
Enable auto scaling	No	This solution monitors the utilization of Kinesis Data Streams every 5 minutes, and scales in/out the number of shards automatically. The solution will scale in/out for a maximum of 8 times within 24 hours.
Maximum Shard number	*<Requires input>*	Required if auto scaling is enabled. The maximum number of shards.

> **Important**
>
> You may observe duplicate logs in OpenSearch if a threshold error occurs in Kinesis Data Streams (KDS). This is because the Fluent Bit log agent uploads logs in chunk (contains multiple records), and will retry the chunk if upload failed. Each KDS shard can support up to 1,000 records per second for writes, up to a maximum total data write rate of 1 MB per second. Estimate your log volume and choose an appropriate shard number.

5. Choose **Next**.
6. In the **Specify OpenSearch domain** section, select an imported domain for **Amazon OpenSearch Service domain**.
7. In the **Log Lifecycle** section, enter the number of days to manage the Amazon OpenSearch Service index lifecycle. The Centralized Logging with OpenSearch will create the associated Index State Management (ISM) policy automatically for this pipeline.
8. In the **Select log processor** section, choose the log processor.
   - When selecting Lambda as a log processor, you can configure the Lambda concurrency if needed.
   - (Optional) OSI as log processor is now supported in these Regions. When OSI is selected, enter the minimum and maximum number of OCU. For more information, see Scaling pipelines.

9. Choose **Next**.
10. Enable **Alarms** if needed and select an existing SNS topic. If you choose **Create a new SNS topic**, please provide a name and an email address for the new SNS topic.
11. Add tags if needed.
12. Choose **Create**.
13. Wait for the application pipeline to turn to "Active" state.

Create a log analytics pipeline (Light Engine)

Follow these steps:

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Log Analytics Pipelines**, choose **Application Log**.
3. Choose **Create a pipeline**.
4. Choose **Amazon EKS** as Log Source, choose **Light Engine**, and choose **Next**.
5. Choose the AWS account in which the logs are stored.
6. Choose an EKS Cluster. If no clusters are imported yet, choose **Import an EKS Cluster** and follow instructions to import an EKS cluster. After that, select the newly imported EKS cluster from the dropdown list.
7. Choose **Next**.

You have created a log source for the log analytics pipeline. Now you are ready to make further configurations for the log analytics pipeline with the Amazon EKS cluster as log source.

1. Select a log config. If you do not find the desired log config from the dropdown list, choose **Create New** and follow instructions in Log Config.
2. Enter a **Log Path** to specify the location of logs you want to collect. You can use, to separate multiple paths. Choose **Next**.
3. In the **Buffer** section, configure Amazon S3 buffer parameters.
   - Amazon S3 buffer parameters

Parameter	Default	Description
S3 Bucket	*A log bucket will be created by the solution.*	You can also select a bucket to store the log data.
Buffer size	`50 MiB`	The maximum size of log data cached at the log agent side before deliverin g to Amazon S3. For more information, see Data Delivery Frequency.
Buffer interval	`60 seconds`	The maximum interval of the log agent to deliver logs to Amazon S3. For more information, see Data Delivery Frequency.
Compression for data records	`Gzip`	The log agent compresse s records before delivering them to the S3 bucket.

4. Choose **Next**.
5. In the **Specify Light Engine Configuration** section, if you want to ingest an associated templated Grafana dashboard, select **Yes** for the sample dashboard.
6. Choose an existing Grafana, or import a new one by making configurations in Grafana.
7. Select an Amazon S3 bucket to store partitioned logs and give a name to the log table. The solution provides a predefined table name, but you can modify it according to your needs.
8. Modify the log processing frequency if needed, which is set to **5** minutes by default with a minimum processing frequency of **1** minute.
9. In the **Log Lifecycle** section, enter the log merger time and lag archive time. The solution provides default values, which you can modify according to your needs.
10. Choose **Next**.
11. Enable **Alarms** if needed and select an exiting SNS topic. If you choose **Create a new SNS topic**, provide a name and an email address for the new SNS topic.

12. Add tags if needed.
13. Choose **Create**.
14. Wait for the application pipeline to turn to "Active" state.

Amazon S3

For Amazon S3, Centralized Logging with OpenSearch ingests logs in a specified Amazon S3 location continuously or performs one-time ingestion. You can also filter logs based on Amazon S3 prefix or parse logs with custom Log Config.

The following guides you to create a log pipeline that ingests logs from an Amazon S3 bucket.

Create a log analytics pipeline (OpenSearch Engine)

Prerequisites

Make sure you have imported an Amazon OpenSearch Service domain. For more information, see Domain operations.

Follow these steps:

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Log Analytics Pipelines**, choose **Application Log**.
3. Choose **Create a pipeline**.
4. Choose **Amazon S3** as Log Source, choose **Amazon OpenSearch Service**, and choose **Next**.
5. Choose the Amazon S3 bucket where your logs are stored. If needed, enter **Prefix filter**, which is optional.
6. Choose **Ingestion mode** based on your need. If you want to ingest logs continuously, select **On-going**; if you only must ingest logs once, select **One-time**.
7. Specify **Compression format** if your log files are compressed, and choose **Next**.

You have created a log source for the log analytics pipeline. Now you are ready to make further configurations for the log analytics pipeline with Amazon S3 as log source.

1. Select a log config. If you do not find the desired log config from the dropdown list, choose **Create New**. Refer to Log Config for more information.
2. Choose **Next**.

3. Specify **Index name** in lowercase.
4. In the **Specify OpenSearch domain** section, select an imported domain for **Amazon OpenSearch Service domain**.
5. In the **Log Lifecycle** section, enter the number of days to manage the Amazon OpenSearch Service index lifecycle. The Centralized Logging with OpenSearch creates the associated Index State Management (ISM) policy automatically for this pipeline.
6. Choose **Next**.
7. Enable **Alarms** if needed and select an existing SNS topic. If you choose **Create a new SNS topic**, provide a name and an email address for the new SNS topic.
8. Add tags if needed.
9. Choose **Create**.
10. Wait for the application pipeline to turn to an "Active" state.

Create a log analytics pipeline (Light Engine)

Follow these steps:

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Log Analytics Pipelines**, choose **Application Log**.
3. Choose **Create a pipeline**.
4. Choose **Amazon S3** as Log Source, choose **Light Engine**, and choose **Next**.
5. Choose the Amazon S3 bucket where your logs are stored. If needed, enter **Prefix filter**, which is optional.
6. Choose **Ingestion mode** based on your need. If you want to ingest the log continuously, select **On-going**.

You have created a log source for the log analytics pipeline. Now you are ready to make further configurations for the log analytics pipeline with Amazon S3 as log source.

1. Select a log config. If you do not find the desired log config from the dropdown list, choose Create New. Refer to Log Config for more information.
2. Choose Next.
3. In the Specify Light Engine Configuration section, if you want to ingest associated templated Grafana dashboards, select Yes for the sample dashboard.

4. You can choose an existing Grafana, or if you must import a new one, you can go to Grafana for configuration.
5. Select an S3 bucket to store partitioned logs and define a name for the log table. We have provided a predefined table name, but you can modify it according to your business needs.
6. The log processing frequency is set to 5 minutes by default, with a minimum processing frequency of 1 minute.
7. In the Log Lifecycle section, enter the log merge time and log archive time. We have provided default values, but you can adjust them based on your business requirements.
8. Select Next.
9. Enable Alarms if needed and select an existing SNS topic. If you choose Create a new SNS topic, provide a name and an email address for the new SNS topic.
10. If desired, add tags.
11. Select Create.
12. Wait for the application pipeline to turn to "Active" state.

Syslog

Centralized Logging with OpenSearch collects syslog logs through UDP or TCP protocol.

The following guides you to create a log pipeline that ingests logs from a syslog endpoint.

Create a log analytics pipeline (OpenSearch Engine)

Prerequisites

Make sure you have imported an Amazon OpenSearch Service domain. For more information, see Domain operations.

Follow these steps:

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Log Analytics Pipelines**, choose **Application Log**.
3. Choose **Create a pipeline**.
4. Choose **Syslog Endpoint** as Log Source, choose **Amazon OpenSearch Service**, and choose **Next**.
5. Select **UDP** or **TCP** with custom port number. Choose **Next**.

You have created a log source for the log analytics pipeline. Now you are ready to make further configurations for the log analytics pipeline with syslog as log source.

1. Select a log config. If you do not find the desired log config from the dropdown list, choose **Create New** and follow instructions in Log Config.
2. Enter a **Log Path** to specify the location of logs you want to collect.
3. Specify **Index name** in lowercase.
4. In the **Buffer** section, choose **S3** or **Kinesis Data Streams**. If you don't want the buffer layer, choose **None**. Refer to the Log Buffer for more information about choosing the appropriate buffer layer.
   - Amazon S3 buffer parameters

Parameter	Default	Description
S3 Bucket	*A log bucket will be created by the solution.*	You can also select a bucket to store the log data.
S3 Bucket Prefix	`AppLogs/<index-prefix>/year=%Y/month=%m/day=%d`	The log agent appends the prefix when delivering the log files to the S3 bucket.
Buffer size	`50 MiB`	The maximum size of log data cached at the log agent side before delivering to Amazon S3. For more information, see Data Delivery Frequency.
Buffer interval	`60 seconds`	The maximum interval of the log agent to deliver logs to Amazon S3. For more information, see Data Delivery Frequency.
Compression for data records	`Gzip`	The log agent compresses records before delivering them to the S3 bucket.

- Kinesis Data Streams buffer parameters

Parameter	Default	Description
Shard number	*<Requires input>*	The number of shards of the Kinesis Data Streams. Each shard can have up to 1,000 records per second and total data write rate of 1MB per second.
Enable auto scaling	No	This solution monitors the utilization of Kinesis Data Streams every 5 minutes, and scales in/out the number of shards automatic ally. The solution will scale in/out for a maximum of 8 times within 24 hours.
Maximum Shard number	*<Requires input>*	Required if auto scaling is enabled. The maximum number of shards.

> **Important**
>
> You may observe duplicate logs in OpenSearch if a threshold error occurs in Kinesis Data Streams (KDS). This is because the Fluent Bit log agent uploads logs in chunk (contains multiple records), and will retry the chunk if upload failed. Each KDS shard can support up to 1,000 records per second for writes, up to a maximum total data write rate of 1 MB per second. Estimate your log volume and choose an appropriate shard number.

5. Choose **Next**.
6. In the **Specify OpenSearch domain** section, select an imported domain for **Amazon OpenSearch Service domain**.

7. In the **Log Lifecycle** section, enter the number of days to manage the Amazon OpenSearch Service index lifecycle. The Centralized Logging with OpenSearch will create the associated Index State Management (ISM) policy automatically for this pipeline.
8. In the **Select log processor** section, choose the log processor.
   - When selecting Lambda as a log processor, you can configure the Lambda concurrency if needed.
   - (Optional) OSI as log processor is now supported in these Regions. When OSI is selected, enter the minimum and maximum number of OCU. For more information, see Scaling pipelines.
9. Choose **Next**.
10. Enable **Alarms** if needed and select an existing SNS topic. If you choose **Create a new SNS topic**, provide a name and an email address for the new SNS topic.
11. Add tags if needed.
12. Choose **Create**.
13. Wait for the application pipeline to turn to "Active" state.

Create a log analytics pipeline (Light Engine)

Follow these steps:

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Log Analytics Pipelines**, choose **Application Log**.
3. Choose **Create a pipeline**.
4. Choose **Syslog Endpoint** as Log Source, choose **Light Engine**, and choose **Next**.
5. Select **UDP** or **TCP** with custom port number. Choose **Next**.

You have created a log source for the log analytics pipeline. Now you are ready to make further configurations for the log analytics pipeline with syslog as log source.

1. Select a log config. If you do not find the desired log config from the dropdown list, choose **Create New** and follow instructions in Log Config.
2. Enter a **Log Path** to specify the location of logs you want to collect.
3. In the **Buffer** section, configure Amazon S3 buffer parameters.
   - Amazon S3 buffer parameters

Parameter	Default	Description
S3 Bucket	*A log bucket will be created by the solution.*	You can also select a bucket to store the log data.
Buffer size	`50 MiB`	The maximum size of log data cached at the log agent side before deliverin g to Amazon S3. For more information, see Data Delivery Frequency.
Buffer interval	`60 seconds`	The maximum interval of the log agent to deliver logs to Amazon S3. For more information, see Data Delivery Frequency.
Compression for data records	`Gzip`	The log agent compresse s records before delivering them to the S3 bucket.

4. Choose **Next**.
5. In the **Specify Light Engine Configuration** section, if you want to ingest an associated templated Grafana dashboard, select **Yes** for the sample dashboard.
6. Choose an existing Grafana, or import a new one by making configurations in Grafana.
7. Select an Amazon S3 bucket to store partitioned logs and give a name to the log table. The solution provides a predefined table name, but you can modify it according to your needs.
8. Modify the log processing frequency if needed, which is set to **5** minutes by default with a minimum processing frequency of **1** minute.
9. In the **Log Lifecycle** section, enter the log merger time and lag archive time. The solution provides default values, which you can modify according to your needs.
10. Choose **Next**.
11. Enable **Alarms** if needed and select an exiting SNS topic. If you choose **Create a new SNS topic**, provide a name and an email address for the new SNS topic.

12. Add tags if needed.
13. Choose **Create**.
14. Wait for the application pipeline to turn to "Active" state.

Pipeline resources

A log analytics pipeline can have more than one log source.

Log sources

You must create a log source first before collecting application logs. Centralized Logging with OpenSearch supports the following log sources:

- Instance group
- Amazon EKS cluster
- Amazon S3
- Syslog

For more information, see concepts.

Instance Group

An instance group represents a group of EC2 Linux instances, which enables the solution to associate a Log Config with multiple EC2 instances quickly. Centralized Logging with OpenSearch uses Systems Manager Agent(SSM Agent) to install/configure Fluent Bit agent, and sends log data to Kinesis Data Streams.

Prerequisites

Make sure that the instances meet the following requirements:

- SSM agent is installed on instances. Refer to install SSM agent on EC2 instances for Linux for more details.
- The AmazonSSMManagedInstanceCore policy is being associated with the instances.
- The OpenSSL 1.1 or later is installed. Refer to OpenSSL Installation for more details.
- The instances have network access to AWS Systems Manager.

- The instances have network access to Amazon Kinesis Data Streams, if you use it as the Log Buffer.
- The instances have network access to Amazon S3, if you use it as the Log Buffer.
- The operating system of the instances is supported by Fluent Bit. Refer to Supported Platform.

(Option 1) Select instances to create an Instance Group

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Log Source**, choose **Instance Group**.
3. Choose the **Create an instance group** button.
4. In the **Settings** section, specify a group name.
5. In the **Configuration** section, select **Instances**. You can use up to 5 tags to filter the instances.
6. Verify that all the selected instances' "Pending Status" is **Online**.
7. (Optional) If the selected instances' "Pending Status" are empty, choose the **Install log agent** button, and wait for "Pending Status" to become **Online**.
8. (Optional) If you want to ingest logs from another account, select a linked account in the **Account Settings** section to create an instance group log source from another account.
9. Choose **Create**.

> **Important**
>
> Important Use the Centralized Logging with OpenSearch console to install Fluent Bit agent on Ubuntu instances in **Beijing (cn-north-1) and Ningxia (cn-northwest-1)** Region will cause installation error. The Fluent Bit assets cannot be downloaded successfully. You must install the Fluent Bit agent by yourself.

(Option 2) Select an Auto Scaling group to create an Instance Group

When creating an Instance Group with Amazon EC2 Auto Scaling group, the solution will generate a shell script that you should include in the EC2 User Data.

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Log Source**, choose **Instance Group**.

3. Choose the **Create an instance group** button.
4. In the **Settings** section, specify a group name.
5. In the Configuration section, select Auto Scaling Groups.
6. In the **Auto Scaling groups** section, select the Auto Scaling group from which you want to collect logs.
7. (Optional) If you want to ingest logs from another account, select a linked account in the **Account Settings** section to create an instance group log source from another account.
8. Choose **Create**. After you created a Log Ingestion using the Instance Group, you can find the generated Shell Script in the details page.
9. Copy the shell script and update the User Data of the Auto Scaling Group's launch configurations or launch template. The shell script will automatically install Fluent Bit, SSM agent if needed, and download Fluent Bit configurations.
10. Once you have updated the launch configurations or launch template, you must start an instance refresh to update the instances within the Auto Scaling group. The newly launched instances will ingest logs to the OpenSearch cluster or the Log Buffer layer.

Amazon EKS Cluster

The EKS Cluster in Centralized Logging with OpenSearch refers to the Amazon Elastic Kubernetes Service (Amazon EKS) from which you want to collect pod logs. Centralized Logging with OpenSearch will guide you to deploy the log agent as a DaemonSet or Sidecar in the EKS Cluster.

> **⚠ Important**
>
> - Centralized Logging with OpenSearch does not support sending logs in one EKS cluster to more than one Amazon OpenSearch Service domain at the same time.
> - Make sure your EKS cluster's VPC is connected to the Amazon OpenSearch Service cluster's VPC so that logs can be ingested. Refer to VPC Connectivity for more details regarding approaches to connect VPCs.

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Log Source**, choose **EKS Cluster**.
3. Choose the **Import a Cluster** button.

4. Choose the **EKS Cluster** where Centralized Logging with OpenSearch collects logs from. (Optional) If you want to ingest logs from another account, select a linked account from the **Account** dropdown to import an EKS log source from another account.
5. Select **DaemonSet** or **Sidecar** as the log agent's deployment pattern.
6. Choose **Next**.
7. Specify the **Amazon OpenSearch Service** where Centralized Logging with OpenSearch sends the logs to.
8. Follow the guidance to establish a VPC peering connection between EKS's VPC and OpenSearch's VPC.
   - Create and accept VPC peering connections
   - Update your route tables for a VPC peering connection
   - Update your security groups to reference peer VPC groups
9. Choose **Next**.
10. Add tags if needed.
11. Choose **Create**.

Amazon S3

The Amazon S3 in Centralized Logging with OpenSearch refers to the Amazon S3 from which you want to collect application logs stored in your bucket. You can choose **On-going** or **One-time** to create your ingestion job.

> ⚠ **Important**
>
> - On-going means that the ingestion job will run when a new file is delivered to the specified Amazon S3 location.
> - One-time means that the ingestion job will run at creation and will only run once to load all files in the specified location.

Syslog

> **Important**
>
> Important To ingest logs, make sure your Syslog generator/sender's subnet is connected to Centralized Logging with OpenSearch's **two** private subnets. Refer to VPC Connectivity for more details about how to connect VPCs.

You can use a UDP or TCP custom port number to collect syslog in Centralized Logging with OpenSearch. Syslog refers to logs generated by Linux instance, routers, or network equipment. For more information, see Syslog in Wikipedia.

Add a new log source

A newly created log analytics pipeline has one log source. You can add more log sources into the log pipeline.

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left navigation pane, under Log Analytics Pipelines, choose Application Log.
3. Choose the log pipeline's ID.
4. Choose Create a source.
5. Follow the instructions in Instance Group, Amazon EKS cluster, Amazon S3, or Syslog to create a log source according to your need.

Log config

Centralized Logging with OpenSearch solution supports creating log configs for the following log formats:

- JSON
- Apache
- NGINX
- Syslog
- Single-line text
- Multi-line text

For more information, refer to supported log formats and log sources.

The following describes how to create a log config for each log format.

Create a JSON config

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Resources**, choose **Log Config**.
3. Choose the **Create a log config** button.
4. Specify Config Name.
5. Choose **JSON** in the log type dropdown list.
6. In the **Sample log parsing** section, paste a sample JSON log and choose **Parse log** to verify if the log parsing is successful.

   For example:

```
{"host":"81.95.250.9", "user-identifier":"-", "time":"08/Mar/2022:06:28:03 +0000",
 "method": "PATCH", "request": "/clicks-and-mortar/24%2f7", "protocol":"HTTP/2.0",
 "status":502, "bytes":24337, "referer": "https://www.investorturn-key.net/
functionalities/innovative/integrated"}
```

7. Check if each field type mapping is correct. You can change the type by selecting the dropdown menu in the second column. For all supported types, see Data Types.

   > **Note**
   >
   > You must specify the datetime of the log using the key "time". If not specified, system time will be added.

8. Specify the **Time format**. The format syntax follows strptime. Check this for details.
9. (Optional) In the **Filter** section, you add some conditions to filter logs at the log agent side. The solution will ingest logs that match ALL the specified conditions only.
10. Select **Create**.

Create an Apache HTTP server log config

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Resources**, choose **Log Config**.

3. Choose the **Create a log config** button.
4. Specify Config Name.
5. Choose **Apache HTTP server** in the log type dropdown menu.
6. In the **Apache Log Format** section, paste your Apache HTTP server log format configuration. It is in the format of /etc/httpd/conf/httpd.conf and starts with LogFormat.

   For example:

   ```
   LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
   ```

7. (Optional) In the **Sample log parsing** section, paste a sample Apache HTTP server log to verify if the log parsing is successful.

   For example:

   ```
   127.0.0.1 - - [22/Dec/2021:06:48:57 +0000] "GET /xxx HTTP/1.1" 404 196 "-"
    "curl/7.79.1"
   ```

8. Choose **Create**.

Create a NGINX log config

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Resources**, choose **Log Config**.
3. Choose the **Create a log config** button.
4. Specify Config Name.
5. Choose **NGINX** in the log type dropdown menu.
6. In the **NGINX Log Format** section, paste your NGINX log format configuration. It is in the format of /etc/nginx/nginx.conf and starts with log_format.

   For example:

   ```
   log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
   '$status $body_bytes_sent "$http_referer" '
   '"$http_user_agent" "$http_x_forwarded_for"';
   ```

7. (Optional) In the **Sample log parsing** section, paste a sample NGINX log to verify if the log parsing is successful.

For example:

```
127.0.0.1 - - [24/Dec/2021:01:27:11 +0000] "GET / HTTP/1.1" 200 3520 "-"
 "curl/7.79.1" "-"
```

8. (Optional) In the **Filter** section, you add some conditions to filter logs at the log agent side. The solution will ingest logs that match ALL the specified conditions only.
9. Select **Create**.

Create a Syslog config

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Resources**, choose **Log Config**.
3. Choose the **Create a log config** button.
4. Specify Config Name.
5. Choose **Syslog** in the log type dropdown menu. Note that Centralized Logging with OpenSearch also supports Syslog with JSON format and single-line text format.

RFC5424

1. Paste a sample RFC5424 log. For example:

```
<35>1 2013-10-11T22:14:15Z client_machine su - - - 'su root' failed for joe on /dev/
pts/2
```

2. Choose **Parse Log**.
3. Check if each field type mapping is correct. You can change the type by selecting the dropdown menu in the second column. For all supported types, see Data Types.

> **Note**
>
> You must specify the datetime of the log using the key "time". If not specified, system time will be added.

4. Specify the **Time format**. The format syntax follows strptime. Check this manual for details. For example:

```
%Y-%m-%dT%H:%M:%SZ
```

5. (Optional) In the **Filter** section, you add some conditions to filter logs at the log agent side. The solution will ingest logs that match ALL the specified conditions only.
6. Select **Create**.

RFC3164

1. Paste a sample RFC3164 log. For example:

```
<35>Oct 12 22:14:15 client_machine su: 'su root' failed for joe on /dev/pts/2
```

2. Choose **Parse Log**.
3. Check if each field type mapping is correct. You can change the type by selecting the dropdown menu in the second column. For all supported types, see Data Types.

> **Note**
>
> Specify the datetime of the log using the key "time". If not specified, system time will be added. Since there is no year in the timestamp of RFC3164, it cannot be displayed as a time histogram in the Discover interface of Amazon OpenSearch Service.

4. Specify the **Time format**. The format syntax follows strptime. Check this for details. For example:

```
%b %m %H:%M:%S
```

5. (Optional) In the **Filter** section, you add some conditions to filter logs at the log agent side. The solution will ingest logs that match ALL the specified conditions only.
6. Select **Create**.

Custom

1. In the **Syslog Format** section, paste your Syslog log format configuration. It is in the format of /etc/rsyslog.conf and starts with template or $template. The format syntax follows Syslog Message Format. For example:

```
<%pri%>1 %timestamp:::date-rfc3339% %HOSTNAME% %app-name% %procid% %msgid% %msg%\n
```

2. In the **Sample log parsing** section, paste a sample NGINX log to verify if the log parsing is successful. For example:

```
<35>1 2013-10-11T22:14:15.003Z client_machine su - - 'su root' failed for joe on /
dev/pts/2
```

3. Check if each field type mapping is correct. Change the type by selecting the dropdown menu in the second column. For all supported types, see Data Types.

> **Note**
>
> Specify the datetime of the log using the key "time". If not specified, system time will be added.

4. Specify the **Time format**. The format syntax follows strptime. Check this manual for details.
5. (Optional) In the **Filter** section, you add some conditions to filter logs at the log agent side. The solution will ingest logs that match ALL the specified conditions only.
6. Select **Create**.

Create a Single-line text config

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Resources**, choose **Log Config**.
3. Choose the **Create a log config** button.
4. Specify Config Name.
5. Choose **Single-line Text** in the log type dropdown menu.
6. Write the regular expression in Rubular to validate first and enter the value. For example:

```
(?<remote_addr>\S+)\s*-\s*(?<remote_user>\S+)\s*\[(?<time_local>\d+/\S+/\d
+:\d+:\d+:\d+)\s+\S+\]\s*"(?<request_method>\S+)\s+(?<request_uri>\S+)\s+\S
+"\s*(?<status>\S+)\s*(?<body_bytes_sent>\S+)\s*"(?<http_referer>[^"]*)"\s*"(?
<http_user_agent>[^"]*)"\s*"(?<http_x_forwarded_for>[^"]*)".*
```

7. In the **Sample log parsing** section, paste a sample Single-line text log and choose **Parse log** to verify if the log parsing is successful. For example:

```
127.0.0.1 - - [24/Dec/2021:01:27:11 +0000] "GET / HTTP/1.1" 200 3520 "-"
 "curl/7.79.1" "-"
```

8. Check if each field type mapping is correct. Change the type by selecting the dropdown menu in the second column. For all supported types, see Data Types.

> **Note**
>
> Specify the datetime of the log using the key "time". If not specified, system time will be added.

9. Specify the **Time format**. The format syntax follows strptime. Check this manual for details.
10. (Optional) In the **Filter** section, you add some conditions to filter logs at the log agent side. The solution will ingest logs that match ALL the specified conditions only.
11. Select **Create**.

Create a Multi-line text config

1. Sign in to the Centralized Logging with OpenSearch Console.
2. In the left sidebar, under **Resources**, choose **Log Config**.
3. Choose the **Create a log config** button.
4. Specify Config Name.
5. Choose **Multi-line Text** in the log type dropdown menu.

Java - Spring Boot

1. For Java Spring Boot logs, you could provide a simple log format. For example:

```
%d{yyyy-MM-dd HH:mm:ss.SSS} %-5level [%thread] %logger : %msg%n
```

2. Paste a sample multi-line log. For example:

```
2022-02-18 10:32:26.400 ERROR [http-nio-8080-exec-1]
 org.apache.catalina.core.ContainerBase.[Tomcat].[localhost].[/].
[dispatcherServlet] : Servlet.service() for servlet [dispatcherServlet] in context
 with path [] threw exception [Request processing failed; nested exception is
 java.lang.ArithmeticException: / by zero] with root cause
```

```
java.lang.ArithmeticException: / by zero
   at com.springexamples.demo.web.LoggerController.logs(LoggerController.java:22)
   at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
   at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke
```

3. Choose **Parse Log**.
4. Check if each field type mapping is correct. You can change the type by selecting the dropdown menu in the second column. For all supported types, see Data Types.

> **Note**
>
> You must specify the datetime of the log using the key "time". If not specified, system time will be added.

5. Specify the **Time format**. The format syntax follows strptime. Check this for details.
6. (Optional) In the **Filter** section, you add some conditions to filter logs at the log agent side. The solution will ingest logs that match ALL the specified conditions only.
7. Select **Create**.

Custom

1. For other kinds of logs, you could specify the first line regex pattern. For example:

```
(?<time>\d{4}-\d{2}-\d{2}\s*\d{2}:\d{2}:\d{2}.\d{3})\s*(?<message>goroutine\s*\d\s*
\[.+\]:)
```

2. Paste a sample multi-line log. For example:

```
2023-07-12 10:32:26.400 goroutine 1 [chan receive]:
runtime.gopark(0x4739b8, 0xc420024178, 0x46fcd7, 0xc, 0xc420028e17, 0x3)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc420053e30 sp=0xc420053e00
 pc=0x42503c
runtime.goparkunlock(0xc420024178, 0x46fcd7, 0xc, 0x1000f010040c217, 0x3)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc420053e70 sp=0xc420053e30
 pc=0x42512e
runtime.chanrecv(0xc420024120, 0x0, 0xc420053f01, 0x4512d8)
  /usr/local/go/src/runtime/chan.go:506 +0x304 fp=0xc420053f20 sp=0xc420053e70
 pc=0x4046b4
runtime.chanrecv1(0xc420024120, 0x0)
```

```
  /usr/local/go/src/runtime/chan.go:388 +0x2b fp=0xc420053f50 sp=0xc420053f20
 pc=0x40439b
main.main()
  foo.go:9 +0x6f fp=0xc420053f80 sp=0xc420053f50 pc=0x4512ef
runtime.main()
  /usr/local/go/src/runtime/proc.go:185 +0x20d fp=0xc420053fe0 sp=0xc420053f80
 pc=0x424bad
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc420053fe8 sp=0xc420053fe0
 pc=0x44b4d1
```

3. Choose **Parse Log**.
4. Check if each field type mapping is correct. You can change the type by selecting the dropdown menu in the second column. For all supported types, see Data Types.

> **Note**
>
> You must specify the datetime of the log using the key "time". If not specified, system time will be added.

5. (Optional) In the **Filter** section, you add some conditions to filter logs at the log agent side. The solution will ingest logs that match ALL the specified conditions only.
6. Select **Create**.

Cross-account ingestion

Centralized Logging with OpenSearch supports ingesting AWS Service logs and Application logs in different AWS accounts within the same Region. After deploying Centralized Logging with OpenSearch in one account (main account), you can launch the CloudFormation stack in a different account (member account), and associate the two accounts (main account and member account) to implement cross-account ingestion.

Concepts

- **Main account**: One account in which you deployed the Centralized Logging with OpenSearch console. The OpenSearch clusters must also be in the same account.
- **Member account**: Another account from which you want to ingest AWS Service logs or application logs.

The CloudFormation stack in the member account has the least privileges. Centralized Logging with OpenSearch must provision some AWS resources in the member account to collect logs, and will assume an IAM role provisioned in the member account to list or create resources.

For more information, refer to the Architecture section.

Add a member account

Step 1. Launch a CloudFormation stack in the member account

1. Sign in to the Centralized Logging with OpenSearch console.
2. In the navigation pane, under **Resources**, choose **Member Accounts**.
3. Choose the **Link an Account** button. It displays the steps to deploy the CloudFormation stack in the member account.
4. Important
5. You must copy the template URL, which will be used later.
6. Go to the CloudFormation console of the member account.
7. Choose the Create stack button and choose With new resources (standard).
8. In the **Create stack** page, enter the template URL you have copied in the **Amazon S3 URL**.

9. Follow the steps to create the CloudFormation stack and wait until the CloudFormation stack is provisioned.
10. Go to the **Outputs** tab to check the parameters, which will be used in **Step 2**.

Step 2. Link a member account

1. Go back to the Centralized Logging with OpenSearch console.
2. (Optional) In the navigation panel, under **Resources**, choose **Member Accounts**.
3. In **Step 2. Link an account**, enter the parameters using the Outputs parameters from **Step 1**.

Parameter	CloudFormation Outputs	Description
Account Name	N/A	Name of the member account.
Account ID	N/A	12-digit AWS account ID.
Cross Account Role ARN	CrossAccountRoleARN	Centralized Logging with OpenSearch will assume this role to operate resources in the member account.
Fluent Bit Agent Installation Document	AgentInstallDocument	Centralized Logging with OpenSearch will use this SSM Document to install Fluent Bit agent on EC2 instances in the member account.
Fluent Bit Agent Configuration Document	AgentConfigDocument	Centralized Logging with OpenSearch will use this SSM Document to deliver Fluent Bit configuration to EC2 instances.
Fluent Bit Agent Installation Document for Windows	AgentInstallDocumentForWindows	Fluent Bit Agent Installation Configuration for Windows.

Parameter	CloudFormation Outputs	Description
Fluent Bit Agent Configuration Document for Windows	AnentConfigDocumentForWindows	Fluent Bit Agent Configuration Document.
Fluent Bit Status Check Document	AgentStatusCheckDocument	Status detection of Fluent Bit.
Cross Account S3 Bucket	CrossAccountS3Bucket	You can use the Centralized Logging with OpenSearch console to enable some AWS Service logs and output them to Amazon S3. The logs will be stored in this account.
Cross Account Stack ID	CrossAccountStackId	CloudFormation stack ID in the member account.
Cross Account KMS Key	CrossAccountKMSKeyARN	Centralized Logging with OpenSearch will use the AWS KMS key to encrypt Amazon SQS.

4. Click the **Link** button.

Pipeline alarms, monitoring, and logs

Pipeline alarms

There are different types of pipeline alarms: log processor alarms, buffer layer alarms, and source alarms (only for application log pipeline). The alarms will be triggered when the defined condition is met.

Log alarm type	Log alarm condition	Description
Log processor alarms	Error invocation # >= 10 for 5 minutes, 1 consecutive time	When the number of log processor Lambda error calls is greater than or equal to 10 within 5 minutes (including 5 minutes), an email alarm will be triggered.
Log processor alarms	Failed record # >= 1 for 1 minute, 1 consecutive time	When the number of failed records is greater than or equal to 1 within a 1-minute window, an alarm will be triggered.
Log processor alarms	Average execution duration in last 5 minutes >= 60000 milliseconds	In the last 5 minutes, when the average execution time of log processor Lambda is greater than or equal to 60 seconds, an email alarm will be triggered.
Buffer layer alarms	SQS Oldest Message Age >= 30 minutes	When the age of the oldest Amazon SQS message is greater than or equal to 30 minutes, it means that the message has not been consumed for at least 30

Log alarm type	Log alarm condition	Description
		minutes, an email alarm will be triggered.
Source alarms (only for application log pipeline)	Fluent Bit output_retried_record_total >= 100 for last 5 minutes	When the total number of retry records output by Fluent Bit in the past 5 minutes is greater than or equal to 100, an email alarm will be triggered.

You can choose to enable log alarms or disable them according to your needs.

Enable log alarms

1. Sign in to the Centralized Logging with OpenSearch console.
2. In the left navigation bar, under Log Analytics Pipelines, choose AWS Service Log or Application Log.
3. Select the log pipeline created and choose **View details**.
4. Select the **Alarm** tab.
5. Switch on **Alarms** if needed and select an existing SNS topic.
6. If you choose **Create a new SNS topic**, you must provide email address for the newly created SNS topic to notify.

Disable log alarms

1. Sign in to the Centralized Logging with OpenSearch console.
2. In the left navigation bar, under Log Analytics Pipelines, choose Application Log or Service Log.Select the log pipeline created and choose View details.
3. Select the Alarm tab.
4. Switch off Alarms.

Monitoring

The following types of metrics are available on the Centralized Logging with OpenSearch console.

Log source metrics

Fluent Bit

- FluentBitOutputProcRecords - The number of log records that this output instance has successfully sent. This is the total record count of all unique chunks sent by this output. If a record is not successfully sent, it does not count towards this metric.
- FluentBitOutputProcBytes - The number of bytes of log records that this output instance has successfully sent. This is the total byte size of all unique chunks sent by this output. If a record is not sent due to some error, then it will not count towards this metric.
- FluentBitOutputDroppedRecords - The number of log records that have been dropped by the output. This means they met an unrecoverable error or retries expired for their chunk.
- FluentBitOutputErrors - The number of chunks that have faced an error (either unrecoverable or retrievable). This is the number of times a chunk has failed, and does not correspond with the number of error messages you see in the Fluent Bit log output.
- FluentBitOutputRetriedRecords - The number of log records that experienced a retry. Note that this is calculated at the chunk level, and the count is increased when an entire chunk is marked for retry. An output plugin may or may not perform multiple actions that generate many error messages when uploading a single chunk.
- FluentBitOutputRetriesFailed - The number of times that retries expired for a chunk. Each plugin configures a Retry_Limit, which applies to chunks. Once the Retry_Limit has been reached for a chunk, it is discarded and this metric is incremented.
- FluentBitOutputRetries - The number of times this output instance requested a retry for a chunk.

Network Load Balancer

- SyslogNLBActiveFlowCount - The total number of concurrent flows (or connections) from clients to targets. This metric includes connections in the SYN_SENT and ESTABLISHED states. TCP connections are not terminated at the load balancer, so a client opening a TCP connection to a target counts as a single flow.
- SyslogNLBProcessedBytes - The total number of bytes processed by the load balancer, including TCP/IP headers. This count includes traffic to and from targets, minus health check traffic.

Buffer metrics

Log Buffer is a buffer layer between the Log Agent and OpenSearch clusters. The agent uploads logs into the buffer layer before being processed and delivered into the OpenSearch clusters. A buffer layer is a way to protect OpenSearch clusters from overwhelming.

Kinesis Data Stream

- KDSIncomingBytes - The number of bytes successfully put to the Kinesis stream over the specified time period. This metric includes bytes from PutRecord and PutRecords operations. Minimum, Maximum, and Average statistics represent the bytes in a single put operation for the stream in the specified time period.
- KDSIncomingRecords - The number of records successfully put to the Kinesis stream over the specified time period. This metric includes record counts from PutRecord and PutRecords operations. Minimum, Maximum, and Average statistics represent the records in a single put operation for the stream in the specified time period.
- KDSPutRecordBytes - The number of bytes put to the Kinesis stream using the PutRecord operation over the specified time period.
- KDSThrottledRecords - The number of records rejected due to throttling in a PutRecords operation per Kinesis data stream, measured over the specified time period.
- KDSWriteProvisionedThroughputExceeded - The number of records rejected due to throttling for the stream over the specified time period. This metric includes throttling from PutRecord and PutRecords operations. The most commonly used statistic for this metric is Average.

When the Minimum statistic has a non-zero value, records will be throttled for the stream during the specified time period.

When the Maximum statistic has a value of 0 (zero), no records will be throttled for the stream during the specified time period.

Amazon SQS

- SQSNumberOfMessagesSent - The number of messages added to a queue.
- SQSNumberOfMessagesDeleted - The number of messages deleted from the queue.

Amazon SQS emits the NumberOfMessagesDeleted metric for every successful deletion operation that uses a valid receipt handle, including duplicate deletions. The following scenarios might

cause the value of the NumberOfMessagesDeleted metric to be higher than expected: - Calling the DeleteMessage action on different receipt handles that belong to the same message: If the message is not processed before the visibility timeout expires, the message becomes available to other consumers that can process it and delete it again, increasing the value of the NumberOfMessagesDeleted metric.

- Calling the DeleteMessage action on the same receipt handle: If the message is processed and deleted, but you call the DeleteMessage action again using the same receipt handle, a success status is returned, increasing the value of the NumberOfMessagesDeleted metric.
- SQSApproximateNumberOfMessagesVisible - The number of messages available for retrieval from the queue.
- SQSApproximateAgeOfOldestMessage - The approximate age of the oldest non-deleted message in the queue.
- After a message is received three times (or more) and not processed, the message is moved to the back of the queue and the ApproximateAgeOfOldestMessage metric points at the second-oldest message that hasn't been received more than three times. This action occurs even if the queue has a redrive policy.
- Because a single poison-pill message (received multiple times but never deleted) can distort this metric, the age of a poison-pill message isn't included in the metric until the poison-pill message is consumed successfully.
- When the queue has a redrive policy, the message is moved to a dead-letter queue after the configured **Maximum Receives**. When the message is moved to the dead-letter queue, the ApproximateAgeOfOldestMessage metric of the dead-letter queue represents the time when the message was moved to the dead-letter queue (not the original time the message was sent).

Log processor metrics

The Log Processor Lambda is responsible for performing final processing on the data and bulk writing it to OpenSearch.

- TotalLogs - The total number of log records or events processed by the Lambda function.
- ExcludedLogs - The number of log records or events that were excluded from processing, which could be due to filtering or other criteria.
- LoadedLogs - The number of log records or events that were successfully processed and loaded into OpenSearch.

- FailedLogs - The number of log records or events that failed to be processed or loaded into OpenSearch.
- ConcurrentExecutions - The number of function instances that are processing events. If this number reaches your concurrent executions quota for the Region, or the reserved concurrency limit on the function, then Lambda throttles additional invocation requests.
- Duration - The amount of time that your function code spends processing an event. The billed duration for an invocation is the value of Duration rounded up to the nearest millisecond.
- Throttles - The number of invocation requests that are throttled. When all function instances are processing requests and no concurrency is available to scale up, Lambda rejects additional requests with a TooManyRequestsException error. Throttled requests and other invocation errors don't count as either Invocations or Errors.
- Invocations - The number of times that your function code is invoked, including successful invocations and invocations that result in a function error. Invocations aren't recorded if the invocation request is throttled or otherwise results in an invocation error. The value of Invocations equals the number of requests billed.

Frequently asked questions

General

Q: What is Centralized Logging with OpenSearch solution?

Centralized Logging with OpenSearch is an AWS Solution that simplifies the building of log analytics pipelines. It provides to customers, as complementary of Amazon OpenSearch Service, capabilities to ingest and process both application logs and AWS service logs without writing code, and create visualization dashboards from templates. Centralized Logging with OpenSearch automatically assembles the underlying AWS services, and provides you with a web console to manage log analytics pipelines.

Q: What are the supported logs in this solution?

Centralized Logging with OpenSearch supports both AWS service logs and EC2/EKS application logs. Refer to the supported AWS services, and the supported application log formats and sources for more details.

Q: Does Centralized Logging with OpenSearch support ingesting logs from multiple AWS accounts?

Yes. Centralized Logging with OpenSearch supports ingesting AWS service logs and application logs from a different AWS account in the same Region. For more information, see cross-account ingestion.

Q: Does Centralized Logging with OpenSearch support ingesting logs from multiple AWS Regions?

Currently, Centralized Logging with OpenSearch does not automate the log ingestion from a different AWS Region. You must ingest logs from other Regions into pipelines provisioned by Centralized Logging with OpenSearch. For AWS services that store the logs in S3 bucket, you can use the S3 Cross-Region Replication to copy the logs to the Centralized Logging with OpenSearch deployed Region, and import incremental logs using the manual mode by specifying the log location in the S3 bucket. For application logs on EC2 and EKS, you must set up the networking (for example, Kinesis VPC endpoint, VPC Peering), install agents, and configure the agents to ingest logs to Centralized Logging with OpenSearch pipelines.

Q: What is the license of this solution?

This solution is provided under the Apache-2.0 license. It is a permissive free software license written by the Apache Software Foundation. It allows users to use the software for any purpose, to distribute it, to modify it, and to distribute modified versions of the software under the terms of the license, without concern for royalties.

Q: How can I find the roadmap of this solution?

This solution uses a GitHub project to manage the roadmap. You can find the roadmap here.

Q: How can I submit a feature request or bug report? You can submit feature requests and bug reports through the GitHub issues. Here are the templates for feature request, bug report.

Setup and configuration

Q: Can I deploy Centralized Logging with OpenSearch on AWS in any AWS Region?

Centralized Logging with OpenSearch provides two deployment options: option 1 with Amazon Cognito User Pool, and option 2 with OpenID Connect. For option 1, customers can deploy the solution in AWS Regions where Amazon Cognito User Pool, AWS AppSync, Amazon Data Firehose (optional) are available. For option 2, customers can deploy the solution in AWS Regions where AWS AppSync, Amazon Data Firehose (optional) are available. Refer to supported Regions for deployment for more information.

Q: What are the prerequisites of deploying this solution?

Centralized Logging with OpenSearch does not provision Amazon OpenSearch Service clusters, and you must import existing OpenSearch clusters through the web console. The clusters must meet the requirements specified in the prerequisites.

Q: Why do I need a domain name with ICP recordal when deploying the solution in AWS China Regions?

The Centralized Logging with OpenSearch console is served via the CloudFront distribution, which is considered as an internet information service. According to the local regulations, any internet information service must bind to a domain name with ICP recordal.

Q: What versions of OpenSearch does the solution work with?

Centralized Logging with OpenSearch supports Amazon OpenSearch Service, with OpenSearch 1.3 or later.

Q: What is the index name rules for OpenSearch created by the Log Analytics Pipeline?

You can change the index name if needed when using the Centralized Logging with OpenSearch console to create a log analytics pipeline.

If the log analytics pipeline is created for service logs, the index name is composed of <Index Prefix>-<service-type>-<Index Suffix>-<00000x>, where you can define a name for Index Prefix and service-type is automatically generated by the solution according to the service type you have chosen. Moreover, you can choose different index suffix types to adjust the index rollover time window.

- YYYY-MM-DD-HH: Amazon OpenSearch Service will roll the index by the hour.
- YYYY-MM-DD: Amazon OpenSearch Service will roll the index by 24 hours.
- YYYY-MM: Amazon OpenSearch Service will roll the index by 30 days.
- YYYY: Amazon OpenSearch Service will roll the index by 365 days.

It should be noted that in OpenSearch, the time is in the UTC 0 time zone.

Regarding the 00000x part, Amazon OpenSearch Service will automatically append a 6-digit suffix to the index name, where the first index rule is 000001, rollover according to the index, and increment backwards, such as 000002, 000003.

If the log analytics pipeline is created for application log, the index name is composed of <Index Prefix>-<Index Suffix>-<00000x>. The rules for index prefix and index suffix, 00000x are the same as those for service logs.

Q: What is the index rollover rules for OpenSearch created by the Log Analytics Pipeline?

Index rollover is determined by two factors. One is the Index Suffix in the index name. If you enable the index rollover by capacity, Amazon OpenSearch Service will roll your index when the index capacity equals or exceeds the specified size, regardless of the rollover time window. Note that if one of these two factors matches, index rollover can be triggered.

For example, we created an application log pipeline on January 1, 2023, deleted the application log pipeline at 9:00 on January 4, 2023, and the index name is nginx-YYYY-MM-DD-<00000x>. At the same time, we enabled the index rollover by capacity and entered 300GB. If the log data volume increases suddenly after creation, it can reach 300GB every hour, and the duration is 2 hours and 10 minutes. After that, it returns to normal, and the daily data volume is 90GB. Then OpenSearch creates three indexes on January 1, the index names are nginx-2023-01-01-000001,

nginx-2023-01-01-000002, nginx-2023-01-01-000003, and then creates one every day Indexes respectively: nginx-2023-01-02-000004, nginx-2023-01-03-000005, nginx-2023-01-04-000006.

Q: Can I deploy the solution in an existing VPC?

Yes. You can either launch the solution with a new VPC or launch the solution with an existing VPC. When using an existing VPC, you must select the VPC and the corresponding subnets. Refer to launch with Amazon Cognito User Pool or launch with OpenID Connect for more details.

Q: How can I change the default CIDR of the solution?

To change the default CIDR of the solution, you must first create a custom CIDR VPC and then deploy CLO using the 'Existing VPC' template.

You may use the following AWS CloudFormation templates to do so. First, deploy this CloudFormation template and when prompted, enter the CIDR prefix you want for your VPC. After the CloudFormation stack has finished deploying, continue with the deployment of the CLO solution using the Launch with an existing VPC option.

Q: I did not receive the email containing the temporary password when launching the solution with Amazon Cognito User Pool. How can I resend the password?

Your account is managed by the Amazon Cognito User Pool. To resend the temporary password, you can find the user pool created by the solution, and delete and recreate the user using the same email address. If you still have the same issue, try with another email address.

Q: How can I create more users for this solution?

If you launched the solution with Amazon Cognito User Pool, go to the AWS Management Console, find the user pool created by the solution, and you can create more users. If you launched the solution with OpenID Connect (OIDC), you should add more users in the user pool managed by the OIDC provider. Note that all users have the same permissions.

Pricing

Q: How will I be charged and billed for the use of this solution?

You are responsible for the cost of AWS services used while running this solution. You pay only for what you use, and there are no minimum or setup fees. Refer to the Centralized Logging with OpenSearch Cost section for detailed cost estimation.

Q: Will there be additional costs for cross-account ingestion?

No. The cost will be the same as ingesting logs within the same AWS account.

Log Ingestion

Q: What is the log agent used in the Centralized Logging with OpenSearch solution? Centralized Logging with OpenSearch uses AWS for Fluent Bit, a distribution of Fluent Bit maintained by AWS. The solution uses this distribution to ingest logs from Amazon EC2 and Amazon EKS.

Q: I have already stored the AWS service logs of member accounts in a centralized logging account. How should I create service log ingestion for member accounts?

In this case, you must deploy the Centralized Logging with OpenSearch solution in the centralized logging account, and ingest AWS service logs using the *Manual* mode from the logging account. Refer to this guide for ingesting Application Load Balancer logs with *Manual* mode. You can do the same with other supported AWS services that output logs to Amazon S3.

Q: Why are there some duplicated records in OpenSearch when ingesting logs via Kinesis Data Streams?

This is usually because there is no enough Kinesis Shards to handle the incoming requests. When a threshold error occurs in Kinesis, the Fluent Bit agent will retry that chunk. To avoid this issue, you must estimate your log throughput, and set a proper Kinesis shard number. Refer to the Kinesis Data Streams quotas and limits. Centralized Logging with OpenSearch provides a built-in feature to scale-out and scale-in the Kinesis shards, and it would take a couple of minutes to scale out to the desired number.

Q: How to install a log agent on CentOS 7?

Refer to Create Instance Group for CentOS 7.

Log Visualization

Q: How can I find the built-in dashboards in OpenSearch?

Refer to the AWS Service Logs and Application Logs to find out if there is a built-in dashboard supported. You also must turn on the *Sample Dashboard* option when creating a log analytics pipeline. The dashboard will be inserted into the Amazon OpenSearch Service under **Global**

Tenant. You can switch to the Global Tenant from the top right coder of the OpenSearch Dashboards.

Troubleshooting

This section provides troubleshooting instructions for deploying and using the solution.

If these instructions don't address your issue, see the Contact AWS Support section for instructions on opening an AWS Support case for this solution.

Error: Failed to assume service-linked role arn:x:x:x:/AWSServiceRoleForAppSync

The reason for this error is that the account has never used the AWS AppSync service. You can deploy the solution's CloudFormation template again. AWS has already created the role automatically when you encountered the error.

You can also go to AWS CloudShell or the local terminal and run the following AWS CLI command to Link AWS AppSync Role

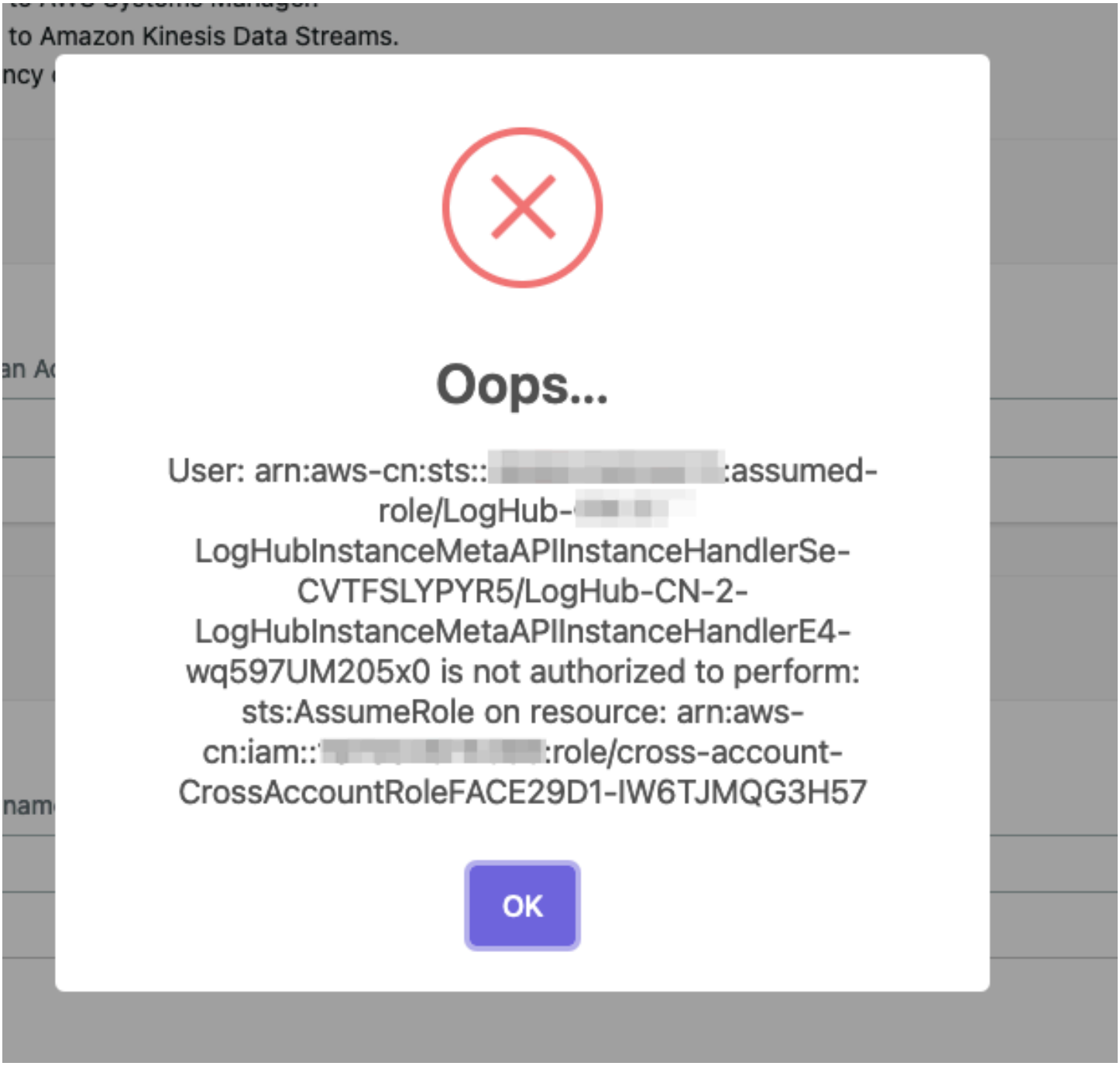
```
aws iam create-service-linked-role --aws-service-name appsync.amazonaws.com
```

Error: Unable to add backend

Centralized Logging with OpenSearch only supports Amazon OpenSearch Service domain with fine-grained access control enabled. You must go to the Amazon OpenSearch Service console, and edit the **Access policy** for the Amazon OpenSearch Service domain.

Error: User xxx is not authorized to perform sts:AssumeRole on resource

Message labeled Oops, user is not authorized to perform sts:AssumeRole on resource.



If you see this error, make sure you have entered the correct information during cross account setup, and then wait for several minutes.

Centralized Logging with OpenSearch uses AssumeRole for cross-account access. This is the best practice to temporarily access the AWS resources in your member account. However, these roles created during cross account setup take seconds or minutes to be affective.

Error: PutRecords API responded with error='InvalidSignatureException'

Fluent-bit agent reports PutRecords API responded with error='InvalidSignatureException', message='The request signature we calculated does not match the signature you provided. Check your AWS Secret Access Key and signing method. Consult the service documentation for details.'

Restart the fluent-bit agent. For example, on EC2 with Amazon Linux2, run the following command:

```
sudo service fluent-bit restart
```

Error: PutRecords API responded with error='AccessDeniedException'

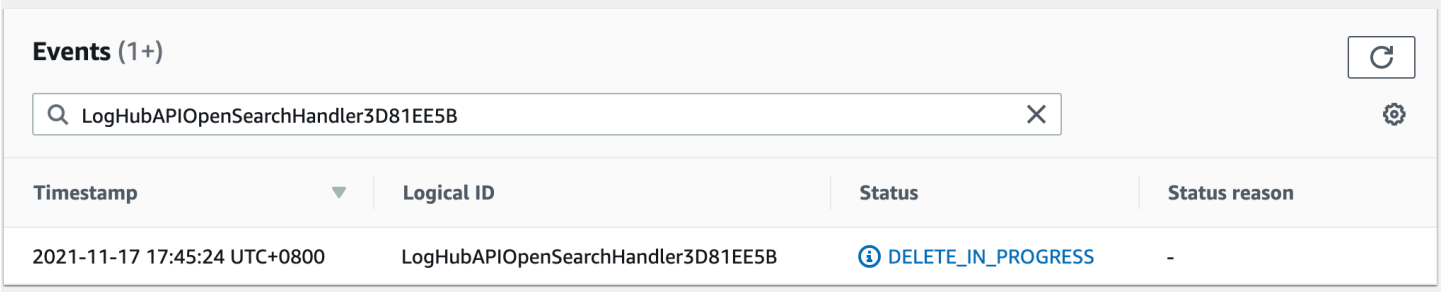
Fluent-bit agent deployed on EKS Cluster reports "AccessDeniedException" when sending records to Kinesis. Verify that the IAM role trust relations are correctly set. With the Centralized Logging with OpenSearch console:

1. Open the Centralized Logging with OpenSearch console.
2. In the left sidebar, under **Log Source**, choose **EKS Clusters**.
3. Choose the **EKS Cluster** that you want to check.
4. Choose the **IAM Role ARN**, which will open the IAM Role in the AWS Management Console.
5. Choose the **Trust relationships** to verify that the OIDC Provider, the service account namespace, and conditions are correctly set.

You can get more information from Amazon EKS IAM role configuration.

CloudFormation stack is stuck on deleting an AWS::Lambda::Function resource when I update the stack

Event with status of DELETE_IN_PROGRESS.



The Lambda function resides in a VPC, and you must wait for the associated elastic network interface resource to be deleted.

The agent status is offline after I restart the EC2 instance

This usually happens if you have installed the log agent, but restart the instance before you create any Log Ingestion. The log agent will auto restart if there is at least one Log Ingestion. If you have a log ingestion, but the problem still exists, you can use systemctl status fluent-bit to check its status inside the instance.

Switched to Global tenant and can't find the dashboard in OpenSearch

This is usually because Centralized Logging with OpenSearch received a 403 error from OpenSearch when creating the index template and dashboard. This can be fixed by re-running the Lambda function manually by following the following steps:

With the Centralized Logging with OpenSearch console:

1. Open the Centralized Logging with OpenSearch console, and find the AWS Service Log pipeline that has this issue.
2. Copy the first 5 characters from the ID section. For example, you should copy c169c from ID c169cb23-88f3-4a7e-90d7-4ab4bc18982c
3. Go to AWS Management Console > Lambda. Paste in the function filters. This will filter in all the Lambda functions created for this AWS Service Log ingestion.
4. Click the Lambda function whose name contains "OpenSearchHelperFn".
5. In the **Test** tab, create a new event with any Event name.
6. Click the **Test** button to trigger the Lambda, and wait for the Lambda function to complete.

7. The dashboard should be available in OpenSearch.

Error from Fluent-bit agent: version `GLIBC_2.25' not found

Refer to Fix version GLIBC_2.25 not found issue.

Contact AWS Support

If you have AWS Business Support+, AWS Enterprise Support, or Unified Operations, you can use the AWS Support Center to get expert assistance with this solution. The following sections provide instructions.

Create case

1. Sign in to Support Center.
2. Choose **Create case**.

How can we help?

1. Choose **Technical**.
2. For **Service**, select **Solutions**.
3. For **Category**, select **Other Solutions**.
4. For **Severity**, select the option that best matches your use case.
5. When you enter the **Service**, **Category**, and **Severity**, the interface populates links to common troubleshooting questions. If you can't resolve your question with these links, choose **Next step: Additional information**.

Additional information

1. For **Subject**, enter text summarizing your question or issue.
2. For **Description**, describe the issue in detail, including the name of this solution and the version you are using, such as this example: **Centralized Logging with OpenSearch vX.Y.Z**.
3. Choose **Attach files**.
4. Attach the information that AWS Support needs to process the request.

Help us resolve your case faster

1. Enter the requested information.
2. Choose **Next step: Solve now or contact us**.

Solve now or contact us

1. Review the **Solve now** solutions.
2. If you can’t resolve your issue with these solutions, choose **Contact us**, enter the requested information, and choose **Submit**.

Upgrade the solution

Time to upgrade: Approximately 20 minutes

> **Important**
>
> Important The following upgrade documentation only supports Centralized Logging with OpenSearch version 2.x and later. If you are using older versions, such as v1.x or any version of Log Hub, refer to the Discussions on GitHub.

Step 1. Update the CloudFormation stack

1. Go to the AWS CloudFormation console.
2. Select the Centralized Logging with OpenSearch main stack, and click the **Update** button.
3. Choose **Replace current template**, and enter the specific **Amazon S3 URL** according to your initial deployment type. Refer to Deployment Overview for more details.

Launch with Amazon Cognito User Pool

Type	Link
Launch with a new VPC	link:https://solutions-reference.s3.amazonaws.com/centralized-logging-with-opensearch/latest/CentralizedLogging.template
Launch with an existing VPC	link:https://solutions-reference.s3.amazonaws.com/centralized-logging-with-opensearch/latest/CentralizedLoggingFromExistingVPC.template

Launch with OpenID Connect (OIDC)

Type	Link
Launch with a new VPC in AWS Regions	link:https://solutions-reference.s3.amazonaws.com/centralized-logging-with-opensearch/latest/CentralizedLoggingWithOIDC.template
Launch with an existing VPC in AWS Regions	link:https://solutions-reference.s3.amazonaws.com/centralized-logging-with-opensearch/latest/CentralizedLoggingFromExistingVPCWithOIDC.template
Launch with a new VPC in AWS China Regions	link:https://solutions-reference.s3.amazonaws.com/centralized-logging-with-opensearch/latest/CentralizedLoggingWithOIDC.template
Launch with an existing VPC in AWS China Regions	link:https://solutions-reference.s3.amazonaws.com/centralized-logging-with-opensearch/latest/CentralizedLoggingFromExistingVPCWithOIDC.template

1. Under **Parameters**, review the parameters for the template and modify them as necessary.
2. Choose **Next**.
3. On Configure stack options page, choose Next.
4. On Review page, review and confirm the settings. Check the box: I acknowledge that AWS CloudFormation might create IAM resources.
5. Choose **Update 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 **UPDATE_COMPLETE** status in approximately 15 minutes.

Step 2. Refresh the web console

Now you have completed all the upgrade steps. Choose the refresh button in your browser.

Uninstall the solution

You will encounter an IAM role missing error if you delete the Centralized Logging with OpenSearch main stack before you delete the log pipelines. Centralized Logging with OpenSearch console launches additional CloudFormation stacks to ingest logs. If you want to uninstall the Centralized Logging with OpenSearch solution. We recommend you to delete log pipelines (incl. AWS Service log pipelines and application log pipelines) before uninstalling the solution.

Step 1. Delete Application Log Pipelines

> **Important**
>
> Delete all the log ingestion before deleting an application log pipeline.

1. Go to the Centralized Logging with OpenSearch console, in the left sidebar, choose **Application Log**.
2. Click the application log pipeline to view details.
3. In the ingestion tab, delete all the application log ingestion in the pipeline.
4. Uninstall/Disable the Fluent Bit agent.
   - EC2 (Optional): after removing the log ingestion from Instance Group. Fluent Bit will automatically stop ship logs, it is optional for you to stop the Fluent Bit in your instances. Here are the commands for stopping Fluent Bit agent.

     ```
     sudo service fluent-bit stop
        sudo systemctl disable fluent-bit.service
     ```

   - EKS DaemonSet (Mandatory): if you have chosen to deploy the Fluent Bit agent using DaemonSet, you must delete your Fluent Bit agent. Otherwise, the agent will continue to ship logs to Centralized Logging with OpenSearch pipelines.

     ```
     kubectl delete -f ~/fluent-bit-logging.yaml
     ```

   - EKS SideCar (Mandatory): remove the fluent-bit agent in your .yaml file, and restart your pod.
5. Delete the Application Log pipeline.
6. Repeat step 2 to Step 5 to delete all your application log pipelines.

Step 2. Delete AWS Service Log Pipelines

1. Go to the Centralized Logging with OpenSearch console, in the left sidebar, choose **AWS Service Log**.
2. Select and delete the AWS Service Log Pipeline one by one.

Step 3. Clean up imported OpenSearch domains

1. Delete Access Proxy, if you have created the proxy using Centralized Logging with the OpenSearch console.
2. Delete Alarms, if you have created alarms using Centralized Logging with the OpenSearch console.
3. Delete VPC peering connection between Centralized Logging with OpenSearch's VPC and OpenSearch's VPC.
   a. Go to Amazon VPC Console.
   b. Choose **Peering connections** in the left sidebar.
   c. Find and delete the VPC peering connection between the Centralized Logging with OpenSearch's VPC and OpenSearch's VPC. You may not have Peering Connections if you did not use the "Automatic" mode when importing OpenSearch domains.
4. (Optional) Remove imported OpenSearch Domains. (This will not delete the Amazon OpenSearch Service domain in the AWS account.)

Step 4. Delete Centralized Logging with OpenSearch stack

1. Go to the CloudFormation console.
2. Find CloudFormation Stack of the Centralized Logging with OpenSearch solution.
3. (Optional) Delete S3 buckets created by Centralized Logging with OpenSearch.

> **Important**
>
> The S3 bucket whose name contains **LoggingBucket** is the centralized bucket for your AWS service log. You might have enabled AWS Services to send logs to this S3 bucket. Deleting this bucket will cause AWS Services failed to send logs.

+ .. Choose the CloudFormation stack of the Centralized Logging with OpenSearch solution, and select the **Resources** tab. .. In the search bar, enter AWS::S3::Bucket. This will show all the S3 buckets created by Centralized Logging with OpenSearch solution, and the **Physical ID** field is the S3 bucket name. .. Go to the Amazon S3 console, and find the S3 bucket using the bucket name. **Empty** and **Delete** the S3 bucket.

4. Delete the CloudFormation Stack of the Centralized Logging with OpenSearch solution.

Additional resources

Grafana

This section introduces how to set up a Grafana environment. If you want the solution to generate dashboards in Grafana automatically, you must perform the following deployment. If you only want to store the data in Amazon S3 without creating dashboards, you can skip this section.

Step 1: Install Grafana

> **Note**
>
> Skip this step if you already have a Grafana environment.

Prerequisite:

An EC2 instance has been launched, supporting both x86 and ARM architecture.

The following steps provide an example using m6g.medium instance type, ARM architecture, and Amazon 2023. For more details, refer to Install Grafana.

```
# Edit/etc/yum.repos.d/grafana.repo file#input below content
[grafana]
name=grafana
baseurl=https://rpm.grafana.com
repo_gpgcheck=1
enabled=1
gpgcheck=1
gpgkey=https://rpm.grafana.com/gpg.key
sslverify=1
sslcacert=/etc/pki/tls/certs/ca-bundle.crt

# install grafana
yum install -y grafana

# Start grafana#and check its running status
systemctl start grafana-server
systemctl status grafana-server
```

```
# grafana listens on port 3000 by default, Users can edit /etc/grafana/grafana.ini to
 modify the configuration

# Acccess grafana#using the default credentials admin / admin#you will be promoted to
 change the password on the first login.
http://{instance-ip}:3000/

# If you need public access, please configure an Application Load Balancer (ALB) on
 your own.
# When configuring the ALB, modify the Idle timeout to 1800 to avoid the following
 error during large data queries (when a single API call exceeds 60 seconds)#
# "a padding to disable MSIE and Chrome friendly error page"
```

Step 2: Authorize the EC2 where Grafana is located to access Athena

Prerequisites:

- You have deployed Grafana on EC2.
- EC2 has been configured with an IAM Instance Profile. You must record the corresponding **role ARN** of the Instance Profile.

Follow these steps:

1. Access IAM Management Console.
2. Search for "AthenaPublicAccessRole" and choose it to access the details page. Record the role ARN, which will be used later.
3. Choose the **Trust relationships** tab.
4. Choose **Edit trust policy**.
5. Choose **Add** next to **Add a principal**.
6. Select **IAM Roles** from the **Principal type** dropdown list.
7. Enter the role ARN that you recorded in Step 2.
8. Choose **Add principal**.
9. Choose **update policy**.

Step 3: Install Amazon Athena plugins

Prerequisites:

- Grafana is installed.
- Grafana is accessible over the public network.

Follow these steps:

1. Access the Grafana console.
2. Select **Administration** from the left navigation pane, and then choose **Plugins**.
3. Select **All** in the **State** section on the right side.
4. In the search box, enter Athena and choose the* Amazon Athena* to access the details page.
5. Choose **Install** on the page and wait for the plugin installation to complete.

Step 4: Create service accounts

Follow these steps:

1. Access the Grafana console.
2. Select **Administration** from the left navigation pane, and then choose **Service accounts**.
3. Select **Add service account**.
4. Enter a display name. For example, "johndoe".
5. Select the role as Admin.
6. Choose **Create**.
7. Choose **Add service account token**.
8. Choose **Generate token**.
9. Choose **Copy to clipboard and close**.

10Save and record this token, which will be used when you must create a pipeline.

OpenSSL 1.1 Installation

Centralized Logging with OpenSearch uses Fluent Bit as the log agent, which requires OpenSSL 1.1 or later. You can install the dependency according to your operating system (OS). It is recommended to make your own AMI with OpenSSL 1.1 installed.

Important

Important If your OS is not listed in the following sections, you can follow the official installation guide to install OpenSSL.

Amazon Linux 2

```
sudo yum install openssl11
```

Ubuntu

22.04

```
ln -s /usr/lib/x86_64-linux-gnu/libsasl2.so.2 /usr/lib/libsasl2.so.3
ln -s /snap/core18/current/usr/lib/x86_64-linux-gnu/libssl.so.1.1 /usr/lib/
libssl.so.1.1
ln -s /snap/core18/current/usr/lib/x86_64-linux-gnu/libcrypto.so.1.1 /usr/lib/
libcrypto.so.1.1
```

20.04

```
ln -s /usr/lib/x86_64-linux-gnu/libsasl2.so.2 /usr/lib/libsasl2.so.3
```

18.04

```
ln -s /usr/lib/x86_64-linux-gnu/libsasl2.so.2 /usr/lib/libsasl2.so.3
```

Debian

GNU/10

```
ln -s /usr/lib/x86_64-linux-gnu/libsasl2.so.2 /usr/lib/libsasl2.so.3
```

GNU/11

```
ln -s /usr/lib/x86_64-linux-gnu/libsasl2.so.2 /usr/lib/libsasl2.so.3
```

Red Hat Enterprise Linux

8.X

OpenSSL 1.1 is installed by default.

7.X

```
sudo su -

yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/
linux_amd64/amazon-ssm-agent.rpm

systemctl enable amazon-ssm-agent
systemctl start amazon-ssm-agent

yum install -y wget perl unzip gcc zlib-devel
mkdir /tmp/openssl
cd /tmp/openssl
wget https://www.openssl.org/source/openssl-1.1.1s.tar.gz
tar xzvf openssl-1.1.1s.tar.gz
cd openssl-1.1.1s
./config --prefix=/usr/local/openssl11 --openssldir=/usr/local/openssl11 shared zlib
make
make install

echo /usr/local/openssl11/lib/ >> /etc/ld.so.conf
ldconfig
```

SUSE Linux Enterprise Server

15

OpenSSL 1.1 is installed by default.

Create Instance Group for CentOS 7

> **Note**
>
> Note: CentOS Linux 7 will reach end of life (EOL) on June 30, 2024. It is not an OS well tested with this solution. Consider this guide as a reference only.

1. Log in to your CentOS 7 machine and install SSM Agent manually.

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/
linux_amd64/amazon-ssm-agent.rpm
sudo systemctl enable amazon-ssm-agent
sudo systemctl start amazon-ssm-agent
```

2. Go to the **Instance Group** panel of Centralized Logging with OpenSearch console, create **Instance Group**, select the CentOS 7 machine, choose **Install log agent**, and wait for its status to be **offline**.
3. Log in to CentOS 7 and install fluent-bit 1.9.3 manually.

```
export RELEASE_URL=${FLUENT_BIT_PACKAGES_URL:-https://packages.fluentbit.io}
export RELEASE_KEY=${FLUENT_BIT_PACKAGES_KEY:-https://packages.fluentbit.io/
fluentbit.key}

sudo rpm --import $RELEASE_KEY
cat << EOF | sudo tee /etc/yum.repos.d/fluent-bit.repo
[fluent-bit]
name = Fluent Bit
baseurl = $RELEASE_URL/centos/VERSION_ARCH_SUBSTR
gpgcheck=1
repo_gpgcheck=1
gpgkey=$RELEASE_KEY
enabled=1
EOF
sudo sed -i 's|VERSION_ARCH_SUBSTR|\$releasever/\$basearch/|g' /etc/yum.repos.d/
fluent-bit.repo
sudo yum install -y fluent-bit-1.9.3-1

# Modify the configuration file
sudo sed -i 's/ExecStart.*/ExecStart=\/opt\/fluent-bit\/bin\/fluent-bit -c \/opt\/
fluent-bit\/etc\/fluent-bit.conf/g' /usr/lib/systemd/system/fluent-bit.service
sudo systemctl daemon-reload
sudo systemctl enable fluent-bit
sudo systemctl start fluent-bit
```

4. Go back to the **Instance Groups** panel of the Centralized Logging with OpenSearch console and wait for the CentOS 7 machine status to be **Online** and proceed to create the instance group.

Upload SSL Certificate to IAM

Upload the SSL certificate by running the AWS CLI command upload-server-certificate similar to the following:

```
aws iam upload-server-certificate --path /cloudfront/ \
--server-certificate-name YourCertificate \
--certificate-body file://Certificate.pem \
--certificate-chain file://CertificateChain.pem \
--private-key file://PrivateKey.pem
```

Replace the file names and Your Certificate with the names for your uploaded files and certificate. Specify the file:// prefix in the certificate-body, certificate-chain, and private key parameters in the API request. Otherwise, the request fails with a MalformedCertificate: Unknown error message.

> **Note**
>
> You must specify a path using the --path option. The path must begin with /cloudfront and must include a trailing slash (for example, /cloudfront/test/).

After the certificate is uploaded, the AWS command upload-server-certificate returns metadata for the uploaded certificate, including the certificate's Amazon Resource Name (ARN), friendly name, identifier (ID), and expiration date.

To view the uploaded certificate, run the AWS CLI command list-server-certificates:

```
aws iam list-server-certificates
```

For more information, see uploading a server certificate to IAM.

Fix version `GLIBC_2.25' not found issue

This error is caused by the old version of glibc. Centralized Logging with OpenSearch with a version later than 1.2 requires glibc-2.25 or above. So you must upgrade the existing version in EC2 first. The upgrade command for different kinds of OS is shown as follows:

> **Important**
>
> We strongly recommend you run the commands with environments first. Any upgrade failure may cause severe loss.

Redhat 7.9

For Redhat 7.9, the whole process will take about 2 hours, and at least 10 GB storage is needed.

```
# install library
yum install -y gcc gcc-c++ m4 python3 bison  fontconfig-devel  libXpm-devel texinfo
 bzip2 wget
echo /usr/local/lib  >> /etc/ld.so.conf

# create tmp directory
mkdir -p /tmp/library
cd /tmp/library

# install gmp-6.1.0
wget https://ftp.gnu.org/gnu/gmp/gmp-6.1.0.tar.bz2
tar xjvf gmp-6.1.0.tar.bz2
cd gmp-6.1.0
./configure --prefix=/usr/local
make && make install
ldconfig
cd ..

# install mpfr-3.1.4
wget https://gcc.gnu.org/pub/gcc/infrastructure/mpfr-3.1.4.tar.bz2
tar xjvf mpfr-3.1.4.tar.bz2
cd mpfr-3.1.4
./configure --with-gmp=/usr/local --prefix=/usr/local
make && make install
ldconfig
cd ..

# install mpc-1.0.3
wget https://gcc.gnu.org/pub/gcc/infrastructure/mpc-1.0.3.tar.gz
tar xzvf mpc-1.0.3.tar.gz
cd mpc-1.0.3
./configure --prefix=/usr/local
make && make install
```

```
ldconfig
cd ..

# install gcc-9.3.0
wget https://ftp.gnu.org/gnu/gcc/gcc-9.3.0/gcc-9.3.0.tar.gz
tar xzvf gcc-9.3.0.tar.gz
cd gcc-9.3.0
mkdir build
cd build/
../configure --enable-checking=release --enable-language=c,c++ --disable-multilib --
prefix=/usr
make -j4 && make install
ldconfig
cd ../..

# install make-4.3
wget https://ftp.gnu.org/gnu/make/make-4.3.tar.gz
tar xzvf make-4.3.tar.gz
cd make-4.3
mkdir build
cd build
../configure --prefix=/usr
make && make install
cd ../..

# install glibc-2.31
wget https://ftp.gnu.org/gnu/glibc/glibc-2.31.tar.gz
tar xzvf glibc-2.31.tar.gz
cd glibc-2.31
mkdir build
cd build/
../configure  --prefix=/usr --disable-profile --enable-add-ons --with-headers=/usr/
include --with-binutils=/usr/bin --disable-sanity-checks --disable-werror
make all && make install
make localedata/install-locales

# clean tmp directory
cd /tmp
rm -rf /tmp/library
```

Ubuntu 22

```
sudo ln -s /snap/core20/1623/usr/lib/x86_64-linux-gnu/libcrypto.so.1.1 /usr/lib/x86_64-
linux-gnu/libcrypto.so.1.1
sudo ln -s /snap/core20/1623/usr/lib/x86_64-linux-gnu/libssl.so.1.1 /usr/lib/x86_64-
linux-gnu/libssl.so.1.1
sudo ln -s /usr/lib/x86_64-linux-gnu/libsasl2.so.2 /usr/lib/libsasl2.so.3
```

Amazon Linux 2023

```
sudo su -

yum install -y wget perl unzip gcc zlib-devel
mkdir /tmp/openssl
cd /tmp/openssl
wget https://www.openssl.org/source/openssl-1.1.1s.tar.gz
tar xzvf openssl-1.1.1s.tar.gz
cd openssl-1.1.1s
./config --prefix=/usr/local/openssl11 --openssldir=/usr/local/openssl11 shared zlib
make
make install

echo /usr/local/openssl11/lib/ >> /etc/ld.so.conf
ldconfig
```

Developer guide

This section provides the source code for the solution.

Source code

Visit our GitHub repository to download the source code for this solution. The solution template is generated using the AWS Cloud Development Kit (AWS CDK). Refer to the README.md file for additional information.

Reference

This section includes information about an optional feature for collecting anonymized 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 invoked, the following information is collected and sent to AWS:

- **Solution ID** - The AWS solution identifier
- **Unique ID (UUID)** - Randomly generated, unique identifier for each Centralized Logging with OpenSearch deployment
- **Timestamp** - Data-collection timestamp
- **Version** - Solution version deployed
- **Data** - Data includes Region where the solution stack is deployed, request type (whether the stack was created, updated, or deleted), template type used, log pipeline parameters, log pipeline alarm parameters, and OpenSearch version and size.

Examples of collected data by type:

Stack Deployment Data

```
{ "Region": "us-east-1", "RequestType": "Create", "Template": "CentralizedLogging" }
```

Pipeline Management Data

```
{ "metricType": "PIPELINE_MANAGEMENT", "isCrossAccountIngestion": false, "sourceType":
 "RDS", "engineType": "OpenSearch", "logProcessorType": "AWS Lambda", "pipelineType":
 "Service", "region": "us-east-1", "logSourceType": "S3", "status": "CREATE_COMPLETE",
 "pipelineId": "aee1c87c-2531-4fd1-b600-0c8071556967" }
```

Pipeline Alarm Management Data

```
{ "metricType": "PIPELINE_ALARM_MANAGEMENT", "pipelineType": "APP", "region": "us-
east-1", "operation": "CREATE", "pipelineId": "dc8c1afa-1e22-4397-8c8b-963df8ac688a" }
```

OpenSearch Metrics Data

```
{ "metricType": "OPENSEARCH_METRICS", "proxyEnabled": true, "freeStorageSpace":
 7000, "nodeCount": 2, "region": "us-east-1", "domainVersion": "2.17", "domainId":
 "cb25869992309965be0f36096463bcb9", "clusterUsedSpace": 100 }
```

AWS owns the data gathered through this survey. Data collection is subject to the AWS Privacy Notice. To opt out of this feature, complete the following steps before launching the AWS CloudFormation template.

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

   ```
   AnonymousData:
     SendAnonymizedUsageData:
       Data: Yes
   ```

   to:

   ```
   AnonymousData:
     SendAnonymizedUsageData:
       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 the Automated deployment section of this guide.

Contributors

- Manish Jangid
- Bryce Lee

- Owen Chang
- Haiyun Chen
- Aiden Dai
- Lalit Grover
- Eva Liu
- Robin Luo
- James Ma
- Joe Shi
- Charles Wei
- Ming Xu

Revisions

Publication date: *March 2023*.

Check the CHANGELOG.md file in the GitHub repository to see all notable changes and updates to the software. The changelog provides a clear record of improvements and fixes for each version.

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents AWS current product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers, or licensors. AWS products or services are provided "as is" without warranties, representations, or conditions of any kind, whether express or implied. AWS responsibilities and liabilities to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.