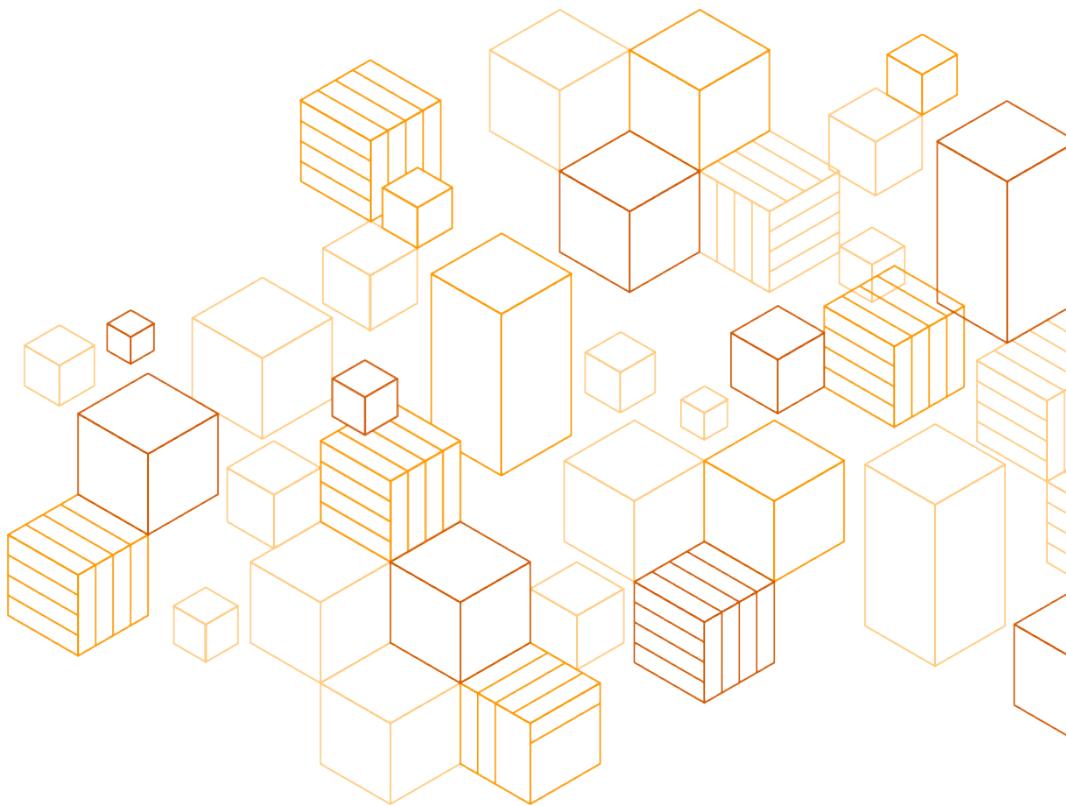


Build a Log Analytics Solution on AWS

September 2021



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 current AWS 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. The responsibilities and liabilities of AWS 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.

© 2021 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Contents

Introduction.....	1
Architecture.....	1
Estimate Your Costs.....	2
Services Used and Costs.....	2
Tutorial	5
Step 1: Set Up Prerequisites	6
Step 2: Create an Amazon Kinesis Data Firehose Delivery Stream	12
Step 3: Install and Configure the Amazon Kinesis Agent on the EC2 Instance	13
Step 4: Create an OpenSearch domain in Amazon OpenSearch Service (successor to Amazon Elasticsearch Service)	15
Step 5: Create a Second Amazon Kinesis Data Firehose Delivery Stream	18
Step 6: Create an Amazon Kinesis Data Analytics Application.....	20
Step 7: View the Aggregated Streaming Data.....	25
Step 8: Clean Up.....	34
Additional Resources.....	36

About this Guide

Log analytics is a common big data use case that allows you to analyze log data from websites, mobile devices, servers, sensors, and more for a wide variety of applications such as digital marketing, application monitoring, fraud detection, ad tech, games, and IoT. In this project, you use Amazon Web Services to build an end-to-end log analytics solution that collects, ingests, processes, and loads both batch data and streaming data, and makes the processed data available to your users in analytics systems they are already using and in near real-time. The solution is highly reliable, cost-effective, scales automatically to varying data volumes, and requires almost no IT administration.

Introduction

[Amazon Kinesis Data Analytics](#) is the easiest way to process streaming data in real time with standard SQL without having to learn new programming languages or processing frameworks. Amazon Kinesis Data Analytics enables you to create and run SQL queries on streaming data so that you can gain actionable insights and respond to your business and customer needs promptly.

This tutorial walks you through the process of ingesting streaming log data, aggregating that data, and persisting the aggregated data so that it can be analyzed and visualized. You create a complete end-to-end system that integrates several AWS services. You analyze a live stream of Apache access log data and aggregate the total request for each HTTP response type every minute. To visualize this data in near real-time, you use a user interface (UI) tool that charts the results.

Architecture

One of the major benefits to using Amazon Kinesis Data Analytics is that an entire analysis infrastructure can be created with a serverless architecture. The system created in this tutorial implements [Amazon Kinesis Data Firehose](#), [Amazon Kinesis Data Analytics](#), and [Amazon OpenSearch Service \(successor to Amazon Elasticsearch Service\)](#). Each of these services is designed for seamless integration with one another. The architecture is depicted below.

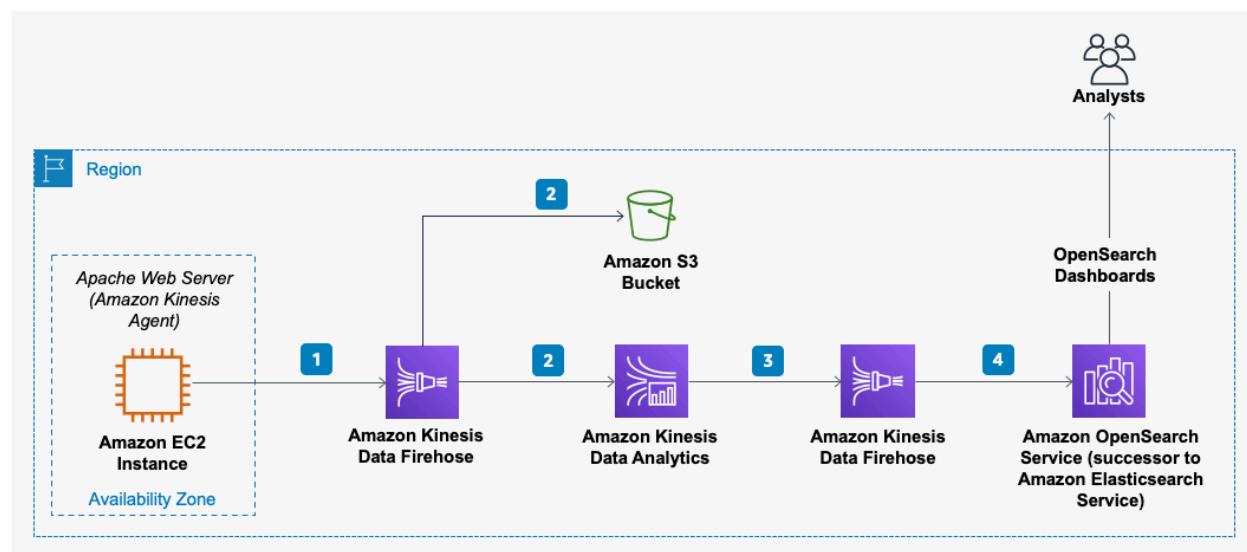


Figure 1: Log analytics solution architecture

In this architecture example, the web server is an Amazon Elastic Compute Cloud ([Amazon EC2](#)) instance. You install the Amazon Kinesis Agent on this Linux instance.

1. The Kinesis Agent continuously forward log records to an Amazon Kinesis Data Firehose delivery stream.
2. Amazon Kinesis Data Firehose writes each log record to Amazon Simple Storage Service ([Amazon S3](#)) for durable storage of the raw log data. Amazon Kinesis Data Analytics continuously runs a SQL statement against the streaming input data.
3. Amazon Kinesis Analytics creates an aggregated data set every minute and outputs that data to a second Kinesis Data Firehose delivery stream.
4. This Firehose delivery stream writes the aggregated data to an OpenSearch domain hosted in Amazon OpenSearch Service.
5. You create a view of the streaming data using OpenSearch Dashboards to visualize the output of your system.

Estimate Your Costs

The total cost of analyzing your Apache access logs varies depending on several factors, including the following:

- how many web log records you ingest
- the complexity of your Amazon Kinesis Analytics SQL queries
- the instance size, storage choice, and redundancy chosen for the OpenSearch domain

This tutorial also creates an EC2 instance to generate a sample Apache access log. The instance size you choose and the amount of time that the instance is running affects the cost.

It costs approximately \$0.51 to complete the tutorial if you use the default configuration recommended in this guide. This estimate assumes that the infrastructure you create during the tutorial is running for 1 hour. A breakdown of the services used and their associated costs is provided in the following section.

Services Used and Costs

AWS pricing is based on your usage of each individual service. The total combined usage of each service creates your monthly bill. For this tutorial, you are charged for the use of Amazon EC2, Amazon Kinesis Data Firehose, Amazon S3, Amazon Kinesis Data Analytics, and Amazon OpenSearch Service.

Amazon EC2

Description: Amazon EC2 provides the virtual application servers, known as instances, to run your web application on the platform you choose. Amazon EC2 allows you to configure and

scale your compute capacity easily to meet changing requirements and demand. It is integrated into Amazon's computing environment, allowing you to leverage the AWS suite of services.

How Pricing Works: Amazon EC2 pricing is based on four components: the instance type you choose (EC2 comes in 40+ types of instances with options optimized for compute, memory, storage, and more), the AWS Region your instances are based in, the software you run, and the pricing model you select (on-demand instances, reserved capacity, spot, etc.). For more information, see [Amazon EC2 pricing](#).

Example: Assume your log files reside on a single Linux t2.micro EC2 instance in the US East region. With an on-demand pricing model, the monthly charge for your virtual machine is \$4.18. For this tutorial, assuming that the log generating instance runs for 1 hour, your EC2 cost is estimated to be **\$0.0116** [= (\$8.35 per month / 30 days per month / 24 hours per day) * 1 hour].

Amazon Kinesis Data Firehose

Description: Amazon Kinesis Data Firehose is a fully managed service for delivering real-time streaming data to destinations such as Amazon S3, Amazon Redshift, or Amazon OpenSearch Service. With Kinesis Data Firehose, you do not need to write any applications or manage any resources. You configure your data producers to send data to Firehose and it automatically delivers the data to the destination that you specified.

How Pricing Works: Amazon Kinesis Data Firehose pricing is based on the volume of data ingested into Amazon Kinesis Data Firehose, which is calculated as the number of data records you send to the service, times the size of each record, rounded up to the nearest 5 KB. For example, if your data records are 42 KB each, Amazon Kinesis Data Firehose counts each record as 45 KB of data ingested. In the US East AWS Region, the price for Amazon Kinesis Data Firehose is \$0.029 per GB of data ingested and decreases as data total increases. For more information, see [Amazon Kinesis Firehose Pricing](#).

Example: In this tutorial, you create two separate Amazon Kinesis Data Firehose delivery streams. One delivery stream receives the data from your Apache access log producer, and the other delivery stream receives the output from an Amazon Kinesis Data Analytics application. Assume the producer sends 500 records per second, and that each record is less than 5 KB in size (typical for an Apache access log record). The monthly estimate for data ingestion into the Kinesis Data Firehose delivery stream in this example is:

- The price in the US East region is \$0.029 per GB of data ingested.
- Record size, rounded up to the nearest 5 KB = 5 KB
- Data ingested (GB per sec) = (500 records/sec * 5 KB/record) / 1,048,576 KB/GB = 0.002384 GB/sec
- Data ingested (GB per month) = 30 days/month * 86,400 sec/day * 0.002384 GB/sec = 6,179.81 GB/month
- Monthly charge: 6,179.81 * \$0.029/GB = \$179.21

For this tutorial, assume that the system is only ingesting data for 1 hour. The cost specifically for this tutorial would be approximately **\$0.25** [= (\$179.21 per month / 30 days per month / 24 hours per day) * 1 hour].

The second Kinesis Data Firehose delivery stream is receiving records at a much less frequent rate. Because the Amazon Kinesis Data Analytics application is outputting only a few rows of data every minute, the cost for that delivery stream is correspondingly smaller. Assuming only five records per minute are ingested, and each record is less than 5 KB, the cost for the delivery stream is **\$0.00005** for the 1-hour duration assumed for this tutorial.

Amazon S3

Description: Amazon S3 provides secure, durable, and highly-scalable cloud storage for the objects that make up your application. Examples of objects you can store include source code, logs, images, videos, and other artifacts that are created when you deploy your application.

Amazon S3 makes it is easy to use object storage with a simple web interface to store and retrieve your files from anywhere on the web, meaning that your website will be reliably available to your visitors.

How Pricing Works: Amazon S3 pricing is based on five components: the type of Amazon S3 storage you use, the region you store your WordPress content (e.g., US East vs. Asia Pacific - Sydney), the amount of data you store, the number of requests you or your users make to store new content or retrieve the content, and the amount of data that is transferred from Amazon S3 to you or your users. For more information, see [Amazon S3 Pricing](#).

Example: Using Standard Storage in the US East Region, if you store 5 GB of content, you pay \$0.115 per month. If you created your account in the past 12 months, and you are eligible for the [AWS Free Tier](#), you pay \$0.00 per month. For this tutorial, assume that the producer creates 5 GB of data. Over a 1-hour period, the total cost for storing the records in Amazon S3 is **\$0.00016/** [= (\$0.115 per month / 30 days per month / 24 hours per day) * 1 hour].

Amazon Kinesis Data Analytics

Description: Amazon Kinesis Data Analytics is the easiest way to process and analyze streaming data in real time with ANSI standard SQL. It enables you to read data from Amazon Kinesis Data Streams and Amazon Kinesis Data Firehose, and build stream processing queries that filter, transform, and aggregate the data as it arrives. Amazon Kinesis Data Analytics automatically recognizes standard data formats, parses the data, and suggests a schema, which you can edit using the interactive schema editor. It provides an interactive SQL editor and stream processing templates so you can write sophisticated stream processing queries in just minutes. Amazon Kinesis Data Analytics runs your queries continuously, and writes the processed results to output destinations such as Amazon Kinesis Data Streams and Amazon Kinesis Data Firehose, which can deliver the data to Amazon S3, Amazon Redshift, and Amazon OpenSearch Service. Amazon Kinesis Data Analytics automatically provisions, deploys, and scales the resources required to run your queries.

How Pricing Works: With Amazon Kinesis Data Analytics, you pay only for what you use. You are charged an hourly rate based on the average number of Kinesis Processing Units (KPU) used to run your stream processing application.

A single KPU is a unit of stream processing capacity comprised of 4 GB memory, 1 vCPU compute, and corresponding networking capabilities. As the complexity of your queries varies, and the demands on memory and compute vary in response, Amazon Kinesis Data Analytics automatically and elastically scales the number of KPUs required to complete your analysis. There are no resources to provision and no upfront costs or minimum fees associated with Amazon Kinesis Analytics. For more information, see [Amazon Kinesis Data Analytics Pricing](#).

Example: This example assumes that the system is running for 1 hour in the US East Region. The SQL query in this tutorial is basic and does not consume more than one KPU. Given that the price for Amazon Kinesis Data Analytics in US East is \$0.11 per KPU-hour, and the tutorial runs for 1 hour, the total cost for the usage of Amazon Kinesis Data Analytics is **\$0.11**.

Amazon OpenSearch Service

Description: Amazon OpenSearch Service makes it easy for you to perform interactive log analytics, real-time application monitoring, website search, and more. Amazon OpenSearch Service offers the latest versions of OpenSearch, support for 19 versions of Elasticsearch (1.5 to 7.10 versions), and visualization capabilities powered by OpenSearch Dashboards and Kibana (1.5 to 7.10 versions).

How Pricing Works: With Amazon OpenSearch Service, you pay only for what you use. There are no minimum fees or upfront commitments. You are charged for Amazon OpenSearch Service instance hours, an [Amazon Elastic Block Store \(Amazon EBS\)](#) volume (if you choose this option), and standard data transfer fees. For more information, see [Amazon OpenSearch Service Pricing](#).

Example: For this tutorial, the total Amazon OpenSearch Service cost can be calculated as follows:

In the **US East (N. Virginia)** Region, an on demand instance type of **r6g.large.search** costs \$0.167 per hour * 1 hour = **\$0.167**.

Tutorial

You can evaluate the simplicity and effectiveness of Amazon Kinesis Data Analytics with this tutorial, which walks you through a simple Amazon Kinesis Data Analytics application.

You perform the following steps in this tutorial:

[Step 1: Set Up Prerequisites](#)

[Step 2: Create an Amazon Kinesis Data Firehose Delivery Stream](#)

[Step 3: Install and Configure the Amazon Kinesis Agent on the EC2 Instance](#)

[Step 4: Create an Opensearch Domain in Amazon OpenSearch Service](#)

[Step 5: Create a Second Amazon Kinesis Data Firehose Delivery Stream](#)

[Step 6: Create an Amazon Kinesis Data Analytics Application](#)

[Step 7: View the Aggregated Streaming Data](#)

Step 8: Clean Up

This tutorial is not meant for production environments and does not discuss options in depth. After you complete the steps, you can find more in-depth information to create your own Amazon Kinesis Data Analytics application in the [Additional Resources](#) section.

Step 1: Set Up Prerequisites

Before you begin analyzing your Apache access logs with Amazon Kinesis Data Analytics, make sure you complete the following prerequisites.

This tutorial assumes that the AWS resources have been created in the US East AWS Region (us-east-1).

Create an AWS Account

If you already have an AWS account, you can skip this prerequisite and use your existing account. To create AWS account:

1. Go to <http://aws.amazon.com/>.
2. Choose **Create an AWS Account** and follow the instructions.

Part of the signup procedure involves receiving a phone call and entering a PIN using the phone keypad.

Start an EC2 Instance

The steps outlined in this tutorial assume that you are using an EC2 instance as the web server and log producer. (For detailed instructions, see [Getting started with Amazon EC2 Linux instances](#).)

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>
2. From the console dashboard, choose **Launch Instance**.
3. On the **Choose an Amazon Machine Image (AMI)** page, choose the **Community AMIs** tab on the lefthand column. In the search box, type **Amazon Linux AMI**. Choose amzn-ami-hvm Amazon Linux AMI (the second one).
4. On the **Choose an Instance Type** page, select the **t2.micro** instance type.

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and how they can meet your computing needs.

Family	Type	vCPUs	Memory (GiB)	Instance Storage (GiB)	EBS-Optimized Available	Network Performance	IPv6 Support
General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
General purpose	t2.micro <small>(Variable ECUs, 1 vCPU, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)</small>	1	1	EBS only	-	Low to Moderate	Yes
General purpose	t2.small	1	2	EBS only	-	Low to Moderate	Yes
General purpose	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
General purpose	t2.large	2	8	EBS only	-	Low to Moderate	Yes
General purpose	t2.xlarge	4	16	EBS only	-	Moderate	Yes
General purpose	t2.2xlarge	8	32	EBS only	-	Moderate	Yes
General purpose	t3.nano	2	0.5	EBS only	Yes	Up to 5 Gigabit	Yes
General purpose	t3.micro	2	1	EBS only	Yes	Up to 5 Gigabit	Yes
General purpose	t3.small	2	2	EBS only	Yes	Up to 5 Gigabit	Yes
General purpose	t3.medium	2	4	EBS only	Yes	Up to 5 Gigabit	Yes
General purpose	t3.large	2	8	EBS only	Yes	Up to 5 Gigabit	Yes
General purpose	t3.xlarge	4	16	EBS only	Yes	Up to 5 Gigabit	Yes
General purpose	t3.2xlarge	8	32	EBS only	Yes	Up to 5 Gigabit	Yes

Next Step: Configure Instance Details

Figure 2: Choose an instance type

5. Choose **Next: Configure Instance Details**.
6. Choose **Create new IAM role**. A new tab opens to create the role.

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances	1	Launch into Auto Scaling Group
Purchasing option	<input type="checkbox"/> Request Spot instances	
Network	vpc-c32c5ab9 (default) <input type="button" value="Create new VPC"/>	
Subnet	No preference (default subnet in any Availability Zone) <input type="button" value="Create new subnet"/>	
Auto-assign Public IP	Use subnet setting (Enable)	
Placement group	<input type="checkbox"/> Add instance to placement group	
Capacity Reservation	Open <input type="button" value="Create new Capacity Reservation"/>	
IAM role	None <input style="border: 2px solid red; background-color: #fff; color: #0072bc; padding: 2px 10px; border-radius: 5px; font-weight: bold; margin-left: 10px;" type="button" value="Create new IAM role"/>	
Shutdown behavior	Stop	
Stop - Hibernate behavior	<input type="checkbox"/> Enable hibernation as an additional stop behavior	
Enable termination protection	<input type="checkbox"/> Protect against accidental termination	
Monitoring	<input type="checkbox"/> Enable CloudWatch detailed monitoring <small>Additional charges apply.</small>	
Tenancy	Shared - Run a shared hardware instance <input type="button" value="Additional charges will apply for dedicated tenancy."/>	
Elastic Inference	<input type="checkbox"/> Add an Elastic Inference accelerator <small>Additional charges apply.</small>	
T2/T3 Unlimited	<input type="checkbox"/> Enable <small>Additional charges may apply</small>	
File systems	<input type="button" value="Add file system"/> <input type="button" value="Create new file system"/>	
Advanced Details		

Next Step: Add Storage

Figure 3: Configure instance details

You want to ensure that your EC2 instance has an AWS Identity and Access Management (IAM) role configured with permission to write to Amazon Kinesis Data Firehose and Amazon CloudWatch. For more information, see [IAM Roles for Amazon EC2](#).

- Choose **Create role**.
- For trusted entity, choose **AWS service**.
- For the use case, choose **EC2**.

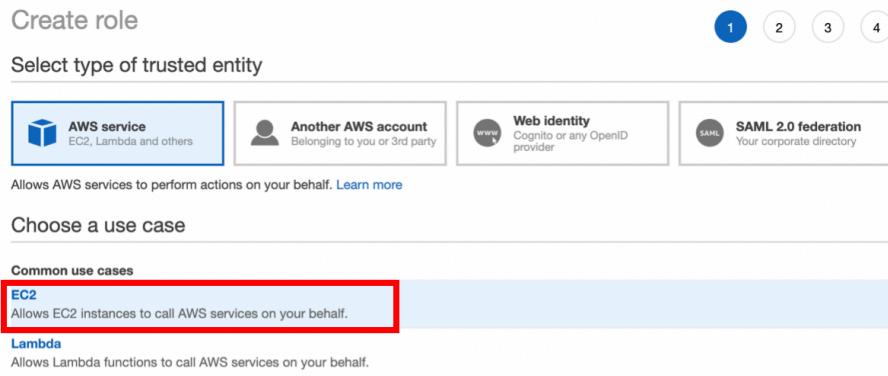


Figure 4: Create new IAM role

- Choose **Next: Permissions**.
- In the search bar, type **KinesisFirehose** and select the check box for **AmazonKinesisFirehoseFullAccess**.

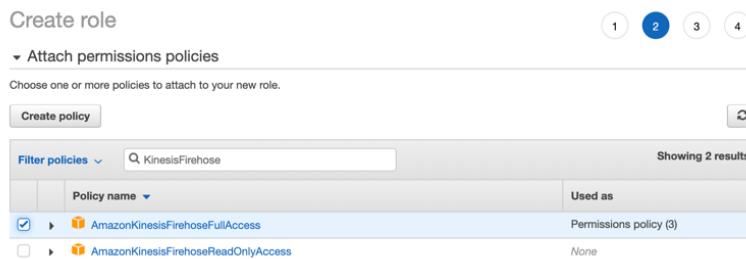


Figure 5: Add AmazonKinesisFirehoseFullAccess policy

- Clear the search bar and type **CloudWatchFull**. Select the check box for **CloudWatchFullAccess**.

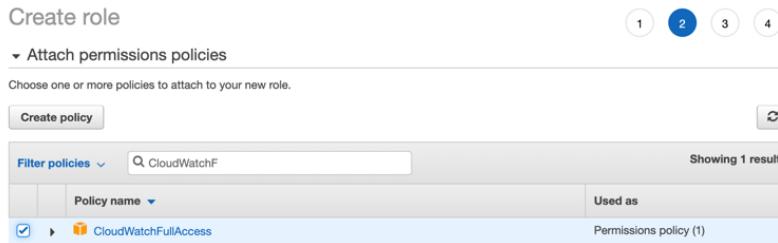
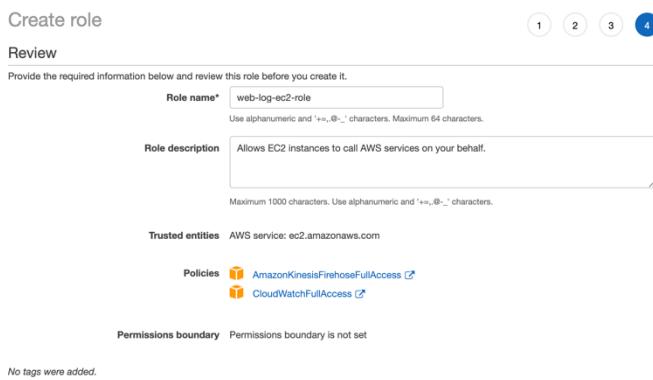


Figure 6: Add CloudWatchFullAccess policy

- f. Choose **Next: Tags** to add optional tags.
- g. Choose **Next: Review** and for **Role name**, type *web-log-ec2-role*.



- h. Choose **Create role**.
8. **Return** to the EC2 launch wizard tab (*Figure 3*) and next to the **IAM role**, click the **refresh** icon. Then, select **web-log-ec2-role**.
9. Choose **Advanced Details** and fill out the **User data** field:

To prepare your EC2 instance, copy and paste the following user data script into the **User data** space. Make sure the lines are single-spaced with no extra whitespace in between.

```
#!/bin/bash
sudo yum update -y
sudo yum install git -y
sudo easy_install pip
sudo pip install pytz
sudo pip install numpy
sudo pip install faker
sudo pip install tzlocal
git clone https://github.com/kiritbasu/Fake-Apache-Log-
Generator.git
mkdir /tmp/logs
cp /Fake-Apache-Log-Generator/apache-fake-log-gen.py
/tmp/logs/
```



Figure 7: Advanced Details – User data field

10. Choose **Review and Launch**.
11. Review the details and choose **Launch**.
12. In the key pair dialog box that appears, choose to create a new key pair or select an existing key pair that you have access to. If you choose to create a new key pair, choose **Download Key Pair** and wait for the file to download.

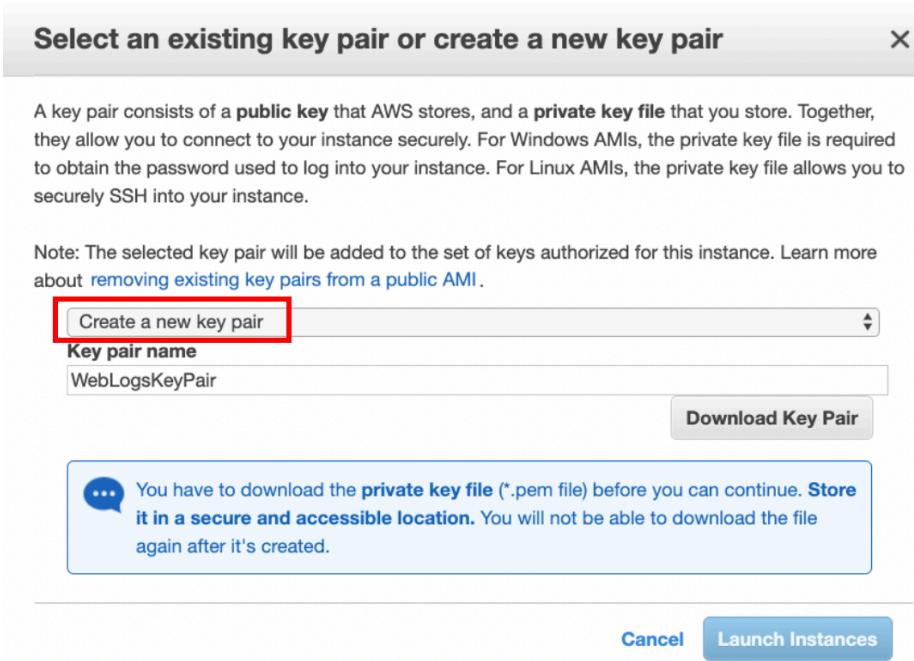


Figure 8: Create a new key pair

13. Choose Launch Instances.

The EC2 instance launches with the required dependencies already installed on the machine. The user data script clones Github onto the EC2 instance and the required file is copied into a new directory named `logs` in the `/tmp` folder. Once the EC2 instance is launched, you need to connect to it via SSH.

Prepare Your Log Files

Because Amazon Kinesis Data Analytics can analyze your streaming data in near real time, this tutorial is much more effective when you use a live stream of Apache access log data. If your EC2 instance is not serving HTTP traffic, you need to generate continuous sample log files. To create a continuous stream of log file data on your EC2 instance, download, install, and run the [Fake Apache Log Generator](#) from Github on the EC2 instance. Follow the instructions on the project page and configure the script for infinite log file generation.

Connect to Your Instance

To connect to your instance, follow the steps in [Connect to Your Linux Instance](#). On the Instances dashboard, you can also select the checkbox next to the instance and choose **Connect** on the top header to view further guidance on how to connect to the instance. The **SSH client** tab lists instructions on how to connect via SSH, using the Key Pair you created.

Once you connect into the EC2 instance, move to the /tmp/logs directory and run the following line of code to start the Fake Apache Log Generator program. Run this line of code multiple times to create multiple log files within the /tmp/logs file.

```
sudo python /tmp/logs/apache-fake-log-gen.py -n 0 -o LOG &
```

Take note of the path to the log file. You need this information later in this tutorial.

Step 2: Create an Amazon Kinesis Data Firehose Delivery Stream

In [Step 1](#), you created log files on your web server. Before they can be analyzed with Amazon Kinesis Data Analytics ([Step 6](#)), you must first load the log data into AWS. Amazon Kinesis Data Firehose is a fully managed service for delivering real-time streaming data to destinations such as Amazon Simple Storage Service (Amazon S3), Amazon Redshift, or Amazon OpenSearch Service.

In this step, you create an Amazon Kinesis Data Firehose delivery stream to save each log entry in Amazon S3 and to provide the log data to the Amazon Kinesis Data Analytics application that you create later in this tutorial.

To create the Amazon Kinesis Data Firehose delivery stream:

14. Open the Amazon Kinesis console at <https://console.aws.amazon.com/kinesis>.
15. In the **Get Started** section, choose **Kinesis Data Firehose**, and then choose **Create Delivery Stream**.
16. On the **Name and source** screen:
 - a. For **Delivery stream name**, enter *web-log-ingestion-stream*.
 - b. For **Choose a source**, select **Direct PUT or other sources**.
 - c. Choose **Next**.
17. On the **Process records** screen, keep the default selections and choose **Next**.
18. On the **Choose a destination** screen:
 - a. For **Destination**, choose **Amazon S3**.
 - b. For **S3 bucket**, choose **Create new**.

- c. In the **Create S3 bucket** window, for **S3 bucket name**, specify a unique name. You do not need to use the name elsewhere in this tutorial. However, Amazon S3 bucket names are required to be globally unique.
 - d. For **Region**, choose **US East (N. Virginia)**.
 - e. Choose **Create S3 Bucket**.
19. Choose **Next**.
20. On the **Configure settings** screen, scroll down to **Permissions**, and for **IAM role**, choose **Create or update IAM role**.

Permissions

IAM role
Kinesis Data Firehose uses this IAM role for all the permissions that the delivery stream needs. To specify different roles for the different permissions, use the API or the CLI. [Learn more](#)

Create or update IAM role **KinesisFirehoseServiceRole-web-log-ingest-us-east-1**
This creates the role or updates it if it already exists, adds the required policies to it, and enables Kinesis Data Firehose to assume it.

Choose existing IAM role
The role that you choose must have policies that include the permissions that Kinesis Data Firehose needs.

[Cancel](#) [Previous](#) [Next](#)

Figure 9: Permissions – IAM role settings

21. Choose **Next**.
22. Review the details of the Amazon Kinesis Data Firehose delivery stream and choose **Create Delivery Stream**.

Step 3: Install and Configure the Amazon Kinesis Agent on the EC2 Instance

Now that you have an Amazon Kinesis Firehose delivery stream ready to ingest your data, you can configure the EC2 instance to send the data using the Amazon Kinesis Agent software. The agent is a standalone Java software application that offers an easy way to collect and send data to Kinesis Data Firehose. The agent continuously monitors a set of files and sends new data to your delivery stream. It handles file rotation, checkpointing, and retry upon failures. It delivers all of your data in a reliable, timely, and simple manner. It also emits [Amazon CloudWatch](#) metrics to help you better monitor and troubleshoot the streaming process.

The Amazon Kinesis Agent can preprocess records from monitored files before sending them to your delivery stream. It has native support for Apache access log files, which you created in [Step 1](#). When configured, the agent parses log files in the Apache Common Log format and

convert each line in the file to JSON format before sending the files to your Kinesis Data Firehose delivery stream, which you created in [Step 2](#).

1. To install the agent, copy and paste the following command, once SSH'd into the EC2 instance. For more information, see [Download and Install the Agent](#).

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

2. For detailed instructions on how to configure the agent to process and send log data to your Amazon Kinesis Data Firehose delivery stream, see [Configure and Start the Agent](#).

To configure the agent for this tutorial, modify the configuration file located at **/etc/aws-kinesis/agent.json** using the following template.

- Replace `filePattern` with the full-path-to-log-file that represents the path to your log files and a wildcard if you have multiple log files with the same naming convention. For example, it might look similar to: `"/tmp/logs/access_log*"`. The value will be different, depending on your use case.
- Replace `name-of-delivery-stream` with the name of the Kinesis Data Firehose delivery stream you created in [Step 2](#).
- The `firehose.endpoint` is `firehose.us-east-1.amazonaws.com` (default).

```
"firehose.endpoint": "firehose.us-east-1.amazonaws.com",
"flows": [
    {
        "filePattern": "/tmp/logs/access_log*",
        "deliveryStream": "name-of-delivery-stream",
        "dataProcessingOptions": [
            {
                "optionName": "LOGTOJSON",
                "LogFormat": "COMMONAPACHELOG"
            }
        ]
    }
]
```

3. Start the agent manually by issuing the following command:

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

Once started, the agent looks for files in the configured location and send the records to the Kinesis Data Firehose delivery stream.

Step 4: Create an OpenSearch Domain in Amazon OpenSearch Service

The data produced by this tutorial is stored in Amazon OpenSearch Service for later visualization and analysis. To create the OpenSearch domain:

1. Open the Amazon OpenSearch Service console at <https://console.aws.amazon.com/es>.
2. Choose **Create a new domain**.
3. On the **Choose deployment type** page, for **Deployment type**, choose a **Development and testing**. For **Version**, choose **OpenSearch 1.0**.
4. Choose **Next**.
5. On the **Configure domain** page, for **Domain name**, type *web-log-summary*.
6. Leave all settings as their default values and choose **Next**.
7. On the **Configure access and security page**:
 - a. For **Network configuration**, choose **Public access**.
 - b. Under **Fine-grained access control**, make sure **Enable fine-grained access control** is selected.
 - c. Choose **Create master user**, and specify the **Master username** as **admin** and set a password.

Fine-grained access control

Fine-grained access control provides numerous features to help you keep your data secure. Features include document-level security, field-level security, read-only users, and OpenSearch Dashboards/Kibana tenants. Fine-grained access control requires a master user.

Set a master user to an IAM account using an ARN, or store a master user in the internal database by creating a master username and password. After your domain is set up, you can use OpenSearch Dashboards/Kibana or the REST APIs to configure additional users and permissions. [Learn more](#)

Enable fine-grained access control

Set IAM ARN as master user
By selecting an IAM ARN as the master user, your domain will only authenticate with IAM roles and users. To access OpenSearch Dashboards/Kibana, enable SAML or Amazon Cognito authentication.

Create master user
By creating a master user, your domain will have the internal user database enabled with HTTP basic authentication.

Master username
Master username must be between 1 and 16 characters.

Master password
Master password must be at least 8 characters long and contain at least one uppercase letter, one lowercase letter, one number, and one special character.

Confirm master password

SAML authentication for OpenSearch Dashboards/Kibana

Figure 10: Fine-grained access control options

- d. In the **Access policy** section, for **Domain access policy**, choose **JSON defined access policy**. Your JSON policy should look like the one shown in Figures 11 or 12.

Note: This is not a recommended setting for production Amazon OpenSearch Service domains. Make sure to terminate this Amazon OpenSearch Service domain after completing the tutorial or apply a more restrictive policy.

Access policy

Access policies control whether a request is accepted or rejected when it reaches the Amazon OpenSearch Service domain. If you specify an account, user, or role in this policy, you must sign your requests. [Learn more](#)

Custom policy builder allows at most 10 elements. Use a JSON-defined access policy to define a policy with more than 10 elements.

Domain access policy Custom access policy

Allow or deny access by AWS account ID, account ARN, IAM user ARN, IAM role ARN, IPv4 address, or CIDR block.

IAM ARN arn:aws:iam::879621327 Allow Remove element

Add element

i IAM-based access policies can conflict with fine-grained access control. [Learn more](#)

Figure 11: Access policy settings

8. Leave all other default settings and choose **Next**. Optionally, add a tag and choose **Next**.
9. Review the details for the Amazon OpenSearch Service domain and choose **Confirm**.

It takes approximately 10 minutes for the Amazon OpenSearch Service domain to be created. While the domain is being created, proceed with the remainder of this guide.

Note: The following example shows a restrictive access policy where the domain is restricted to only allow traffic from a specific IP address.

Add or edit the access policy

```

1  {
2    "Version": "2012-10-17",
3    "Statement": [
4      {
5        "Effect": "Allow",
6        "Principal": {
7          "AWS": "*"
8        },
9        "Action": "es:*",
10       "Resource": "arn:aws:es:us-east-1::domain//*",
11       "Condition": {
12         "IpAddress": {
13           "aws:SourceIp": "/32"
14         }
15       }
16     }
17   ]
18 }
```

Figure 12: Example restrictive access policy

Step 5: Create a Second Amazon Kinesis Data Firehose Delivery Stream

Now that you have somewhere to persist the output of your Amazon Kinesis Data Analytics application, you need a simple way to get your data into your Amazon OpenSearch Service domain. Amazon Kinesis Data Firehose supports Amazon OpenSearch Service as a destination, so create a second Firehose delivery stream:

1. Open the Amazon Kinesis console at <https://console.aws.amazon.com/kinesis>.
2. Choose **Create Delivery Stream**.
3. In the **Source** section:
 - e. For **Choose a source**, select **Direct PUT**.
 - f. For **Destination**, choose **Amazon OpenSearch Service**.
 - g. For **Delivery stream name**, enter *web-log-aggregated-data*.
4. In the **Transform records** section, leave the default values.
5. In the **Destination** settings:
 - h. For OpenSearch domain, choose the domain you created in [Step 4](#). (You may need to wait until the OpenSearch domain has finished processing. Click the refresh button periodically to refresh the list of domains).
 - i. For **Index**, type *request_data*.
 - j. For **Index rotation**, choose **No rotation** (default).
 - k. For **Retry duration**, leave the default value of 300 seconds.
 - l. In the **Back up settings** section, for **Backup mode**, choose **Failed data only**.
 - m. For **S3 backup bucket**, choose **Create**.
 - n. In the **Create S3 bucket** window, for **S3 bucket name**, specify a unique name. You do not need to use the name elsewhere in this tutorial. However, Amazon S3 bucket names are required to be globally unique.
 - o. For **Region**, choose **US East (N. Virginia)**.
 - p. Choose **Create S3 Bucket**.
6. Choose **Next**.
7. On the **Configure settings** screen, you can leave all fields set to their default values. However, you will need to choose an IAM role so that Amazon Kinesis Firehose can write to your OpenSearch domain on your behalf. For **IAM role**, choose **Create new or choose**.

8. Choose **Next**.
9. Review the details for your Amazon Kinesis Data Firehose delivery stream and choose **Create Delivery Stream**.
10. Add permissions for your Kinesis Data Firehose delivery stream to access your OpenSearch cluster:
 - a. Select the newly created **web-log-aggregated-data** stream and choose the **IAM role**, under the **Configuration** tab.

The screenshot shows the IAM role configuration for a Kinesis Data Firehose delivery stream. It consists of three main sections:

- Server-side encryption (SSE)**: Shows "Disabled".
- Destination error logs**: Shows "Enabled".
- Permissions**: Shows the IAM role "KinesisFirehoseServiceRole-web-log-aggre-us-east-1-1632109587669" highlighted with a red box.

Figure 13: Web-log-aggregated-data stream IAM role

- b. In the IAM window that opens, choose **Add inline policy** (on the right).
- c. On the **Create policy** page, choose a service and search for **OpenSearch** in the search box.
- d. Under **Actions**, select the check box for **All OpenSearch Service**.
- e. Expand the **Resources** tab, select the **Specific** radio button and check **Any in this account**. Choose **Review policy**.



Figure 14: Create policy – Any resources check box

- f. On the Review policy page, name the policy *OpenSearchAccess* and choose **Create policy**.

Step 6: Create an Amazon Kinesis Data Analytics Application

You are now ready to create the Amazon Kinesis Data Analytics application to aggregate data from your streaming web log data and store it in your OpenSearch domain. To create the Amazon Kinesis Data Analytics application:

1. Open the Amazon Kinesis Analytics console at <https://console.aws.amazon.com/kinesisanalytics>.
2. Choose **Create new application**.
3. For **Application name**, type *web-log-aggregation-tutorial*.
4. Leave the **Runtime** value as the default and choose **Create application**.
5. To configure the source data for the Amazon Kinesis Data Analytics application, choose **Connect streaming data**.

6. Under **Source**, choose **Kinesis Firehose delivery stream** and select **web-log-ingestion-stream** that you created in [Step 2](#).

Configure source for web-log-aggregation-1

Connect to an existing Kinesis data stream or Data Firehose delivery stream, or easily create and connect to a new demo data stream. Each application can connect to one streaming data source.

Source [Info](#)

Source

Kinesis data stream
Use a data stream for rapid and continuous data intake and aggregation.

Kinesis Data Firehose delivery stream
Use a delivery stream to deliver real-time streaming data to destinations such as Amazon S3, Amazon Redshift, Amazon Elasticsearch Service, or Splunk.

Delivery stream

arn:aws:firehose:us-east-1::deliverystream/web-log-ingestion-stream1 [Browse](#) [Create](#)

Format: arn:aws:firehose:\${Region}:\${AccountId}:deliverystream/\${DeliveryStreamName}

Record preprocessing with AWS Lambda

Kinesis Data Analytics can invoke your Lambda function to preprocess records before they are used in this application. To preprocess records, your Lambda function must be compliant with the required record transformation output model.

Off

On

IAM role for reading source stream

Create or choose IAM role with the required permissions. [Learn more](#)

Create / update IAM role kinesis-analytics-web-log-aggregation-1-us-east-1 with required policies

Choose from IAM roles that Kinesis Data Analytics can assume

Figure 15: Specify the Kinesis Firehose delivery stream

7. Scroll down to the **Schema** section and choose **Discover schema**., Amazon Kinesis Data Analytics analyzes the source data in your Kinesis Data Firehose delivery stream and creates a formatted sample of the input data for your review:

The screenshot shows the 'Schema' tab in the Amazon Kinesis Data Analytics console. A success message 'Schema was successfully discovered.' is displayed. Under 'Schema options', 'Use discovered schema' is selected. In the 'Schema preview' section, the 'Formatted data' tab is active, showing a table with 10 rows of log data. The columns are host, datetime, request, response, and bytes.

host	datetime	request	response	bytes
VARCHAR(16)	VARCHAR(32)	VARCHAR(64)	INTEGER	INTEGER
213.165.46.41	30/Nov/2021:02:...	GET /posts/posts...	200	5034
53.90.111.62	30/Nov/2021:03:...	POST /posts/post...	200	5016
80.39.240.167	30/Nov/2021:04:...	GET /posts/posts...	301	4973
101.237.59.72	30/Nov/2021:05:...	GET /posts/posts...	200	5042
86.93.117.231	30/Nov/2021:06:...	PUT /posts/posts...	200	4978
160.76.186.21	30/Nov/2021:07:...	PUT /posts/posts...	200	4965
53.205.133.40	30/Nov/2021:08:...	PUT /posts/posts...	200	5110
123.88.215.46	30/Nov/2021:08:...	GET /posts/posts...	200	4943
180.47.118.13	30/Nov/2021:09:...	GET /posts/posts...	200	5019
206.52.76.200	30/Nov/2021:09:...	GET /posts/posts...	200	5011

Figure 16: Schema discovery

8. Leave all values set to their defaults, and choose **Save and continue**. You are taken back to the hub screen for your Amazon Kinesis Data Analytics application.
9. To create the SQL that analyzes the streaming data, expand **Steps to configure your application** and choose **Configure SQL**.
10. When prompted, choose **Yes, start application**.

After approximately 60 to 90 seconds, the **Source data** section presents you with a sample of source data that is flowing into your source delivery stream.

11. In the SQL editor, enter the following SQL code:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM"
    (datetime TIMESTAMP, status INTEGER, statusCount
INTEGER);

-- Create pump to insert into output
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
    INSERT INTO "DESTINATION_SQL_STREAM"

-- Select all columns from source stream
SELECT
    STREAM ROWTIME as datetime,
    "response" as status,
    COUNT(*) AS statusCount
        FROM "SOURCE_SQL_STREAM_001"
        GROUP BY
            "response",
            FLOOR(("SOURCE_SQL_STREAM_001".ROWTIME -TIMESTAMP
'1970-01-01 00:00:00') minute / 1 TO MINUTE);
```

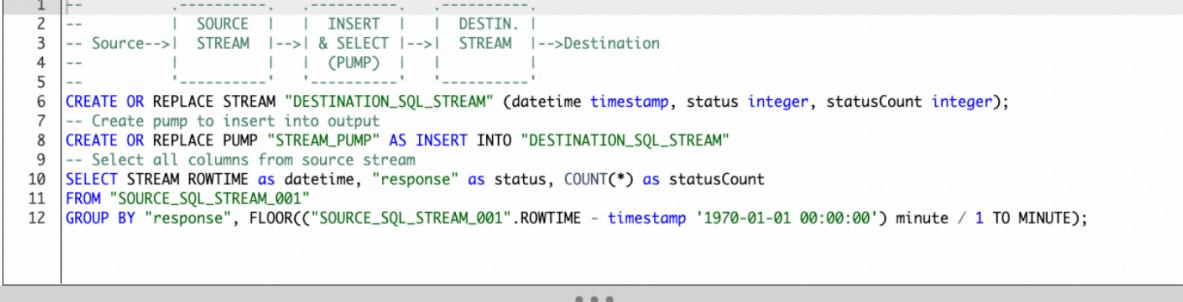
The code creates a STREAM and a PUMP:

- A *stream* (in-application) is a continuously updated entity that you can SELECT from and INSERT into (like a TABLE).
- A *pump* is an entity used to continuously ‘SELECT...FROM’ a source STREAM and INSERT SQL results into an output STREAM.

Finally, an output stream can be used to send results into a destination.

12. Choose **Save and run SQL**. After about 1 minute, Amazon Kinesis Data Analytics displays the output of the query.

13. To save the running output of the query, choose the **Destination** tab and choose **Connect to a destination**.



```

1  |-- .-----.
2  |  | SOURCE |  | INSERT |  | DESTIN. |
3  -- Source-->| STREAM |-->| & SELECT |-->| STREAM |-->Destination
4  --   |  |  | (PUMP) |  |  |
5  |
6 CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (datetime timestamp, status integer, statusCount integer);
7 -- Create pump to insert into output
8 CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
9 -- Select all columns from source stream
10 SELECT STREAM ROWTIME as datetime, "response" as status, COUNT(*) as statusCount
11 FROM "SOURCE_SQL_STREAM_001"
12 GROUP BY "response", FLOOR(("SOURCE_SQL_STREAM_001".ROWTIME - timestamp '1970-01-01 00:00:00') minute / 1 TO MINUTE);

```

Application status: RUNNING

Source	Real-time analytics	Destination																																				
<p>In-application streams:</p> <p><input checked="" type="radio"/> DESTINATION_SQL_STREAM</p> <p><input type="radio"/> error_stream</p> <p style="margin-top: 10px;">Start streaming results</p> <p style="margin-top: 10px;">Download CSV</p>																																						
<p style="margin-bottom: 5px;">Q Filter by column name</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 25%;">ROWTIME</th> <th style="width: 25%;">DATETIME</th> <th style="width: 25%;">STATUS</th> <th style="width: 25%;">STATUSCOUNT</th> </tr> </thead> <tbody> <tr><td>2020-03-19 00:23:00.0</td><td>2020-03-19 00:23:00.0</td><td>200</td><td>13049</td></tr> <tr><td>2020-03-19 00:24:00.0</td><td>2020-03-19 00:24:00.0</td><td>200</td><td>11271</td></tr> <tr><td>2020-03-19 00:24:00.0</td><td>2020-03-19 00:24:00.0</td><td>301</td><td>461</td></tr> <tr><td>2020-03-19 00:24:00.0</td><td>2020-03-19 00:24:00.0</td><td>500</td><td>261</td></tr> <tr><td>2020-03-19 00:24:00.0</td><td>2020-03-19 00:24:00.0</td><td>404</td><td>507</td></tr> <tr><td>2020-03-19 00:25:00.0</td><td>2020-03-19 00:25:00.0</td><td>200</td><td>11232</td></tr> <tr><td>2020-03-19 00:25:00.0</td><td>2020-03-19 00:25:00.0</td><td>301</td><td>506</td></tr> <tr><td>2020-03-19 00:25:00.0</td><td>2020-03-19 00:25:00.0</td><td>404</td><td>496</td></tr> </tbody> </table>			ROWTIME	DATETIME	STATUS	STATUSCOUNT	2020-03-19 00:23:00.0	2020-03-19 00:23:00.0	200	13049	2020-03-19 00:24:00.0	2020-03-19 00:24:00.0	200	11271	2020-03-19 00:24:00.0	2020-03-19 00:24:00.0	301	461	2020-03-19 00:24:00.0	2020-03-19 00:24:00.0	500	261	2020-03-19 00:24:00.0	2020-03-19 00:24:00.0	404	507	2020-03-19 00:25:00.0	2020-03-19 00:25:00.0	200	11232	2020-03-19 00:25:00.0	2020-03-19 00:25:00.0	301	506	2020-03-19 00:25:00.0	2020-03-19 00:25:00.0	404	496
ROWTIME	DATETIME	STATUS	STATUSCOUNT																																			
2020-03-19 00:23:00.0	2020-03-19 00:23:00.0	200	13049																																			
2020-03-19 00:24:00.0	2020-03-19 00:24:00.0	200	11271																																			
2020-03-19 00:24:00.0	2020-03-19 00:24:00.0	301	461																																			
2020-03-19 00:24:00.0	2020-03-19 00:24:00.0	500	261																																			
2020-03-19 00:24:00.0	2020-03-19 00:24:00.0	404	507																																			
2020-03-19 00:25:00.0	2020-03-19 00:25:00.0	200	11232																																			
2020-03-19 00:25:00.0	2020-03-19 00:25:00.0	301	506																																			
2020-03-19 00:25:00.0	2020-03-19 00:25:00.0	404	496																																			

Figure 17: Results of query and Destination tab

14. Under **Destination**, choose **Kinesis Firehose delivery stream** and select the **web-log-aggregated-data** stream that you created in [Step 5](#).
15. For **In-application stream**, choose **DESTINATION_SQL_STREAM**.

Configure destination for web-log-aggregation-1

Point to the destinations where you want the results loaded.

Destination [Info](#)

Delivery stream

Format: arn:aws:firehose:us-east-1:XXXXXXXXXX:deliverystream/web-log-aggregate [Browse](#) [Create](#)

Format: arn:aws:firehose:\${Region}:\${AccountId}:deliverystream/\${DeliveryStreamName}

In-application stream name [Info](#)

In-application streams are continuous flows of data records. You create in-application streams in SQL to contain the data you want to persist to the specified destination.

Connect in-application stream

Choose an existing in-application stream
 Specify a new in-application stream

In-application stream

DESTINATION_SQL_STREAM [▼](#) [C](#)

Output format

JSON
 CSV

[Cancel](#) [Save changes](#)

16. Leave all other options set to their default values and choose **Save changes**.

Step 7: View the Aggregated Streaming Data

After approximately 5 minutes, the output of the SQL statement in your Amazon Kinesis Data Analytics application will be written to your OpenSearch domain. Amazon OpenSearch Service has built-in support for [OpenSearch Dashboards](#), a tool that allows users to explore and visualize the data stored in an OpenSearch cluster. To view the output of your Amazon Kinesis Analytics application in OpenSearch Dashboards:

1. Open the Amazon OpenSearch Service console at <https://console.aws.amazon.com/es/>.

2. In the **Domain** column, choose the OpenSearch domain called **web-log-summary** that you created in [Step 4](#).
3. On the **Overview** tab, click the link next to **OpenSearch Dashboards**.
4. Enter the username and password you created in [Step 4](#).

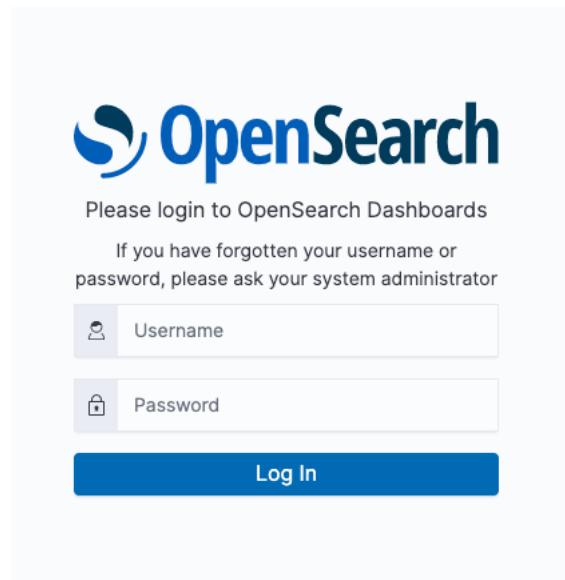


Figure 18: OpenSearch login screen

Because this is the first time you are opening OpenSearch Dashboards in your OpenSearch domain, you need to configure it. This configuration includes giving your user permissions to access the index in OpenSearch and giving Kinesis Data Firehose permission to write to the index.

5. In the Welcome screen, choose Explore on my own. Choose Global for tenant. In the left toolbar, choose **Security** and then choose **Role Mappings**.

The screenshot shows the 'OpenSearch Dashboards' interface under the 'Security' tab. The main heading is 'Get started'. A box contains instructions for defining API calls and data access. It lists two main steps: 'Create roles' and 'Map users'. Step 1, 'Create roles', includes a sub-step for 'Optional: Configure audit logs' and a sub-step for 'Optional: Purge cache'.

Get started

The OpenSearch security plugin lets you define the API calls that users can make and the data they can access. The most basic configuration consists of these steps.

1 Create roles

Roles are reusable collections of permissions. The default roles are a great starting point, but you might need to create custom roles that meet your exact needs. [Learn more](#)

[Explore existing roles](#) [Create new role](#)

2 Map users

After a user successfully authenticates, the security plugin retrieves that user's roles. You can map roles directly to users, but you can also map them to backend roles. [Learn more](#)

[Map users to a role](#) [Create internal user](#)

Optional: Configure audit logs

Audit logs let you track user access to your OpenSearch cluster and are useful for compliance purposes. [Learn more](#)

[Review Audit Log Configuration](#)

Optional: Purge cache

By default, the security plugin caches authenticated users, along with their roles and permissions. This option will purge cached users, roles and permissions.

[Purge cache](#)

Figure 19: Security options in OpenSearch Dashboards

6. Verify an **Open Distro Security Role** named **all_access** role is listed. If it is not listed, choose Create role and choose the Role **all_access**. See Amazon Opensearch Service fine-grained access control [documentation](#) for more information.

You need to modify the **Users** associated with the **all_access** role.

7. Choose the **Mapped users** tab.

The screenshot shows the 'all_access' role configuration in the OpenSearch Dashboards interface. The 'Mapped users' tab is selected. It displays two entries: a user named 'admin' and a backend role ARN. The 'Manage mapping' button is highlighted. A tooltip explains that users can map two types of users: users and backend roles, and that a backend role directly maps to roles through an external authentication system.

User type	User
<input type="checkbox"/> User	admin
<input type="checkbox"/> Backend role	arn:aws:iam:[REDACTED]role/[REDACTED]

Figure 20: Role Mappings

- a. Choose **Manage mapping** and under **Users**, choose **admin**. This is the user you used to log into OpenSearch Dashboards.
- b. Under Backend roles, enter the ARN for the IAM role you are logged into in your AWS account. The ARN can be found in the IAM console page, under **Users**. Click into your user account and copy the **User ARN**.
- c. Repeat this step for the **firehose delivery role** you created earlier, found under **Roles** in the IAM Console, by choosing **Add another backend role**.

You should now have three entries listed for the **all_access** Role.

The screenshot shows the 'Map user' section of the OpenSearch Dashboards interface. At the top, there's a navigation bar with the OpenSearch Dashboards logo, a search bar, and a user icon. Below the navigation, the path 'Security / Roles / all_access / Map user' is displayed. The main area is titled 'Map user' and contains instructions: 'Map users to this role to inherit role permissions. Two types of users are supported: user, and backend role. Learn more'. Under the 'Users' heading, there's a search input field containing 'admin' with an 'X' button and a dropdown arrow, and a blue 'Create new internal user' button. A note below says 'Look up by user name. You can also create new internal user or enter external user.' Under the 'Backend roles' heading, there are two entries: 'arn:aws:iam::[REDACTED]:role/[REDACTED]' and 'arn:aws:iam::[REDACTED]:role/service-role/KinesisFirehose'. Each entry has a red 'Remove' button to its right. Below these, a blue 'Add another backend role' button is visible. At the bottom right are 'Cancel' and 'Map' buttons.

Figure 21: Entries for all_access role

- d. Choose **Map**.
8. Choose the OpenSearch Dashboards icon on the top left to return to the Home dashboard.
9. Choose **OpenSearch Dashboards, Visualize & analyze**.

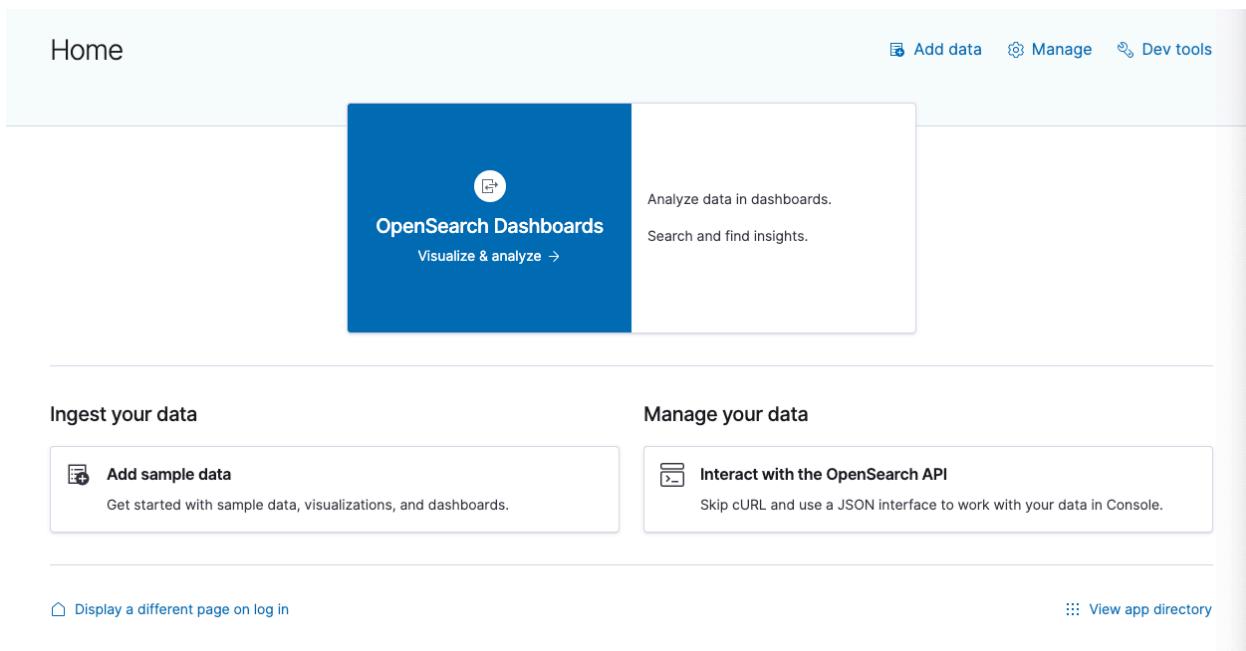


Figure 22: OpenSearch Dashboards dashboard – Add your data

10. Choose **Create index pattern**. In the **Index pattern** field, type *request_data**. This entry uses the OpenSearch index name that you created in [Step 5](#).

This screenshot shows the "Create index pattern" step in the OpenSearch Dashboards interface. The left sidebar has "Index Patterns" selected. The main area has a heading "You have data in OpenSearch. Now, create an index pattern." It explains that an index pattern identifies specific indices to explore. A large blue button at the bottom left says "+ Create index pattern". To the right of the text is a decorative graphic of overlapping circles and arrows. At the bottom, there's a link "Want to learn more? Read documentation".

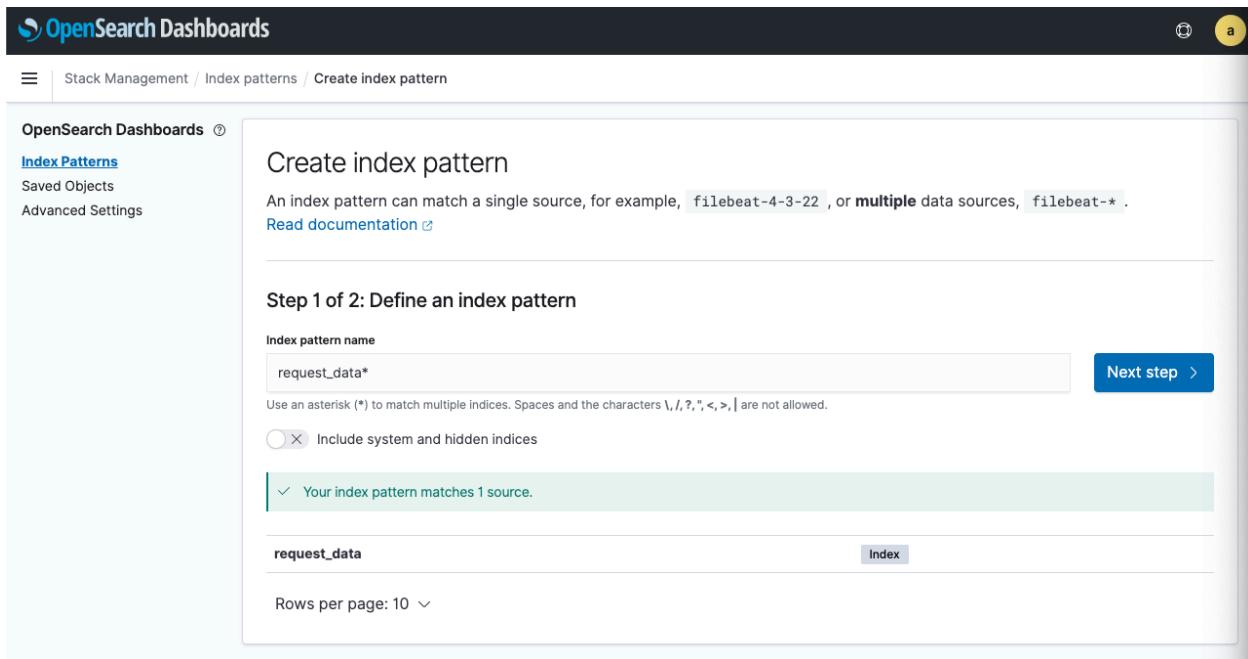


Figure 23: Create index pattern

OpenSearch Dashboards automatically identifies the DATETIME field in your input data, which contains time data.

11. Choose **Next step**.
12. Choose **Create index pattern**.

To visualize the data in your OpenSearch index, you will create and configure a line chart that shows how many of each HTTP response type were included in the source web log data per minute.

To create the line chart:

- a. In the toolbar, choose **Visualize**, and then choose **Create new visualization**.

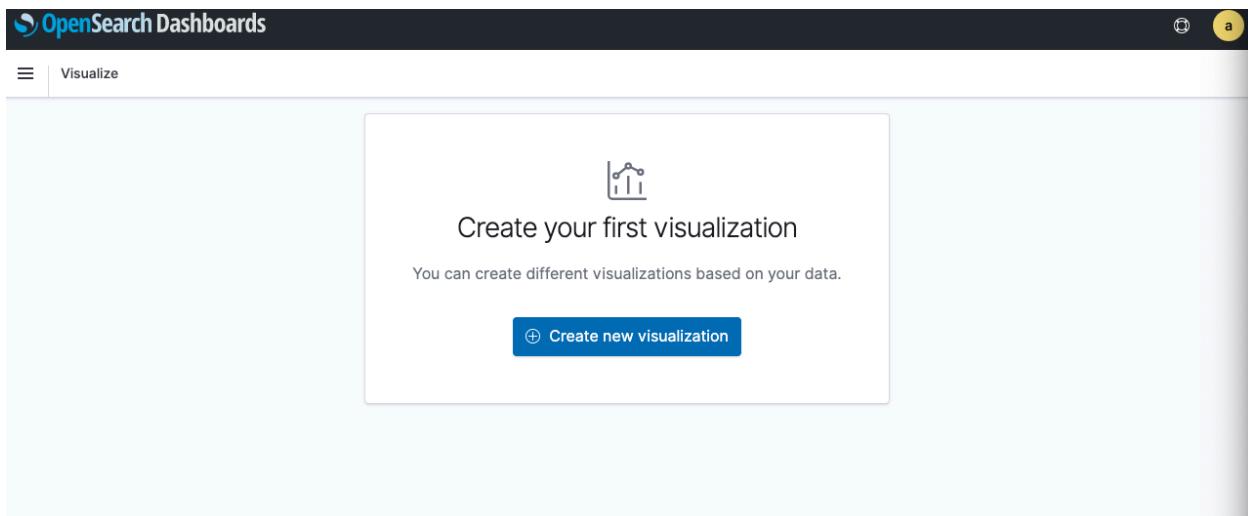


Figure 24: Create new visualization

- b. Choose **Line chart**.
- c. For **Choose a search**, select **request_data***.
To configure your chart, you first need to tell OpenSearch Dashboards what data to use for the y-axis:
- d. In the **metrics** section, choose the arrow next to **Y-Axis**.
- e. Under **Aggregation**, choose **Sum**.
- f. Under **Field**, choose **STATUSCOUNT**.
Now you need to configure the x-axis:
- g. In the **buckets** section, select **X-axis** with the **addition button**.
- h. Under **Aggregation**, choose **Terms**.
- i. Under **Field**, choose **STATUS**.
- j. To run the query and view the line chart, choose **Update** on the bottom right.

The screenshot shows the AWS CloudWatch Metrics Dashboard configuration interface. It is divided into two main sections: **Metrics** and **Buckets**.

Metrics Section:

- Y-axis:** Aggregation is set to **Sum** for the field **STATUSCOUNT**.
- Advanced:** An **Add** button is available.

Buckets Section:

- X-axis:** Aggregation is set to **Terms** for the field **STATUS**.
- Order by:** Metric: Sum of STATUSCOUNT.
- Order:** Descending, Size 5.
- Custom label:** An empty input field.
- Advanced:** Buttons for **Add**, **Group other values in separate bucket**, and **Show missing values**.

Figure 25: Configure X and Y-axis values

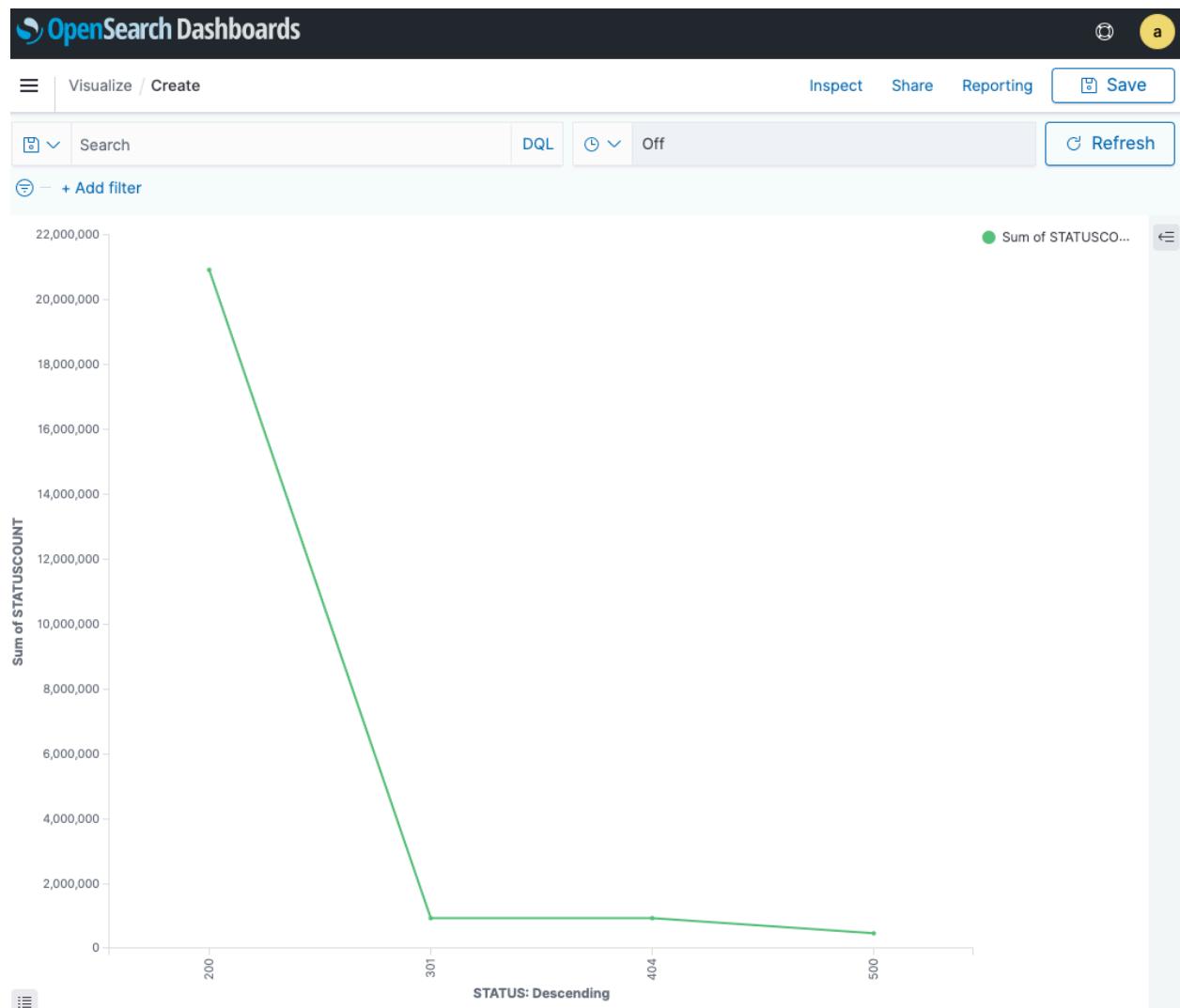


Figure 26: Query results in OpenSearch Dashboards

Step 8: Clean Up

After completing this tutorial, be sure to delete the AWS resources that you created so that you no longer accrue charges.

Terminate the EC2 Instance

If you created a new EC2 instance to generate a continuous stream of Apache access log data, you will need to stop or terminate that instance to avoid further charges.

1. Navigate to the EC2 console at <https://console.aws.amazon.com/ec2>.
2. Select **Running Instances**, and find the instance you used to generate your Apache access logs.

3. On the **Actions** menu, choose the instance, and then choose **Instance State**, then **Terminate**.
4. Read the warning regarding instance termination, and choose **Yes, Terminate**.

If you used an existing EC2 instance with Apache access logs and you do not plan to stop or terminate the instance, you should stop the Amazon Kinesis Agent so that no additional records are sent to the Kinesis Data Firehose delivery stream. Stop the agent with the following command:

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

Delete the OpenSearch domain in Amazon OpenSearch Service

1. Navigate to the Amazon OpenSearch Service console at <https://console.aws.amazon.com/es>
2. Locate and select the domain **web-log-summary** that you created in [Step 4](#).
3. On the **Actions** menu, choose **Delete domain**.
4. On the **Delete domain** confirmation, select the check box and choose **Delete**.

Delete the Amazon S3 Bucket and Bucket Objects

1. Navigate to the Amazon S3 console at <https://console.aws.amazon.com/s3>.
2. Locate the S3 bucket that you created in [Step 2](#).
3. Right-click the bucket name and choose **Delete Bucket**.
4. To confirm the deletion, type the bucket name and choose **Delete**.

Note: At this point in the tutorial, you have terminated or stopped any services that accrue charges while ingesting and processing data. Because the data producer has been stopped, you will not incur additional charges for Amazon Kinesis Data Firehose and Amazon Kinesis Data Analytics since data is not being ingested or processed. You can safely leave them in place for later reference or future development. However, if you wish to remove all resources created in this tutorial, continue with the following steps.

Delete the Amazon Kinesis Data Analytics Application and the Amazon Kinesis Data Firehose Delivery Streams

1. Navigate to the Amazon Kinesis console at <https://console.aws.amazon.com/kinesis>.

2. Choose **Go to Analytics**.
3. Locate and select the name of the Amazon Kinesis Data Analytics application called **web-log-aggregation-tutorial** that you created in [Step 6](#) to view its details.
4. Choose **Application details**.
5. On the **Actions** menu, choose **Delete application**.
6. To confirm the deletion, in the confirmation modal, choose **Delete application**.
7. Navigate to the Amazon Kinesis Data Firehose console at <https://console.aws.amazon.com/firehose>.
8. Choose the Firehose delivery stream called **web-log-ingestion-stream** that you created in [Step 2](#).
9. On the **Actions** menu, choose **Delete**.
10. To confirm the deletion, enter the name of the delivery stream and choose **Delete**.
11. Repeat items 7 through 10 for the second delivery stream called **web-log-aggregated-data** that you created in [Step 5](#).

Additional Resources

We recommend that you continue to learn more about the concepts introduced in this guide with the following resources:

- For detailed information on Amazon Kinesis Analytics, see [Amazon Kinesis Analytics: How It Works](#).
- For information on how to develop your own Amazon Kinesis Analytics application, with specific information about its SQL extensions, windowed queries, and joining multiple streams, see [Streaming SQL Concepts](#).
- For additional examples, see [Example Amazon Kinesis Analytics Applications](#).